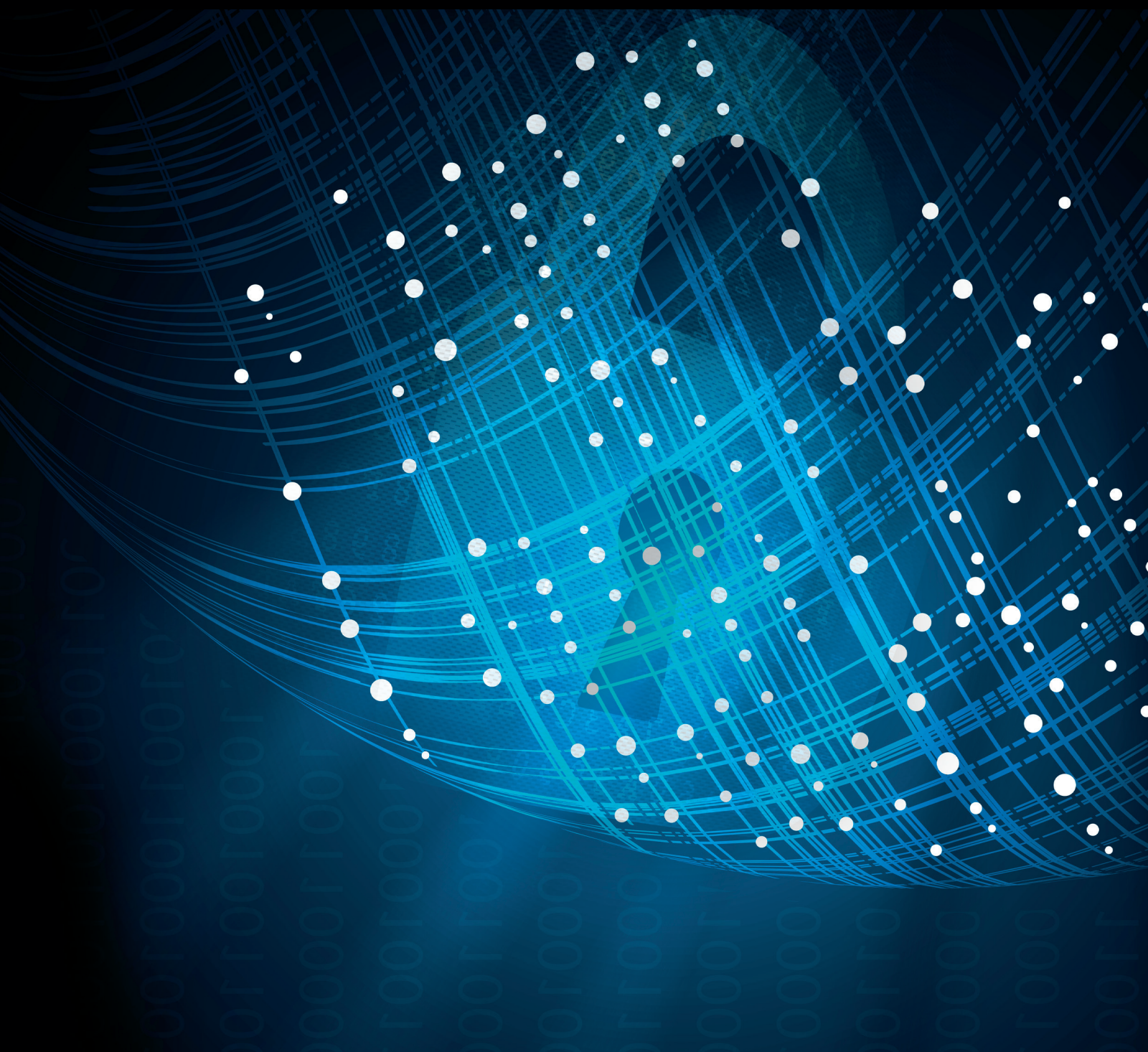
Security and Communication Networks

Big Data Analytics for Cyber Security

Lead Guest Editor: Pelin Angin
Guest Editors: Bharat Bhargava and Rohit Ranchal



Big Data Analytics for Cyber Security

Security and Communication Networks

Big Data Analytics for Cyber Security

Lead Guest Editor: Pelin Angin
Guest Editors: Bharat Bhargava and Rohit Ranchal

Editorial Board

Contents

Hindawi
Security and Communication Networks
Volume 2019, Article ID 4109836, 2 pages
https://doi.org/10.1155/2019/4109836

Editorial

Big Data Analytics for Cyber Security

Pelin Angin,[1] **Bharat Bhargava**,[2] **and Rohit Ranchal**[3]

[1]*Department of Computer Engineering, Middle East Technical University, Ankara, Turkey*
[2]*Department of Computer Science, Purdue University, West Lafayette, IN, USA*
[3]*IBM Watson Health Cloud, Cambridge, MA, USA*

Correspondence should be addressed to Pelin Angin; pangin@ceng.metu.edu.tr

Received 4 August 2019; Accepted 8 August 2019; Published 4 September 2019

The era of Internet of Things with billions of connected devices has created an ever larger surface for cyber attackers to exploit, which has resulted in the need for fast and accurate detection of those attacks. The developments in mobile computing, communications, and mass storage architectures in the past decade have brought about the phenomenon of big data, which involves unprecedented amounts of valuable data generated in various forms at a high speed. The ability to process these massive amounts of data in real time using big data analytics tools brings along many benefits that could be utilized in cyber threat analysis systems. By making use of big data collected from networks, computers, sensors, and cloud systems, cyber threat analysts and intrusion detection/prevention systems can discover useful information in real time. This information can help detect system vulnerabilities and attacks that are becoming prevalent and develop security solutions accordingly.

Big data analytics will be a must-have component of any effective cyber security solution due to the need of fast processing of the high-velocity, high-volume data from various sources to discover anomalies and/or attack patterns as fast as possible to limit the vulnerability of the systems and increase their resilience. Even though many big data analytics tools have been developed in the past few years, their usage in the field of cyber security warrants new approaches considering many aspects including (a) unified data representation, (b) zero-day attack detection, (c) data sharing across threat detection systems, (d) real time analysis, (e) sampling and dimensionality reduction, (f) resource-constrained data processing, and (g) time series analysis for anomaly detection.

This special issue has attracted original contributions that utilize and build big data analytics solutions for cyber security in a variety of fields. All submissions underwent a meticulous review process, and nine papers were accepted for publication in this special issue. The following is a short summary of the findings of each of these papers.

Cyber Physical Power Systems (CPPS) are a critical infrastructure and therefore a favorable target of cyber-attacks. In "VHDRA: A Vertical and Horizontal Intelligent Dataset Reduction Approach for Cyber-Physical Power Aware Intrusion Detection Systems," the authors proposed the use of the Nonnested Generalized Exemplars (NNGE) algorithm and showed that it is among the most accurate and suitable classification methods for developing an intrusion detection system for CPPS because of its ability to classify multiclass scenarios and handle heterogeneous datasets. Furthermore, VHDRA proposed mechanisms to improve the classification accuracy and speed of the NNGE algorithm and reduce the computational resource consumption. It achieves this by vertical reduction of the dataset features by selecting only the most significant features and horizontally reduces the size of data while preserving original key events and patterns within the datasets using the State Tracking and Extraction Method approach.

In "Integrating Traffics with Network Device Logs for Anomaly Detection," the authors presented Traffic-Log Combined Detection (TLCD), which is a multistage intrusion analysis system that overcomes the inefficacy of existing anomaly detection systems that search logs or traffics alone for evidence of attacks but do not perform further analysis of attack processes. TLCD correlates log data with traffic characteristics to reflect the attack process and construct a federated detection platform. Specifically, it can discover the process steps of a cyberattack, reflect the current network status, and reveal the behaviors of normal users.

Experiments with different cyberattacks demonstrated that TLCD provides high accuracy and a low false positive rate.

Role-based access control (RBAC) is a predominant access control model and is widely used in both commercial and research settings. A key requirement of RBAC is to identify appropriate roles that capture business needs. Role mining is a common approach to discover user roles from existing datasets using data mining. The interdependent relationships between user permissions must be considered to prevent security vulnerabilities. In "RMMDI: A Novel Framework for Role Mining Based on the Multi-Domain Information," the authors proposed a role mining framework based on multi-domain information. It utilizes the information from multiple domains such as physical, network, and digital, to find the relationships and similarity between user permissions, and aggregates the interdependent permissions under the same role using multi-view community detection methods.

Governments and enterprises are frequently exposed to coordinated cyberattacks such as advanced persistent threat (APT). Such attacks require exploiting multiple systems within an organization to gain unauthorized access to data for an extended period by staying undetected. Detection and prevention of such attacks requires classifying the disparate data from multiple systems based on its semantics and correlating it through a comprehensive analysis. The article titled "HeteMSD: A Big Data Analytics Framework for Targeted Cyber-Attacks Detection Using Heterogeneous Multisource Data" addresses these gaps by complementing the analysis using human security experts. It presents a multilayer design of the framework and discusses the identification of security related characteristics in the data, classification of data based on degree of security semantics, and different types of correlation analysis.

In "Optimizing Computer Worm Detection Using Ensembles," the authors addressed the problem of detecting computer worms in networks. They focused particularly on the problem of detecting sophisticated computer worms that use code obfuscation techniques and developed a behavioral machine learning model to detect computer worms. The achieved results are promising in terms of accuracy and generalization to new datasets.

In "Malware Detection on Byte Streams of PDF Files Using Convolutional Neural Networks," the authors designed a convolutional neural network to tackle malware detection on PDF files. They collected malicious and benign PDF files and manually labeled the byte sequences within them. The proposed network was designed to interpret high-level patterns among collectable spatial clues, predicting whether the given byte sequence has malicious actions or not. The experimental results showed that the proposed approach outperforms several machine learning models.

Due to the numerous benefits of cloud computing, it is becoming the go-to technology for hosting services and storing data. However, the utilization of cloud brings inherent risks and uncertainty due to lack of visibility into the cloud and loss of control over operations applied to shared data. A key requirement in cloud-based data storage and sharing is to ensure the integrity of shared data. In "Integrity Audit of Shared Cloud Data with Identity Tracking," the authors proposed a public auditing scheme for dynamic group-oriented data sharing in cloud environments. They introduced a new role called Rights Distribution Center (RDC) to track the membership and identity of users. The approach enables performing third party audits to verify data integrity while protecting the privacy of user identity.

Automated data mining can help in extracting important information from unstructured text for various cybersecurity use cases. However, lack of a high-quality large labeled dataset has been a hindrance for information security research. Crowdsourcing can be an effective way to quickly obtain a large labeled dataset at low cost, but the crowd annotations may be of lower quality than those of experts. In "Multifeature Named Entity Recognition in Information Security Based on Adversarial Learning," the authors proposed solutions by first identifying the common features in crowdsourced annotations using generative adversarial networks. Due to the diversity and specificity of the entity categories in cybersecurity, only the basic word and character features can be used, but these features alone are not sufficient for effective named entity recognition. To address this, the domain dictionary and sentence dependency features were used as additional features to again identify the entities and improve the quality of crowdsourcing annotations.

The rise of cloud computing has resulted in data storage and computation being delegated to the untrusted cloud, leading to a series of challenging security and privacy threats. While fully homomorphic encryption can be used to protect the privacy of cloud data and solve the trust problem of a third party, the key problem of achieving fully homomorphic encryption is reducing the increasing noise during the ciphertext evaluation. In "Generalized Bootstrapping Technique Based on Block Equality Test Algorithm," the authors investigated the bootstrapping procedure used to construct a fully homomorphic encryption scheme. They proposed a new block homomorphic equality test algorithm and gave an instance based on the FH-SIMD scheme. Both theoretical analysis and experiment simulation demonstrated the high performance of the proposed bootstrapping algorithm.

Conflicts of Interest

The Editors declare that there are no conflicts of interest.

Pelin Angin
Bharat Bhargava
Rohit Ranchal

Hindawi
Security and Communication Networks
Volume 2019, Article ID 6816943, 15 pages
https://doi.org/10.1155/2019/6816943

Research Article

VHDRA: A Vertical and Horizontal Intelligent Dataset Reduction Approach for Cyber-Physical Power Aware Intrusion Detection Systems

Hisham A. Kholidy[1] **and Abdelkarim Erradi**[2]

[1]*Department of Network and Computer Security (NCS), College of Engineering, State University of New York (SUNY) Polytechnic Institute, 100 Seymour Rd., Utica, NY, USA*
[2]*Computer Science and Engineering Department, Qatar University, Doha, Qatar*

Correspondence should be addressed to Hisham A. Kholidy; hisham_dev@yahoo.com

Received 19 November 2018; Accepted 7 March 2019; Published 17 June 2019

Guest Editor: Rohit Ranchal

The Cypher Physical Power Systems (CPPS) became vital targets for intruders because of the large volume of high speed heterogeneous data provided from the Wide Area Measurement Systems (WAMS). The Nonnested Generalized Exemplars (NNGE) algorithm is one of the most accurate classification techniques that can work with such data of CPPS. However, NNGE algorithm tends to produce rules that test a large number of input features. This poses some problems for the large volume data and hinders the scalability of any detection system. In this paper, we introduce VHDRA, a Vertical and Horizontal Data Reduction Approach, to improve the classification accuracy and speed of the NNGE algorithm and reduce the computational resource consumption. VHDRA provides the following functionalities: (1) it vertically reduces the dataset features by selecting the most significant features and by reducing the NNGE's hyperrectangles. (2) It horizontally reduces the size of data while preserving original key events and patterns within the datasets using an approach called STEM, State Tracking and Extraction Method. The experiments show that the overall performance of VHDRA using both the vertical and the horizontal reduction reduces the NNGE hyperrectangles by 29.06%, 37.34%, and 26.76% and improves the accuracy of the NNGE by 8.57%, 4.19%, and 3.78% using the Multi-, Binary, and Triple class datasets, respectively.

1. Introduction

The CPPS are vital components that require special cybersecurity efforts. The deregulation and multipoint communication between consumers and utilities has introduced more complexities that make power system operation very difficult to manage. The WAMS plays an important role in monitoring and controlling of the CPPS since it provides large volume of information and an efficient communication infrastructure. However, this introduces cyber security vulnerabilities to these systems. Intruders may exploit such vulnerabilities to create cyberattacks against the electric power grid. The CPPS need to be resilient to cyberattacks through a precise and scalable attack classification technique that can deal with the large volume of high speed data provided by the WAMS and facilitate the autonomic control of the complex operation of the CPPS.

Several approaches have been proposed to secure the CPPS systems such as the behavior rule-based monitoring devices methodology [1] in the smart grid that is used to detect the insider threats, the anomaly detection techniques [2], which extract the normal behaviors from various communication protocols of Industrial Control Systems (ICSs) to create a full description of the communication pattern, The Specification-Based IDS [3] that monitors system security states and sends the alerts when the system behavior approaches an unsafe or disallowed state, the common path mining approach [4] that creates an IDS using heterogeneous data for detecting power system cyberattacks using the State Tracking and Extraction Method (STEM) algorithm [5] to

preprocess data and then uses frequent item set mining to extract common paths associated with specific system behaviors, and recently a NNGE with a Hoeffding Adaptive Trees approach [5, 6] that is used to create an offline and online Event Intrusion Detection Systems using STEM to process the power system security datasets. However, these approaches are still neither accurate nor scalable enough to process the high speed big data of the CPPS [5, 7–9]. To build an efficient security framework that well adapt the CPPS, the following security requirements are recommended by several well-established IT security organizations such as the Department of Homeland Security (DHS) [10], SANS [11], Check Point Software [12], and ENISA [13]: (1) the security framework should provide the required tools to perform the classification and detection of attacks, security assessment, and auditing processes in an accurate and fast way [11], (2) the security framework should be integrated with the current IT security solutions such as IDS to provide monitoring and log analysis capabilities, security for network communications, hosts, and control access to physical control systems [4], (3) the architecture of the security framework should be more scalable, robust, and flexible to deal with heterogeneous attacks and heterogeneous CPPS [5], and (4) the detection and response time of the security framework should be short enough to prevent attackers from completing their reconnaissance and/or installing any illicit monitoring or disrupting malware. According to these requirements, a scalable, accurate, and fast classification approach is required to work with the CPPS. The NNGE algorithm is among the most accurate classification techniques that can work with heterogeneous datasets formats such as the WAMS data [5, 7]. Although the accuracy of NNGE as a standalone classification method is lower [7, 14], it is still the most suitable classification method to develop a cypher physical power aware intrusion detection systems because of its ability to classify multiclass scenarios and sequential data and handle heterogeneous datasets formats such as discrete, nominal or symbolic, continuous, and nonvalue features [15–17]. In this work, we introduce a new data reduction approach called VHDRA, Vertical and Horizontal Data Reduction Approach, which includes feature selection, exemplar pruning, feature reduction, and system states compression methods. VHDRA improves the classification accuracy and speed and reduces the computational resource consumption of the NNGE algorithm. It provides the following functionalities.

(1) A vertical reduction for the dataset features by selecting the most significant features: to this target, we developed a new fitness function for the Particle Swarm Optimization (PSO) algorithm [18] that adopts the classification function of the NNGE algorithm through selecting the significant features whose values are closer to a margin of the covering hyper-rectangle.

(2) Pruning of nongeneralized exemplars using the highest ranked features of the PSO: VHDRA uses the Evolutionary Pruning Algorithms (EPA-NNGE) [19] to improve the classification accuracy of NNGE and to reduce the model size by reducing the hyperrectangles and ignoring the nonselected features among the significant ones defined by PSO.

(3) A horizontal reduction for the size of the dataset while preserving original key events and patterns within the datasets using an approach called STEM, State Tracking and Extraction Method [6], which is used to quantize and reduce the heterogeneous datasets to reduce STEM tracks system states from measurements and creates a compressed sequence of states for each observed scenario.

To evaluate the accuracy of the VHDRA, we compare the detection rates of the NNGE using VHDRA against current classification approaches including the NNGE with its best feature selection approach, namely, the Correlation based Feature Selection (CFS) [6]. The comparison uses an existing intrusion detection power grid dataset [15]. To evaluate the improvement in the detection speed and computational resource consumption, we compare the number of reduced exemplars using the NNGE with CFS, VHDRA with the horizontal reduction, VHDRA with the vertical reduction, and VHDRA with both the horizontal and the vertical reduction together. The real time implementation of VHDRA consists of the following three phases: (1) training and configuration phase. At this phase, the important features are extracted from the real time dataset using the modified PSO fitness function, and the k-fold cross-validation algorithm is applied to compute the dataset threshold. Furthermore, the quantization intervals of the STEM are created based on the dataset data boundaries. (2) Online detection phase: at this phase, the integrated PSO and Evolutionary Pruning NNGE (EPA-NNGE) approach is used to classify the online events and fires alerts when an attack is detected. (3) Update phase: at this phase, the quantization intervals and the STEM's states are updated and inserted into the quantized datasets and duplicated values are deleted. The threshold value is also updated and new behaviors and events are added to the dataset. This paper is organized as follows: after Section 1 introduces the NNGE algorithm, the CCPS testbed, and the test datasets, Section 2 surveys the state of the art of the previous attempts to improve the NGGE. After that, Section 3 introduces the improved NNGE algorithm and the VHDRA, then Section 4 discusses the experimental analysis of the results, and, finally, Section 5 concludes the paper and draws the furfure work.

1.1. The NNGE. NNGE [4, 6, 20] is an instance based classifier in which the algorithm creates if then else like rules represented by generalized exemplars. Generalized exemplars may be singles in which case the exemplar represents exactly one example from the training database. Alternatively, generalized exemplars may be hyper rectangles which represent more than one example of the same class from the training database. Training the NNGE algorithm is an incremental process which includes steps named classification, generalization, and dynamic feedback. Each labeled example in the training database is first classified by comparing the new example to all known hyperrectangles and single examples. The classification step in training uses the same methodology

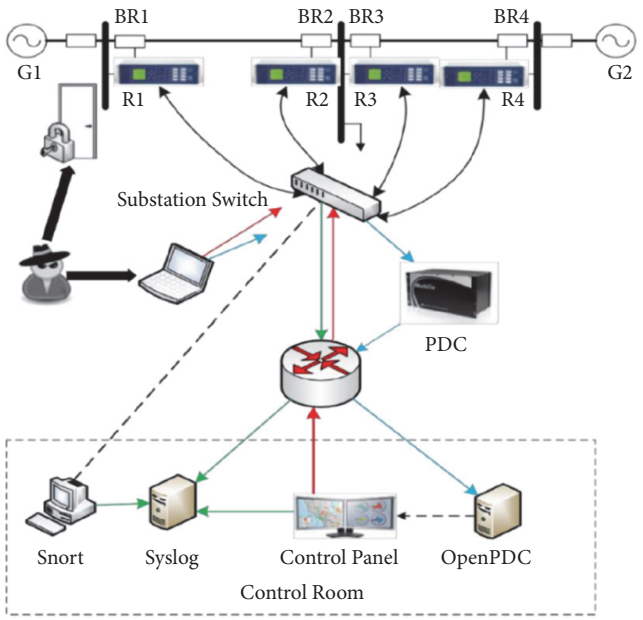


Figure 1: Power system framework for generating test datasets [15].

and distance metric as standalone NNGE classification. Hyperrectangles are generalized rules which represent a class and single examples are previous examples of a class which do not fit into a hyperrectangle. The dynamic feedback is used to adjust feature and exemplar weights used by the distance function.

After training, new examples are classified by calculating the Euclidean distance metric from the example to all exemplars. The new example is classified as the class of the nearest exemplar. The NNGE algorithm preforms generalization by merging exemplars, forming hyper rectangles in attribute space that represent conjunctive rules with internal disjunction. The algorithm forms a generalization each time a new example is added to the database, by joining it to its nearest neighbor of the same class.

1.2. The CPPS Testbed and Datasets. The datasets that we used to evaluate our approach are described in detail in [15]. It consists of synchrophasor measurements from Phasor Measurement Units (PMUs) of four substations. As shown in the testbed diagram of Figures 1 and 2, G1 and G2 are power generators. R1 through R4 are Intelligent Electronic Devices (IEDs) that can switch the breakers on or off. These breakers are labeled BR1 through BR4. Line one spans from breaker one (BR1) to breaker two (BR2) and line two spans from breaker three (BR3) to breaker four (BR4).

Each IED automatically controls one breaker; thus R1 controls BR1, R2 controls BR2, and so on accordingly. The IEDs use a distance protection scheme which trips the breaker on detected faults whether actually valid or faked since they have no internal validation to detect the difference. Operators can also manually issue commands to the IEDs R1 through R4 to trip the breakers BR1 through BR4. The manual override is used when performing maintenance on the lines or other system components. To enhance the cyberattack detection rate, the security attributes such as relay control panel SNORT [21] logs are included in the test datasets. The size of this heterogeneous dataset is approximately 38 Gigabytes and it includes 128 features (e.g., 29 attributes for a single PMU measurement, and four PMUs generate 116 features along with 12 log attributes), which includes nine power system events and 36 cyberattacks. Details of the attributes have been introduced in previous work [5, 6, 15]. The test datasets were generated under 41 scenarios when the power system operated normally (*No Event*) and is distributed by natural faults (*Natural Event*) and cyberattackers (*Attack*). Therefore, the 41 scenarios can be combined into three classes. To characterize and identify normal performance of the power systems, the test datasets were combined as *No Event* and *Natural Event* classes. The dataset is then randomly sampled at one percent and grouped into single Binary Class (i.e., *Normal* and *Attack*), Triple Class (*No Event*, *Natural Event* and *Attack*), and Multiclass(i.e., $1, 2 \ldots 41$). More details of the datasets are found in [15].

1.3. Threat Model and Attacks Scenarios in the Dataset. The dataset has five attacks categories described as follows [15].

(1) *Cyber Command Injection Attacks*. The CPPS Relays are tripped by remote command injection attack. In this attack scenario, a network packet capture tool was used to capture commands used to remotely trip the relay. These commands are replayed on the network from an attacker PC connected to the network switch. In another scenario of this attack in the dataset, insiders physically trip a relay from the face plate.

(2) *Man-In-The-Middle (MITM) Attacks*. These are used to emulate faults and contingencies. The MITM attacks alter power system measurements transmitted from field devices to control room systems. Implemented MITM attacks include a false data injection attack which alters current and voltage phasors and replay attacks which resend captured RTU frames from a previous period. Both of these attacks are used to confuse an operator or automate algorithm monitoring the systems. Faults, generator loss, load changes, and transmission line loss are simulated in the dataset. The MITM attack results in the testbed demonstrate that the attacker randomly alters current phasors in an RTU stream. The range of current for the normal operation in this case is between 200 and 550 Amps. This attack is carried out between the MTU and a control center computer.

(3) *Two Varieties of DOS Attacks Being Used*. First, scripts are available to send high volumes of network traffic (floods) to a network target in attempt to overwhelm network processors and memory. This causes a loss of communication which in turn leads to loss of system monitoring capability. Second, two Python based protocol mutation engines are available to send mutated packets against the IEEE C37.118 protocol [22] that is used for network communication between the CPPS devices. Attacks against this protocol lead to denial of service and a loss of visibility of the state of the power system. The first protocol mutation engine randomly flips bits in network packets. The second protocol mutation engine sends intelligently manipulated packets.

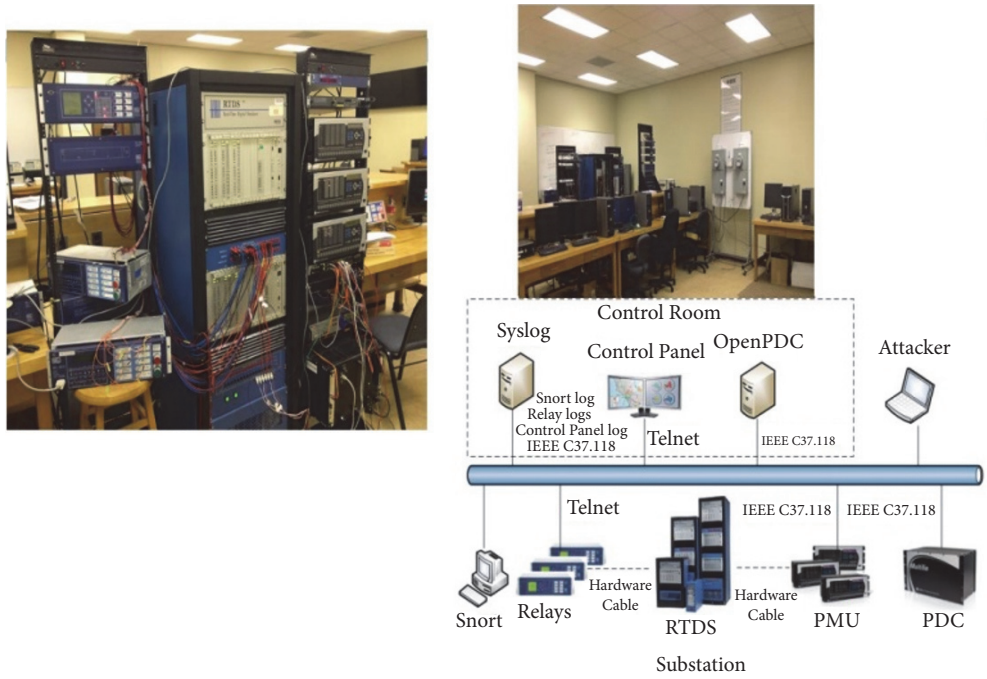


Figure 2: The real-time smart grid testbed.

(4) *A Replay Attack.* This is used to replay an authentication session by an attacker to fool a computer into granting access. This attack appears in the dataset in a form or retransmission of a network data transmission and is usually used to gain authentication in a fraudulent manner.

(5) *Physical HMI and UI Manipulation Attacks.* These are provided to simulate invalid changes to relay settings. These relay settings changes include change threshold and timer values as well as disabling the relay completely. HMI and UI manipulation attacks are automated by an AutoIt script. Such attacks mimic effects of insiders taking illicit control actions and malware taking control of software systems to manipulate control devices. Usually the spoofing, MITM, sniffing, command injection, and DOS attacks lead to these categories of attacks.

2. State of the Art

The recent state of the art refers to several approaches that aim to select feature subsets with maximum discrimination capacity data reduction. We survey these approaches as follows. In [23], authors proposed a feature selection framework that is augmented with a learning method, termed generalized PDF [24] projection theorem (GPPT), to reconstruct the distribution in high-dimensional raw data space from the low-dimensional feature subspace information loss issue in feature reduction. In [25] authors proposed a Bayesian classification approach for automatic text categorization using class-specific features. The proposed approach follows Baggenstoss's PDF projection theorem [24, 26] to reconstruct PDFs in raw data space from the class-specific PDFs in low-dimensional feature space and build a Bayes classification rule. The same approach is extended in [27] using a new divergence measure to measure multidistribution divergence for multiclass classification. However, among the current states of the art, the CFS is the most accurate feature selection approach that works with NNGE [6, 17]. There are several research works that have been conducted to improve the classification accuracy of the NNGE algorithm; in this section, we briefly highlight them. Daniela et al. [19] investigate the ability of an evolutionary pruning mechanism to improve the predictive accuracy of a classifier based on nonnested generalized exemplars. In [28], authors proposed some NNGE variants based on the analyses of the impact of three elements of the NNGE classifier on the classification accuracy of the NNGE algorithm. These elements are the hyperrectangles splitting procedure, the pruning of nongeneralized exemplars, and the presentation order of training instances. In [19] authors used the NNGE to create rules for classifying the attacks by using the Ant-Miner Algorithm. First they created rules using NNGE. After that, they synthesized the rules by removing repetition rules by custom developed rule mining NNGE parser which removes repeated rules obtained. This parser is later pruned using the Ant-Miner algorithm. In [28], authors used the NNGE algorithm for the classification of the rice grains. The classification accuracy rate was high and NNGE was mixed with other image processing and machine learning approaches.

3. The Improved NNGE Algorithm

The improved NNGE algorithm uses our new VHDRA to provide a scalable and accurate classification solution that reduces the attack detection time and the computational

resources consumption. In our experiments, we evaluate the influence of the following three factors on the accuracy and computational performance of the NNGE. These factors are as follows.

(a) The reduction of the input features using a modified Particle Swarm Optimization (PSO) fitness function (Vertical Reduction): the PSO algorithm is used to compute the learning features weights and then ranking the learning features according to their computed weights. After that, the ones with the highest weight (i.e., which means that this feature enables the NNGE to accurately define the shortest Euclidean distance) are only selected to be used with the NNGE. To achieve this, we have developed a fitness function for the PSO algorithm to compute the learning feature weights of the weighted Euclidean distance of the NNGE between a new example and a set of exemplars. In this way, we efficiently integrate PSO with NNGE. We call this kind of reduction the vertical reduction of the input data.

(b) Pruning of nongeneralized exemplars using the highest ranked features of the PSO: the NNGE algorithm learns incrementally by first classifying, then generalizing each new example. When classifying an instance, one or more hyperrectangles may be found that the new instance is a member of wrong class. The algorithm prunes these so that the new example is no longer a member. Once classified, the new instance is generalized by merging it with the nearest exemplar of the same class, which may be a single instance or a hyperrectangle. Authors of [29] proved that generalizing exemplars result in improved classification performance over standard nearest neighbor. The only thing that may pose a problem is that the algorithm tends to produce rules that test a large number of input features that in turn hinder the scalability of the classification model. To this target, we use the reduced features produced by the PSO algorithm to solve this problem and furthermore we reduce the dataset input records using a STEM.

(c) Reducing the input dataset records using a STEM Algorithm (Horizontal Reduction): the STEM [5] is a new data processing and compression method to quantize and compress the heterogeneous datasets to reduce the size of data while preserving original key events and patterns within the datasets. STEM tracks system states from measurements and creates a compressed sequence of states for each observed scenario. It preprocesses the power system events to minimize the state space and number of rules generated by NNGE and results in high classification accuracy, short classification time, and small model building time.

The details of the previous three factors are described in the following three sections.

3.1. VHDRA Vertical Reduction Using a Modified PSO Fitness Function. The previous attempts of feature selection approaches that were tested with the NNGE algorithm such as CFS Expert Knowledge [6, 17], the Mutual Information based Feature Selection (MIFS) with the Joint Mutual Information (JMI) method [11], and the MISF with the Joint Mutual Information Maximisation (JMIM) method [11] have treated all features as equally important in computing the Euclidean distance to the nearest hyper rectangles and this makes them not accurate or suitable for the CPPS where each feature has a different weight. Furthermore, they give insignificant improvements in domains with relevant features such as the CPPS, where any of the features may influence the others. In this section, we introduce a new mechanism that ranks the input features based on their significance and considers the relevant features and their influence on the covering hyper-rectangle of the NGGE algorithm. A feature is considered more significant if its value is closer to a margin of the covering hyper-rectangle. The significant features enable the NNGE to accurately define the shortest Euclidean distance between a new example and a set of exemplars in memory to make a decision whether the new example belongs to a particular class. We implement our approach using the particle swarm optimization (PSO) algorithm that performs well in domains that have a large number of relevant and/or irrelevant features. PSO is one of stochastic optimization methods that are based on the swarming strategies in fish schooling and bird flocking [18]. It considers each solution to the problem in a D-dimensional space as a particle flying through the problem space with a certain position and velocity and finds the optimal solution in the complex search space through the interaction of particles in the population. The implementation of PSO requires few parameters to be adjusted and is able to escape from local optima. The velocity and position of the ith particle are denoted by the two vectors, respectively, $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$ and $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. Each particle moves in the search space according to its previous computed best particle position (*pbest)* and the location of the best particle in the entire population (*gbest)*. The velocity and position of the particles are updated using the following [18]:

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot rand1_t \cdot \left[pbest_i(t) - x_i(t)\right] + c_2 \cdot rand2_t \cdot \left[\text{gbest}(t) - x_i(t)\right] \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

where the velocity of the ith particle at iteration t *is given by* $v_i(t)$ and its position is given by $x_i(t)$ at the same iteration t, w is a weight factor to balance the global and local search function of particles, c_1 and c_2 are two learning factors which control the influence of the social and cognitive components and they are usually set to 2, *rand1* and *rand2* are two random numbers within the range of $[0, 1]$, $pbest_i(t)$ is the best previous position that corresponds to the best fitness value for ith particle at iteration t, and $gbest(t)$ is the global best particle by all particles at iteration t. The fitness value of the particle is evaluated after changing its position to $x_i(t+1)$. The *gbest and pbest* are updated according to the current position

```
Initialize population
While (number of generations, or the stopping criterion is not met)
  For i = 1 to number of particles
    If the fitness of X_i is greater than the fitness of pbest_i
    then Update pbest_i = X_i
    For k ∈ Neighborhood of X_i
      If the fitness of X_k is greater than that of gbest then
        Update gbest = X_k
    Next k
    For each dimension j
      v_ij(t + 1) = w × v_ij(t) + c_1 × rand_1 × (pbest_ij − x_ij(t))
       + c_2 × rand_2 × (gbest_j − x_ij(t))
      if v_ij(t + 1) ∉ (V_min, V_max) then
            v_ij(t + 1) = max(min(V_max, v_ij(t + 1)), V_min)
            x_ij(t + 1) = x_ij(t) + v_ij(t + 1)
    Next j
  Next i
Next generation until stopping criterion
```

Algorithm 1: The modified PSO Algorithm.

of the particles. The new particle velocity of each dimension $v_i(t+1)$ is tied to a maximum velocity V_{max} that is initialized by the user. As the PSO processes are repeated, all particles evolve toward the optimum solution. We give the pseudo code of the PSO algorithm at iteration t according to [18] with some modification as shown in Algorithm 1.

Our modification focuses on adapting the PSO to work with the NNGE algorithm by developing a new fitness function $x(t)$ as shown in (3), (4), and (5). The PSO fitness function defines the correct classification rate using the features picked by each particle. Figure 3 shows the flowchart of the PSO algorithm.

$$x_i(t) = \emptyset.\Omega_t(A_i) + \theta.(n - |(A_i)|) \quad (3)$$

$$\Omega_t(A_i) = \min_{f \in A_i}(\gamma_i(f)) \quad (4)$$

$$\gamma_i(f) = \min\left\{E_i(f) - H_i^{min}(f), H_i^{max}(f) - E_i(f)\right\} \quad (5)$$

where

(i) $x_i(t)$ is the fitness of particle i (one record of the dataset) at iteration t and it denotes how much a particle i features values are closer to a margin of the covering hyper-rectangle H which is going to be split through the NNGE classifier using the selected subset of features A of particle i. In other words, the main target of the fitness function is to choose the feature which ensures the most "balanced split" and in case there is a tie (two or more features have the same distance to a margin of H), the attribute leading to the largest number of training examples included in one of the splitting hyper-rectangles will be chosen.

(ii) A_i is the feature subset of particle i at iteration t; that is, $A = \{f1, f2, \ldots..f_{|A|}\}$.

(iii) $|(A_i)|$ is the length of the feature subset without the nonvalue features

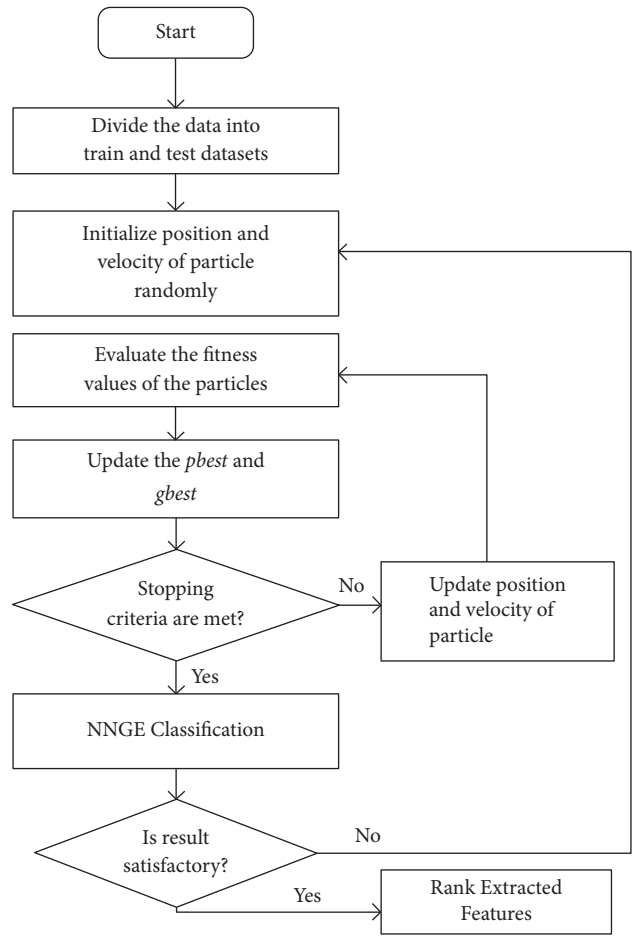


Figure 3: Feature ranking using PSO and the proposed fitness.

(iv) n is the total length of the feature subset including the nonvalue features.

(v) Ω_t is a measure of the classifier performance. It returns, for the whole subset of features A of particle i at iteration t, the shorts distance that any of these features can achieve to a margin of the covering hyper-rectangle H.

(vi) $\emptyset, \theta$ are two parameters that control the relative weight of classifier performance and feature subset length, $\emptyset \in [0, 1]$ and $\theta = 1 - \emptyset$. This formula denotes that the classifier performance and feature subset length have different effect on fitness function. In our experiments, we consider that classifier performance is more important than subset length because most of the power grid dataset records are of similar size and they have very few nonvalue features, so we set them to $\emptyset = 0.9, \theta = 0.1$.

(vii) γ_i denotes how much a certain feature f value *is* closer to a margin of the covering hyper-rectangle H.

(viii) E_i is the conflicting example of particle i; it represents an example record of dataset that needs to be classified.

(ix) H_i^{min}, H_i^{max} are the minimum and maximum margin values, respectively, of the covering hyper-rectangle H.

The iteration of the PSO will continue and stop when either one of the stopping criteria is met; (i) maximum number of iterations is defined to PSO or (ii) the fitness of the proposed feature subset has exceeded the maximum fitness value being set. We will use the fitness function given in (3) to compute the fitness of each particle in the dataset. For each feature f, the parameter γ_i will be computed; then at each point a stopping criterion is met; (ω) the minimum value of γ_i parameters corresponding to each feature f is selected. After that, all features are sorted according to their significance to the NNGE classification from the smallest to the largest one. Only few numbers of good features that exceed a particular threshold T that is computed during the training phase are selected. The threshold identified to select the most significant features is computed in the training phase for the Binary, Triple, and Multiclass datasets. To compute these three thresholds, the fitness of the PSO is defined by using the k-fold cross-validation, where k = 128, number of features in the dataset. The training dataset is divided into k subsets of the same size. One subset is used for validation. Using the remaining k–1 subsets, we build the new PSO fitness function based on (3), (4), and (5) that compute the nearest hyper rectangles of the NNGE. Each subset is used once for validation using one feature, and the process is repeated k times. We compute the threshold T using (6) by computing the average of the measure of fitness (γ_i), defined at (5), for all features from the 128 trials.

$$T = \frac{\left(\sum_{i=1}^{n} \gamma_i\right)}{n} \tag{6}$$

where n denotes number of validation data.

In the second phase, NNGE is used for classification using the top significant features that have fitness values ω lower than T. The classification step is based on the computation of the distance $D(E,H)$ between an example $E = (E1, E2, \ldots, En)$ and a hyper-rectangle H as given in (7) [7]. Figure 4 illustrates an example of splitting variants for numerical attributes.

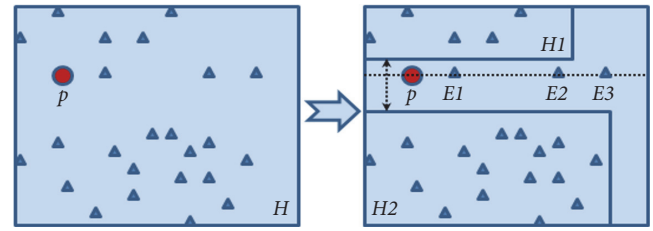


Figure 4: An example of NNGE splitting variants based on the extracted input features.

$$D(E, H) = \sqrt{\sum_{i=1}^{n} \left(w_i \frac{d(E_i, H_i)}{E_i^{max} - E_i^{min}} \right)} \tag{7}$$

where

(i) N is number of features in the current Example E.

(ii) E_j^{max}, E_j^{min} define the range of values over the training set which correspond to attribute i.

(iii) H_i is the interval $[H_i^{min}, H_i^{max}]$.

(iv) d is the distance between the features values and the corresponding hyper-rectangle "side" and it is computed according to

$$d(E_i, H_i) = \begin{cases} 0, & H_i^{min} \le E_i \le H_i^{max} \\ H_i^{min} - E_i, & E_i < H_i^{min} \\ E_i - H_i^{max}, & E_i > H_i^{max} \end{cases} \tag{8}$$

As shown in Figure 4, the left picture shows that the initial instances (filled triangle) belong to one hyper-rectangle H. After splitting, right picture shows that each instance is assigned to a particular hyper-rectangle *H1 or H2 according to* the closest margin strategies described before; some example instances *p1, E1, E2,* and *E3* represent a tie case described before; then, in such case, one of the features of these examples which leads to the largest number of training examples included in one of the splitting hyper-rectangles *H1* or *H2* will be chosen.

3.2. VHDRA Vertical Reduction Using the Evolutionary Pruning. There are two main approaches to reduce the size of classifiers: prepruning and postpruning. The prepruning approach aims to select the good training instances or prototypes and those are aiming at selecting the relevant attributes. This is achieved by VHDRA through the vertical data reduction using the PSO algorithm. The postpruning approach is applied to a set $H = \{H_1, H_2, \ldots, H_K\}$ of NNGE hyperrectangles once it has been generated with the aim of reducing its size and improving its classification accuracy.

In this paper, the selection of the hyperrectangles is based on the evolution of a population of binary encoded elements corresponding to various subsets of the initial set of hyperrectangles. In [19], two evolution pruning algorithms are introduced, the first version of the algorithm called EP-NNGE (Evolutionary Pruning in NNGE) and the EPA-NNGE algorithm. Authors of [19] proved that EPA-NNGE achieves high accurate classification results. In this paper, we use the EPA-NNGE to prune the hyperrectangles of the NNGE. EPA-NNGE is based on the idea of evolving a population of M binary strings containing K components. Each element x of the population corresponds to a subset of H; for example, if a component x_k has the value 1, it means that H_k is selected into the model, while if it is 0, it means that H_k is not selected. The quality of an element x is quantified using two measures: one is related to the accuracy of the classifier based on the selected hyperrectangles $H(x)$ and the other is related to the reduction of the model size. Thus the fitness is given by

$$f(x) = \lambda Acc(\mathscr{H}(x)) + (1-\lambda)\frac{|\mathscr{H}| - |\mathscr{H}(x)|}{|\mathscr{H}|} \quad (9)$$

where

(i) Acc denotes the accuracy that is computed by counting the correctly classified instances covered by the hyperrectangle that also means the total number of instances covered by the hyperrectangle after excluding the conflicting examples.

(ii) $|H|$ denotes the number of hyperrectangles.

(iii) $\lambda \in (0, 1)$ is a parameter controlling the compromise between the two quality measures.

The population elements of the EPA-NNGE algorithm are evaluated using (9). The computation of the classification accuracy of the EPA-NNGE algorithm is based on the computation of the distance between a test instance and a hyperrectangle and only the selected attributes (as are they specified by the corresponding part Xs of the population elements) are only considered. This means instead of using (7), we will use

$$D(E, H) = \sqrt{\sum_{i=1}^{n}\left(X_s w_i \frac{d(E_i, H_i)}{E_i^{max} - E_i^{min}}\right)} \quad (10)$$

3.3. VHDRA Horizontal Data Reduction Using STEM. The horizontal reduction approach uses the STEM. The STEM was used to preprocess power system data from a diverse set of sensors. Sensors include PMU, relay logs, control panel logs, and a network monitor called SNORT. The PMU provides mostly continuous data types and the other sensors provide discrete inputs. Section 4.3 gives a practical example of the STEM, and its detailed steps are discussed in [6] and are summarized in the following.

(1) Collect raw data: a PMU provides measurements of different electrical quantities from PMU, relay logs, control panel logs, and SNORT.

(2) Merge raw data: the previous measurements are upsampled prior to merging.

(3) Quantization: measurements with large state space should be quantized in such a way that the quantization interval maintains important patterns in the data.

(4) State mapping and compression: each row is mapped to a state ID. Unique states are provided an ID and repeated states use the same ID as previous instances of that state. After mapping, each row will have an assigned state ID. Compression removes repeated states in a sequence. After compression, each event is represented by a temporally ordered list of states and a label. After that, STEM sequentially captures only distinct states, compresses states, and stores it in the database. In this way the data size is significantly reduced by intelligently pruning the repetitive states.

4. Experimental Analysis and Results

We have conducted three types of experiments to evaluate the following.

(1) The effectiveness of the VHDRA vertical data reduction is by using the new fitness function of the PSO in selecting the significant features that their values are closer to a margin of the covering hyper-rectangle of the NNGE and the impact of this reduction on the classification accuracy of the NNGE algorithm

(2) The impact of the vertical data reduction is by using the EPA-NNGE pruning algorithm on the classification accuracy of the NNGE and on reducing the model size which in turns reduces the computational resources consumption. These experiments evaluate the reduction of the computational resources consumption in terms of the number of reduced hyperrectangles and ignored features that are defined by the pruning process in the training phase of the NNGE.

(3) There is the impact of the horizontal reduction of the STEM on the NNGE classification accuracy and the reduction of the model size in terms of the number of reduced hyperrectangles.

4.1. Evaluate the Impact of the VHDRA Vertical Reduction Using the New PSO Fitness Function. In these experiments, we evaluate the impact of the VHDRA vertical data reduction using the new PSO Euclidean distance fitness function on the NNGE classification accuracy and the model size. The experiments use the power grid dataset described in Section 1.2. In the training phase, we compute a particular threshold to extract the most significant features. The threshold values in the Binary, Triple, and Multiclass datasets, respectively, are 44.38, 61.23, and 81.78. Any feature with a fitness value larger than these thresholds is ignored. The algorithm defines the most significant 17 features for the Binary class dataset, 19 features for the Triple class dataset, and 22 features for the Multiclass datasets as shown in Table 1.

Table 1: The most significant selected features fitness values in the binary, triple, and multiclass datasets.

	Binary class dataset			Triple class dataset			Multiclass	
Order	Feature name	Fitness value	Order	Feature name	Fitness value	Order	Feature name	Fitness value
1	R2-CPA1	9.2	1	R1-VPA1	12.5	1	relay1_log	22.2
2	R3-CPA1	11.1	2	relay1_log	13.2	2	relay3_log	25.3
3	relay1_log	12.5	3	R1-CPM1	16.0	3	R4-VPA2	29.1
4	relay4_log	13.8	4	R2-VPA1	18.8	4	R4-VPM2	33.6
5	relay2_log	17.1	5	relay4_log	18.9	5	R4-VPA3	37.1
6	relay3_log	19.7	6	R2-CPM1	22.0	6	relay4_log	42.5
7	R1-VPA1	20.1	7	R3-VPA1	26.3	7	relay2_log	49.8
8	R4-CPA1	21.0	8	R3-CPA1	29.4	8	snort_log1	51.0
9	snort_log3	28.5	9	R3-CPM1	33.0	9	snort_log2	55.3
10	R1-CPA1	31.0	10	relay2_log	38.2	10	R4-VPM1	59.3
11	R2-VPA1	34.8	11	R4-VPM1	39.0	11	R2-CPA1	62.5
12	R3-VPA1	35.2	12	R4-CPA1	39.7	12	R3-CPA1	64.0
13	R4-CPM1	39.4	13	R4-CPM1	41.3	13	R2-VPA1	68.0
14	R4-Power	40.3	14	R3-Power	47.0	14	R3-VPA1	69.4
15	R3-Power	41.0	15	R2-Power	50.4	15	snort_log3	72.9
16	R3-CPM1	42.9	16	R1-CPA1	53.2	16	R4-CPM1	73.3
17	R2-Power	43.8	17	R4-VPA1	54.3	17	R3-Power	73.5
			18	R2-CPA1	57.0	18	R1-VPA1	75.4
			19	relay3_log	59.4	19	R2-Power	77.5
						20	R2-CPM1	79.5
						21	R4-Power	80.0
						22	R3-CPM1	81.0

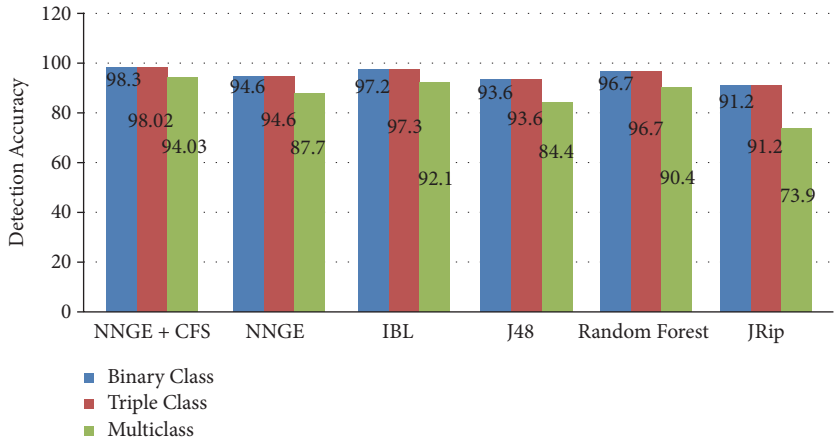


Figure 5: A comparison between the VHDRA using the PSO and the other existing approaches using the Binary, Triple, and Multi-class Datasets.

Table 2: NNGE classification rate using the Binary, Triple, and Multiclass datasets.

Feature selection	Binary class	Triple class	Multiclass
With (%)	98.38	98.01	94.03
Without (%)	65.42	66.41	23.66

Table 3: A comparison between the VHDRA using the PSO algorithm and the other existing approaches using the binary, triple, and multiclass datasets.

Approach	Binary class		Triple class		Multiclass	
	No. of features	Detection rate (%)	No. of features	Detection rate (%)	No. of features	Detection rate (%)
VHDRA (PSO)	*17*	*98.38*	*19*	*98.01*	*22*	*94.03*
NNGE (CFS)	28	94.68	28	94.62	28	87.77
IBL	28	97.20	17	97.38	28	92.10
J48	28	93.69	28	93.69	28	84.45
Random Forest	28	96.77	28	96.77	28	90.41
JRip	28	91.20	28	91.22	129	73.94

To test the accuracy of the selected features, we apply the NNGE classifier using (6) to compute the classification rates using the three datasets; see Table 2. In the following, we compare the output of our VHDRA with the new PSO fitness function against the most accurate five classification algorithms that we have tested before [11], namely, the traditional NNGE, Instance-based Learning (IBL), J48 tree, Random Forest, and JRip. According to our previous experiments [11], the best feature selection approach among the existing ones is the CFS. We used the CFS with the previous mentioned five classification algorithms using the Binary, Triple, and Multiclass datasets to compare the classification accuracy of these approaches against our approach. Table 3 and Figure 5 show that VHDRA with the new PSO fitness function uses less number of features and outperforms the classification accuracy of the current classification algorithms.

4.2. Evaluate the Impact of the VHDRA's Vertical Reduction Using the EPA-NNGE Pruning Algorithm. In these experiments, we evaluate the impact of the VHDRA vertical data reduction using the EPA-NNGE pruning algorithm on the classification accuracy and the model size. In these experiments, we use the significant features selected in the previous experiments of Section 4.1 using the modified PSO fitness function as follows: 17 features from the Binary dataset, 19 features from the Triple dataset, and 22 from the Multiclass datasets. Since our main goal of our approach is both to improve the classification accuracy and to reduce the model size, we edit the evolutionary process by a fitness function based on a value of λ that corresponds to an equilibrium point at which the EPA-NNGE accuracy rate and the hyperrectangles reduction rate are equal. The EPA-NNGE accuracy rate is computed for each dataset as a ratio between the numbers of correctly classified records/instances to the total number of records/instances in the dataset. The hyperrectangles reduction ratio is defined as $(|H| - |H(xbest)|)/|H|)$ where xbest is the instance with the corresponding best $f(x)$ value which is computed using (8). The influence of the parameter λ on the accuracy and on the reduction of the model size is evaluated for the Binary, Triple, and Multiclass datasets as shown in Figures 6, 7, and 8, respectively. The best values of λ that corresponds to the equilibrium point in the three datasets

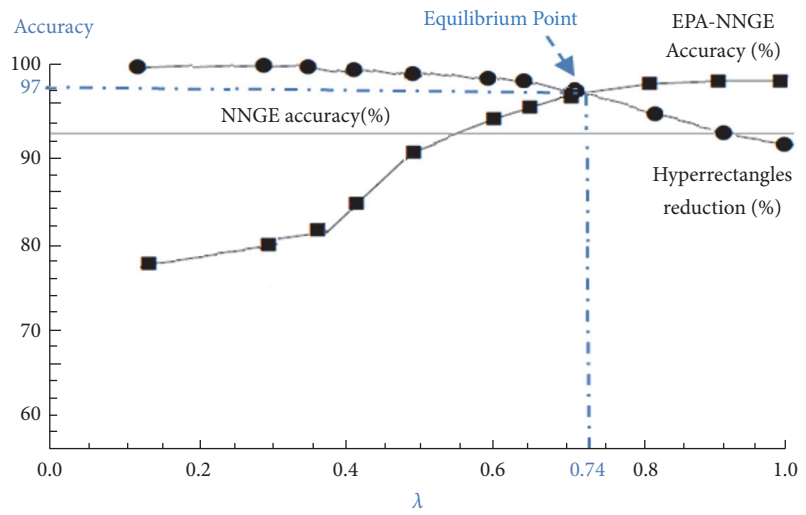


Figure 6: Influence of λ on the EPA-NNGE accuracy gain and hyperrectangles reduction using the Binary class dataset.

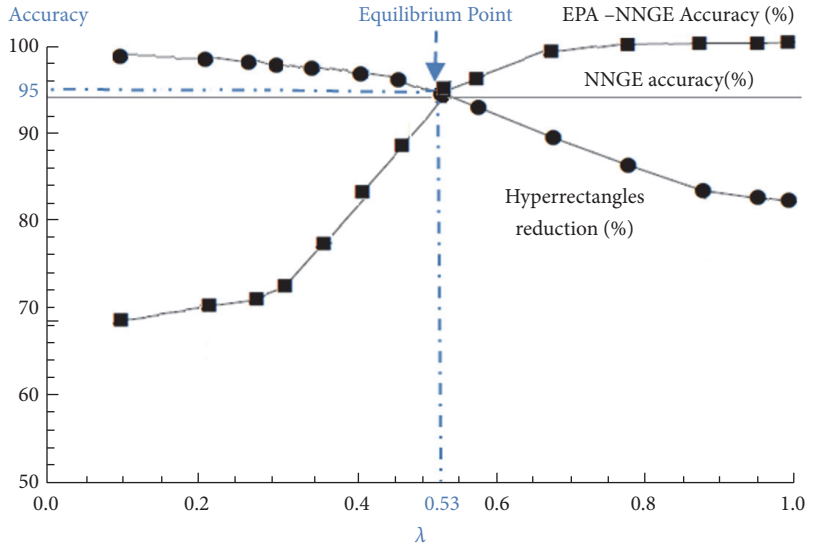


Figure 7: Influence of λ on the EPA-NNGE accuracy gain and hyperrectangles reduction using the Triple class dataset.

are 0.74, 0.53, and 0.44, respectively. To evaluate the impact of the pruning algorithm on the reduction of the model size and accuracy gain, we consider both the hyperrectangles reduction ratio described before and the reduced number of features as shown in Table 4.

An overall view of the accuracy gain ratio, hyperrectangles reduction ratio, and attributes reduction ratio for the Binary, Triple, and Multiclass Dataset is shown in Figure 9 for the VHDRA reduction using the EPA-NNGE and PSO versus the traditional NNGE with CFS without the pruning capabilities. The accuracy gain is computed as $(Acc(VHDRA) - Acc(NNGE))/Acc(NNGE) * 100)$. The ratio of the reduced hyperrectangles is computed as $|H_{VHDRA}|/|H_{NNGE}| * 100)$. The ratio of the reduced features is computed as $(N_{VHDRA}/N_{NNGE} * 100)$. Table 5 shows the accuracy gain, features reduction ratio, and hyperrectangles reduction ratio. The largest gain in accuracy (9.25%) was obtained using the Multiclass dataset. This can be explained by the fact that this dataset has the highest hyperrectangles reduction ratio (13.07%) and the lowest feature reduction ratio (35.71%). The smallest reduction in the model size (including the feature reduction and hyperrectangles reduction ratios) occurs using the Triple class dataset. This is why this dataset has the lowest accuracy gain (4.29%).

4.3. Evaluation of the Horizontal Data Reduction Using STEM. In this section, we evaluate the impact of the horizontal data reduction using STEM on the NNGE classification accuracy and the Model Size. The following STEM steps are applied.

(1) After collecting and merging raw data, we use the significant features selected in the previous experiments using the PSO and EPA approaches as follows: 15 features from the Binary dataset, 14 features from the

Table 4: A comparison between the VHDRA vertical reduction and the other existing approaches.

	Binary class *Size: 81,664 MB, 128 features*		Triple class *Size: 81,669 MB, 128 features*		Multiclass *Size: 78,963 MB, 128 features*	
Approach	No. of features	Detection rate (%)	No. of features	Detection rate (%)	No. of features	Detection rate (%)
VHDRA (PSO+ EPA)	*15* *(9.57MB)*	*99.21* *EPA Accu. gain:0.83%*	*14* *(8.93MB)*	*98.91* *EPA Accu. gain:0.90%*	*18* *(11.10MB)*	*97.03* *EPA Accu. gain: 3.0%*
VHDRA (PSO)	17 (10.85MB)	98.38	19 (12.12MB)	98.01	22 (13.57MB)	94.03
NNGE (CFS)	28 (17.86MB)	94.68	28 (17.91MB)	94.62	28 (17.77MB)	87.77
IBL	28 (17.97MB)	97.20	17 (10.87MB)	97.38	28 (17.62MB)	92.10
J48	28 (18.18MB)	93.69	28 (17.96MB)	93.69	28 (18.92MB)	84.45
Random Forest	28 (18.29MB)	96.77	28 (18.20MB)	96.77	28 (17.54MB)	90.41
JRip	28	91.20	28	91.22	129	73.94

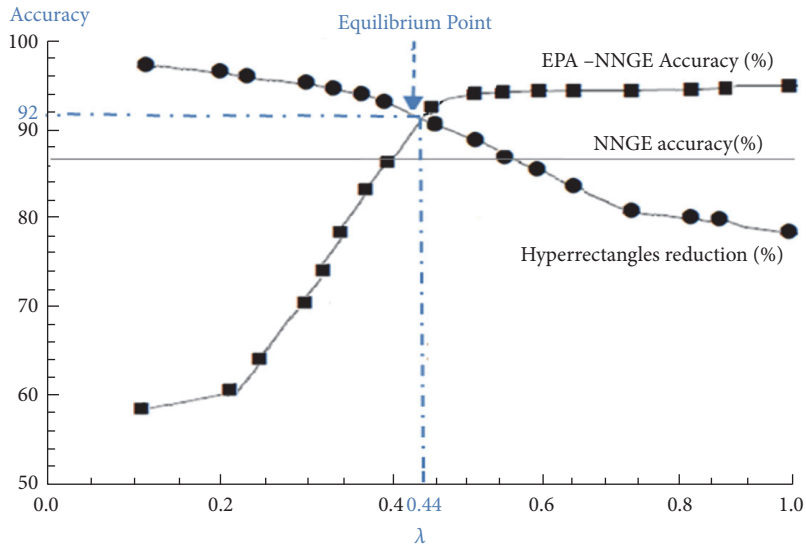


Figure 8: Influence of λ on the EPA-NNGE accuracy gain and hyperrectangles reduction using the Multi–class dataset.

Table 5: A comparison between the VHDRA and the traditional NNGE with CFS.

	Binary class	Triple class	Multiclass
Accuracy gain (%)	4.53	4.29	*9.25*
Feature reduction ratio (%)	46.43	*50*	35.71
Hyperrectangles reduction ratio (%)	9.68	7.41	*13.07*

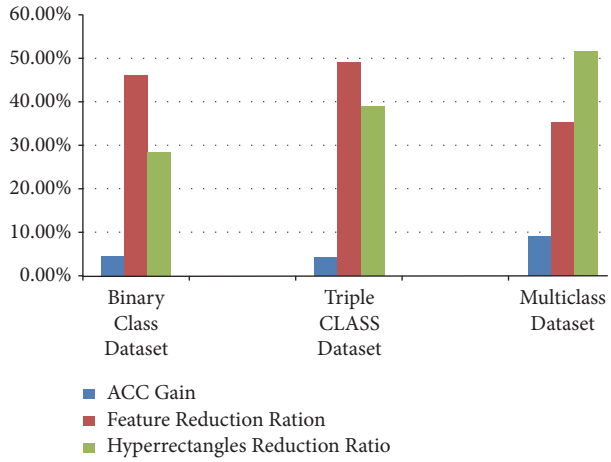


Figure 9: VHDRA with EPA-NNGE and PSO vs. NNGE-CFS: accuracy gain ratio, hyperrectangles reduction ratio, and attributes reduction ratio for the Binary, Triple, and Multi-class Dataset.

Triple dataset, and 18 from the Multiclass datasets; see Table 6. The numeric values of each feature are then quantized using domain expert inputs. The quantization intervals are shown in Table 6. To compare VHDRA to the NNGE with CFS, we repeated this step with the 28 features of the NNGE with CFS.

(2) Assigning states to each sample is as follows.

(a) Any row of the quantized data that contains the same quantization values for each feature is assigned the same state such as {S0, S1 . . . S14}

(b) The states of each data are inserted into the Quantized Datasets using the selected features.

(3) The duplicated values (i.e., the records in the dataset have the same state and marker values) are deleted. Only unique records are saved. Using this step, the size of the dataset was reduced from 5MB raw datasets to 6KB. This reduction can significantly enhance the detection speed.

The horizontal reduction improves the performance of the NNGE with CFS by 29.15%, 21.42%, and 17.81% using the Binary, Triple, and Multiclass datasets, respectively. The overall performance of VHDRA using both the vertical and the horizontal reduction improves the accuracy of the NNGE with CFS by 1.25%, 1.18%, and 5.23% using the Binary, Triple, and Multiclass datasets, respectively. STEM works also for online detection. The dataset is used to train the system and the quantization intervals are defined based on the dataset data boundaries. For online detection, the dataset is periodically updated and quantization intervals are modified. Furthermore, the states of each data are also updated and inserted into the Quantized Datasets and duplicated values are deleted.

Table 7 compares the detection accuracy gain ratio and the hyperrectangle reduction ratio of all VHDRA elements against the NNGE with CFS. From this table, we can notice

Table 6: Measurement quantization.

Feature	Quantization interval name	Quantization interval enumeration	Range
Voltage	Low, Normal, High	{0, 1, 2}	{(0, 130MV], (120MV, 146MV], (146MV, inf)}
Current	Low, Normal, Warming, High	{0, 1, 2, 3}	{(0, 100A], (100A, 700A], (700A, 12000A], (1200A, inf)}
Frequency	Low, Normal, High	{0, 1, 2}	{(0, 59.8Hz], (59.8Hz, 60.2Hz], (60.2Hz, inf)}
Impedance	Zone1, Zone2, Normal	{0, 1, 2}	{(0, 7.9], (7.9, 9.875], [9.875, inf)}
Control, relay, and SNORT log	Yes, No	{0, 1}	{0, 1}

Table 7: A comparison between VHDRA elements against NNGE with CFS.

	Binary class		Triple class		Multiclass	
	Accuracy gain (%)	Hyperrect. Reduction (%)	Accuracy gain (%)	Hyperrect. Reduction (%)	Accuracy gain (%)	Hyperrect. Reduction (%)
VHDRA (vertical and horizontal)	4.19	*37.34*	3.78	*26.76*	8.57	*29.06*
VHDRA (vertical)	*4.53*	9.68	*4.29*	7.41	*9.25*	13.07
VHDRA (horizontal)	1.25	29.15	1.18	21.42	5.23	17.81

that VHDRA with its vertical and horizontal reduction features achieves the best performance by reducing the NNGE hyperrectangles while keeping good detection accuracy.

5. Conclusion and Future Work

The large volume of high speed heterogeneous data of the CPPS makes their network targets for intrusions. In this paper, we introduced the VHDRA, a Vertical and Horizontal Data Reduction Approach, to improve the detection accuracy and speed and reduce the computational resource consumption of the NNGE algorithm. VHDRA reduces the NNGE's hyperrectangles by pruning the nongeneralized exemplars using the highest ranked features of the PSO.

It also reduces the size of dataset while preserving original key events and patterns within the datasets using the State Tracking and Extraction Method. The experiments show that the NNGE using VHDRA outperforms the current classification techniques for the Multi-, Binary, and Triple class datasets, respectively, as follows: the vertical reduction improves the accuracy of the NNGE with CFS by 9.25%, 4.53%, and 4.29% and reduces the features by 35.71%, 46.43%, and 50%. It also reduces the hyperrectangles by 13.07%, 9.68%, and 7.41%. On the other hand, the horizontal reduction improves the accuracy of the NNGE with CFS by 5.23%, 1.25%, and 1.18%.

The overall performance of VHDRA using both the vertical and the horizontal reduction reduces the hyperrectangles by 29.06%, 37.34%, and 26.76% and improves the accuracy of the NNGE with CFS by 8.57%, 4.19%, and 3.78%.

For future work, we will evaluate the scalability, actual computational resource consumption, and the speed of the VHDRA. Furthermore, we will also study the influence of quantizing and clustering the input data of the STEM using an intelligent model that integrates an Expert System with a Neural Network one instead of using domain expert input data on the accuracy and computational performance of our approach. Furthermore, to detect zero-day attacks, we will integrate our finite state Hidden Markov Model [30] to predict multistage and zero-day attacks in CPPS.

Data Availability

Reference [15] refers to the dataset used in this research. Furthermore, the tools used to implement the framework are given in the references.

Disclosure

The statements made herein are solely the responsibility of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was made possible by NPRP Grant # NPRP9-005-1-002 from the Qatar National Research Fund (a member of Qatar Foundation).

References

[1] H. Bao, R. Lu, B. Li, and R. Deng, "BLITHE: behavior rule-based insider threat detection for smart grid," *IEEE Internet of Things Journal*, vol. 3, no. 2, pp. 190–205, 2016.

[2] S. Pan, *Cybersecurity testing and intrusion detection for cyber-physical power systems [Ph.D. thesis]*, Mississippi State University, 2014.

[3] H. Wang, H. Sun, C. Li, S. Rahnamayan, and J. Pan, "Diversity enhanced particle swarm optimization with neighborhood search," *Information Sciences*, vol. 223, pp. 119–135, 2013.

[4] D. Zaharie, L. Perian, and V. Negru, "A view inside the classification with non-nested generalized exemplars," in *Proceedings of the IADIS European Conference on Data Mining*, 2011.

[5] U. Adhikari, *Event and intrusion detection systems for cyber-physical power systems [Ph.D. thesis]*, Mississippi State University, 2015.

[6] D. Zaharie, L. Perian, V. Negru, and F. Zamfirache, "Evolutionary pruning of non-nested generalized exemplars," in *Proceedings of the 6th IEEE International Symposium on Applied Computational Intelligence and Informatics, SACI 2011*, Timişoara, Romania, May 2011.

[7] U. Adhikari, M. Thomas, and P. Shengyi, "Applying non-nested generalized exemplars classification for cyber-power event and intrusion detection," *IEEE Transactions on Smart Grid*, vol. 9, pp. 3928–3941, 2018.

[8] T. B. Rasmussen, G. Yang, A. H. Nielsen, and Z. Dong, "A review of cyber-physical energy system security assessment," in *Proceedings of the IEEE Power and Energy Society PowerTech Conference*, pp. 1–6, Manchester, United Kingdom, June 2017.

[9] J. Valenzuela, J. Wang, and N. Bissinger, "Real-time intrusion detection in power system operations," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1052–1062, 2013.

[10] "National Infrastructure Protection Plan, DHS," 2018, https://www.dhs.gov/xlibrary/assets/NIPP_RiskMgmt.pdf.

[11] D. Harp and B. Gregory-Brown, *The State of Security in Control Systems Today*, SANS Institute, June 2015.

[12] Protecting Industrial Control Systems and SCADA Networks, Check Point Software Technologies Ltd., May 2015.

[13] *Protecting Industrial Control Systems, Recommendations for Europe and Member States*, ENISA, 2011.

[14] K. Grudziński, M. Grochowski, and W. Duch, "Pruning classification rules with reference vector selection methods," *ICAISC*, 2010.

[15] U. Adhikari, T. H. Morris, and S. Pan, "A cyber-physical power system test bed for intrusion detection systems," in *Proceedings of the 2014 IEEE Power & Energy Society General Meeting*, pp. 1–5, National Harbor, MD, USA, July 2014.

[16] H. A. Kholidy and F. Baiardi, "CIDD: A cloud intrusion detection dataset for cloud computing and masquerade attacks," in *Proceedings of the 9th International Conference on Information Technology: New Generations (ITNG)*, Las Vegas, Nev, USA, April 2012.

[17] Q. Chen, H. A. Kholidy, S. Abdelwahed, and J. Hamilton, "Towards realizing a distributed event and intrusion detection system," in *Proceedings of the International Conference on Future Network Systems and Security (FNSS 2017)*, Springer, Gainesville, Fla, USA, August 2017.

[18] M. Brent, *Instance Based Learning: Nearest Neighbor with Generalization*, University of Waikato, Hamilton, New Zealand, 1995.

[19] NimmyCleetus and K. A. Dhanya, "Rule induction using ant-miner algorithm," *International Journal of Scientific & Engineering Research*, vol. 5, no. 2, 2014.

[20] B. Martin, "Instance-Based Learning: Nearest Neighbour with Generalisation," Working Paper Series 95/18 Computer Science, University of Waikato, Hamilton, New Zealand, 1995.

[21] H. A. Kholidy, A. Erradi, S. Abdelwahed, and F. Baiardi, "A risk mitigation approach for autonomous cloud intrusion response system," *Computing: Archives for Scientific Computing*, vol. 98, no. 11, pp. 1111–1135, 2016.

[22] K. E. Martin, D. Hamai, M. G. Adamiak et al., "Exploring the IEEE standard C37.118-2005 synchrophasors for power systems," *IEEE Transactions on Power Delivery*, vol. 23, no. 4, pp. 1805–1811, 2008.

[23] V. K. Ojha, K. Jackowski, A. Abraham, and V. Snasel, "Dimensionality reduction, and function approximation of poly(lactic-co-glycolic acid) micro- and nanoparticle dissolution rate," *International Journal of Nanomedicine*, 2015.

[24] P. M. Baggenstoss, "Class-specific feature sets in classification," *IEEE Transactions on Signal Processing*, vol. 47, no. 12, pp. 3428–3432, 1999.

[25] B. Tang, H. He, P. M. Baggenstoss, and S. Kay, "A bayesian classification approach using class-specific features for text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 99, 2016.

[26] P. M. Baggenstoss, "The PDF projection theorem and the class-specific method," *IEEE Transactions on Signal Processing*, vol. 51, no. 3, pp. 672–685, 2003.

[27] B. Tang, S. Kay, and H. He, "Toward optimal feature selection in naive bayes for text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 9, pp. 2508–2521, 2016.

[28] O. Aki, A. Güllü, and E. Uçar, "Classification of rice grains using image processing and machine learning techniques," in *Proceedings of the International Scientific Conference*, Gabrovo, Bulgaria, 2015.

[29] H. A. Kholidy, A. Tekeoglu, S. Iannucci et al., "Attacks detection in SCADA systems using an improved non-nested generalized exemplars algorithm," in *Proceedings of the 12th IEEE International Conference on Computer Engineering and Systems (ICCES 2017)*, December 2017.

[30] H. A. Kholidy, A. Erradi, S. Abdelwahed, and A. Azab, "A finite state hidden markov model for predicting multistage attacks in cloud systems," in *Proceedings of the 12th IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC)*, Dalian, China, August 2014.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 5695021, 10 pages
https://doi.org/10.1155/2019/5695021

Research Article

Integrating Traffics with Network Device Logs for Anomaly Detection

Jiazhong Lu,[1] Fengmao Lv,[2] Zhongliu Zhuo,[1] Xiaosong Zhang,[1] Xiaolei Liu,[1] Teng Hu,[1] and Wei Deng[2]

[1]*Center for Cyber Security, University of Electronic Science and Technology of China, Chengdu 611731, China*
[2]*School of Statistics, Southwestern University of Finance and Economics, Chengdu 611130, China*

Correspondence should be addressed to Fengmao Lv; fengmaolv@126.com and Xiaosong Zhang; cdgbsjfzx@126.com

Received 25 December 2018; Revised 30 April 2019; Accepted 15 May 2019; Published 13 June 2019

Guest Editor: Pelin Angin

Advanced cyberattacks are often featured by multiple types, layers, and stages, with the goal of cheating the monitors. Existing anomaly detection systems usually search logs or traffics alone for evidence of attacks but ignore further analysis about attack processes. For instance, the traffic detection methods can only detect the attack flows roughly but fail to reconstruct the attack event process and reveal the current network node status. As a result, they cannot fully model the complex multistage attack. To address these problems, we present Traffic-Log Combined Detection (TLCD), which is a multistage intrusion analysis system. Inspired by multiplatform intrusion detection techniques, we integrate traffics with network device logs through association rules. TLCD correlates log data with traffic characteristics to reflect the attack process and construct a federated detection platform. Specifically, TLCD can discover the process steps of a cyberattack attack, reflect the current network status, and reveal the behaviors of normal users. Our experimental results over different cyberattacks demonstrate that TLCD works well with high accuracy and low false positive rate.

1. Introduction

Cyberattacks usually leave footprints on network devices. Typically, an attacker's attack path jumps through multiple routers or servers and then uploads malicious code (e.g., XSS script), implants virus (e.g., botnet), and submits Trojaned software or unofficial patch containing malicious payloads [1–7]. Generally, the footprints left by cyberattacks are spatiotemporally dispersed across logs of different victims' machines [8]. For instance, XSS script attack may leave evidence in server's weblog. However, as the logs are dispersed across diverse disconnected sources, piecing together the contextual information of each malicious footprint still needs human involvement. Therefore, directly leveraging the logs for anomaly detection is ineffective. On the other hand, network traffic can also provide complementary evidence for attack-related activities, such as anomalous data about connections from IRC/HTTP/DNS servers to botnet. However, it is insufficient to precisely detect attack behaviors and grasp a complete view of attacks with only the network traffic data.

To date, most existing log-based or traffic-based intrusion detection methods have the following limitations: (1) They only focus on a single or a few logs, lacking context information (especially the contacts in internal network). (2) The traffic characteristics are not diverse enough to achieve good detection performance. (3) Both the log-systems and the traffic-systems heavily rely on hefty equipment, which incurs very large cost overhead [9]. (4) The false positives and false negatives are not satisfactory in realistic detection process [10]. In this paper, we propose to integrate traffics with network device logs for detecting cyberattacks. Specifically, we collect logs and traffics from switch, router, firewall, and servers. Then we use fuzzy association rules to integrate the device logs with traffics to reconstruct the cyberattack.

Overall, the main contributions of this paper are listed as follows: (1) We propose a novel combined detection method to reconstruct the attack process. (2) Our method

can effectively integrate multiple network device logs with traffics by leveraging fuzzy association rules. (3) We conduct extensive evaluation of TLCD over diverse cyberattacks (e.g., phishing, XSS, and botnet). The experimental results clearly demonstrate the effectiveness of our method.

2. Related Work

In general, network intrusion detection mainly includes signature-based detection and anomaly-based detection [11, 12]. Specifically, signature-based detection relies on existing signature databases to detect malware infections. By using signature-based detection methods, malwares can be effectively identified through pattern matching. However, signature-based detection techniques have the fatal disadvantage that a new malware infection cannot be detected if its signature is not contained in the signature database.

Anomaly detection is a technique for detecting abnormal behaviors that deviate from normal behaviors [13, 14]. Specifically, it aims to detect events in the monitored domain different from the pattern defined by normal behaviors [13, 15]. Basically, the normal behavior of the network needs to be identified first. Compared to the signature-based detection, the main advantage of anomaly-based intrusion detection is the ability to detect new or unknown attacks, since abnormal behaviors can also occur when the signature of new malware is not available. However, due to the complexity of the behaviors from different networks and applications, it is difficult to accurately identify the normal behaviors. The existing anomaly detection methods are usually based on device logs or traffic flows alone [16]. In general, their methods are too simple to achieve satisfactory results [17]. Additionally, they fail to effectively reconstruct the attack conditions [18]. On the contrast, our detection method is multinetwork device interrelated and verified and can further improve the accuracy and reflect the state of the network environment at the time.

3. Method

This work mainly proposes to implement anomaly detection through integrating multiple network device logs with traffics. Specifically, the device logs and traffics are integrated through association rules. Our method can effectively improve the detection performance and reconstruct the network attack process, which enables us to grasp a complete view of the entire network environment.

3.1. Method Overview. Due to the diversity of network attacks, the network environment is quite complex. For instance, botnets must first send control commands to each C&C server and then to the controlled-host, while worm needs first to upload malware code to target-host and then infect others computers by the target-host. Therefore, the network device logs and traffic flows can play very important roles in cyberattack detection. To collect our data for anomaly detection, we first obtain the traffic flows with port mirroring and adopt TCPDUMP to extract useful traffic attributes (such as the source port, destination port, protocol number, source IP, destination IP, the number of packet size, and send time). Additionally, we also extract log information (such as Data, Time, Module, Level, PID, Type, Action, Application, Reason, etc.) from the gateway's internal routers, switches, firewalls, and servers. After that, we attempt to extract the mapping relations between logs and traffics with association rules. Finally, the extracted relationships can be used to generate the time stamps of log records and reconstruct the attack process.

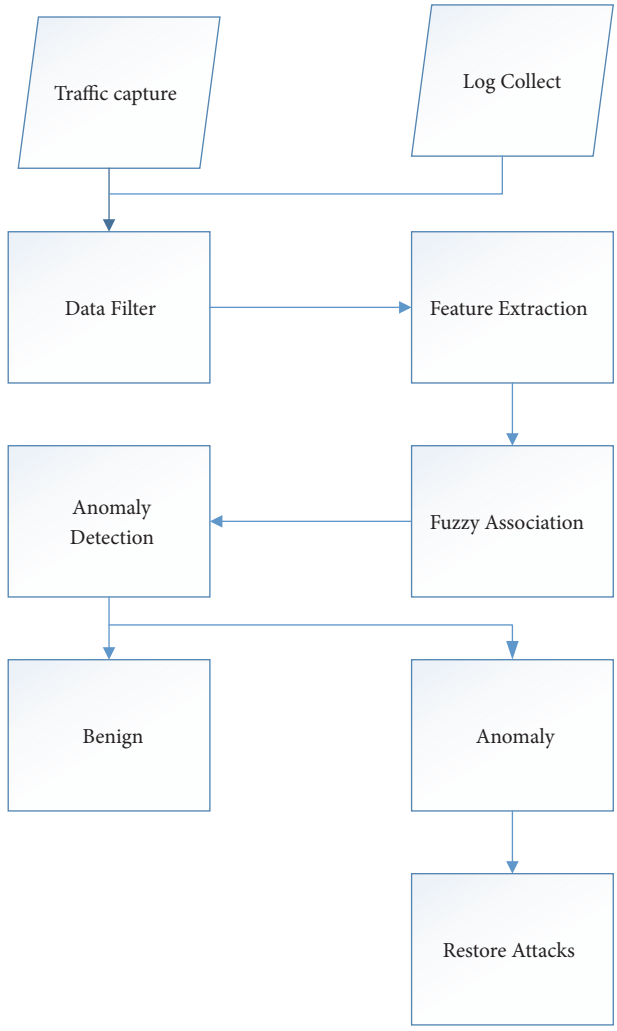


Figure 1: The overview of TLCD.

In Figure 1, we display the overview of TLCD. The locations of the traffic captures and log collectors are shown in Figure 2. Traffic capture modules are placed at the university servers and enterprise servers, which are connected outside the Internet. In this way, we can capture both the inbound and outbound data in real time. The inputs to TLCD are multiple raw data from network devices (e.g., router, switch, firewall, and servers). The detection is specifically designed to be format agnostic for both the traffics and logs. Through a parser plugin, TLCD can handle any input format of traffics and logs. As the detection task is usually featured by large-scale data, filtering is necessary for reducing the detection cost and time

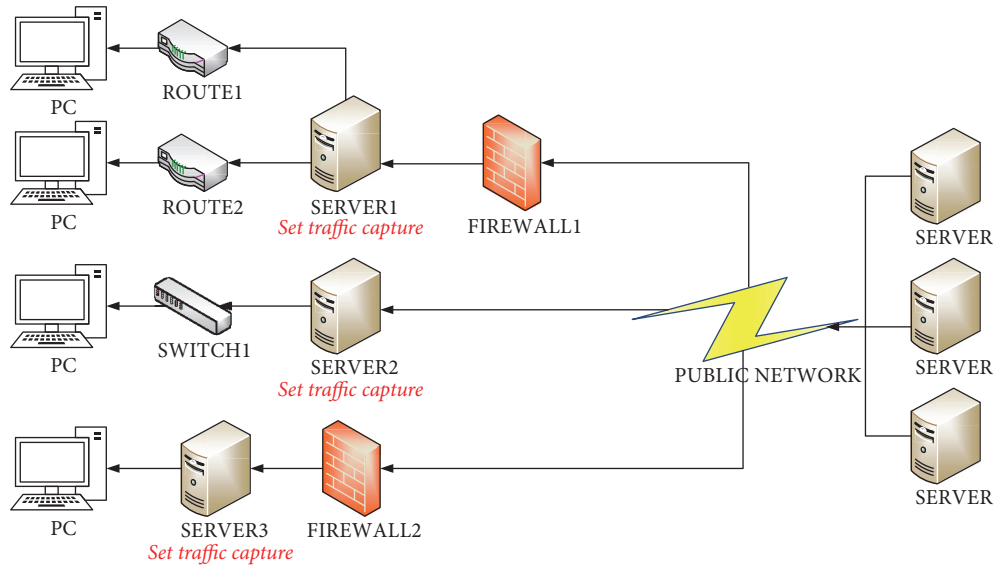


Figure 2: The deployment of the traffic captures and log collectors (server-1, firewall-1, router-1, and router-2 are deployed in enterprise. Servers-2, switch-1, firewall-1, and servers-3 are deployed in campus).

overhead. Therefore, we will filter out the irrelevant data in both logs and traffics. In the feature extraction module, we extract 63 traffic features (including 5 new TCP flags) and 16 log features. The log features are mainly used to reveal what happened before and after the cyberattack and auxiliarily detect the anomaly behaviors. The fuzzy association module aims at modeling strong associations between traffics and logs. After obtaining the confidence intervals of candidate rules, the ones among high confidence intervals are used to construct strong association rules. Note that there are many kinds of mappings here. It is possible that multiple traffics relate to one single log or one traffic relates to several logs.

To present the mechanisms of TLCD, we list the details as follows:

(1) Phase I, reprocessing: the traffic captures and log collectors first collect the original traffics and logs, which are considered as the inputs to TLCD. Then, these traffic-log inputs are reprocessed via the parser plugin and the filter module. Specifically, the filter module filters out the irrelevant data in both logs and traffics.

(2) Phase II, feature extraction: the feature extraction module consists of five components, including the traffic correlation (to analyze the traffic packets for malware), temporal correlation (to obtain the time characteristics of malware), combination correlation (to model the strong relevance of malware traffic), TCP flag (to record the sending and responding of traffic data), and log-information (to record the log information for attack reconstruction) components.

(3) Phase III, fuzzy association module: this module aims to integrate the traffics with logs through association rules.

Table 1: Details of the TCP flags.

TCP flags	TCP handshake situation
	ACK, URG, FIN, RST values
	The destination IP repeatedly responds with ACK = 1
	The destination IP only has ACK = 1, SYN = 1 and FIN=1
	The source IP only has SYN = 1

(4) Phase IV, anomaly detection module: advanced machine learning techniques are adopted to recognize malicious data as anomalies.

(5) Phase V, attack reconstruction module: the reconstruction module leverages the associations between the logs and traffics to generate the time stamps of log records, which can be finally used to reconstruct the attack process.

3.2. Feature Extraction

3.2.1. Network Traffics. To collect the traffic-log data, we have monitored the university-enterprise network for one month. The features used in our framework include temporal correlation features [19], TCP flag features (displayed in Table 1), and log features (displayed in Table 2). Usually, some cyberattacks, such as botnets and phishing emails, need to automatically send commands through programs. These automatic attack commands, more or less, contain inherent patterns. Specifically, we capture the traffics from 4 common cyberattacks (XSS, HTTP botnet, P2P botnet, and phishing) and find that different network attacks behave differently in the TCP handshake stage. For instance, phishing mail transmission process adopts POP3 and IMP4 protocol, allowing attackers to send different types of files. Also, the

Table 2: Detailsof the network device logs.

	Firewall logs	Traffic logs	Event logs	Network logs	Security logs	System logs	Cron logs	Mail logs	Messages logs	Mysqld logs
Data	*	*	*	*	*	*	*	*	*	*
Time	*	*	*	*	*	*	*	*	*	*
Module	*		*			*				
Level	*		*		*		*	*	*	*
PID			*			*	*	*	*	*
Type	*			*		*	*	*	*	*
Action		*					*	*	*	*
Source		*								
Destination		*								
Translated Source		*								
Translated Destination		*								
Duration		*								
Bytes Sent		*								
Bytes Received		*								
Application		*				*				
Reason	*	*	*	*	*	*	*	*	*	*

corresponding data volume can be very small. Therefore, the phishing mails can result in the same TCP handshake states to the normal ones, and it can be kinda easy to establish a connection. However, in a botnet, an attacker needs to control the C&C server and send a large number of commands, which will inevitably cause a handshake failure in TCP handshake. In Table 1, we display the details of TCP flags. Specifically, SYN denotes whether a connection is established, FIN and ACK denote the corresponding responses, and RST denotes the connection reset. Note that the ACK information can be used together with SYN and FIN as evidences for attack detection. For instance, if both SYN and ACK are activated, it means that the connection is established with confirmation. On the contrast, if only SYN is activated, we can conclude that that the connection is established without confirmation. Usually, most of the unreachable attacks can only activate SYN. Additionally, for the situation with FIN and RST activated and SYN unactivated, the firewalls may still detect the SYN/FIN packet. When such a packet appears in the situation, it is most likely that the network has been attacked. As the ACK/FIN packet represents a completed TCP connection, a normal FIN packet is always marked by ACK. A "NULL" packet is the packet not marked by any TCP flags (URG, ACK, PSH, RST, SYN, and FIN are all set to 0). For normal network activities, the TCP stack can never generate packets featured by unreasonable TCP flag combinations; otherwise the networks have been attacked. Therefore, the TCP flag features can provide useful information about the network status [19].

3.2.2. Network Device Logs. In Table 2, we display the details of network device logs. As we can see, different types of network device logs have different characteristics. Specifically, the firewall logs record the events between the inside and outside the network, such as port filtering, hazard level, and authentication; the traffic logs record current traffic conditions, such as packet size, IP address, and duration; the event logs record events that occur during the execution of the system, in order to provide traces for activity monitor and problem diagnosis; the network logs record the process of network access, such as data packet request or uploading; the security logs mainly record the operations of network devices and the system errors; the system logs record the hardware and software errors, as well as events that occur in the monitoring system, allowing the user to check the cause of errors and find traces left by the attacker; the Cron logs record periodic tasks in Linux (Cron reads the configuration files and writes them in memory when Linux starts. As there exist some cyberattacks featured by cyclicality, Cron logs are effective for identifying this type of attack); the mail logs allow the administrators to get the copies of messages processed by the Domino system router (when the mail log is enabled, Domino will check the messages as they go through MAIL.BOX, and save their copies to MAILJRN.NSF for future recovery); the message logs are plain text files that will be first checked for error messages when a problem occurs; the MySQL logs contain information of log-err, query log, log-slow-queries, log-update, and log-bin. By default, all logs are created in the MySQL directory. In this work, we extract attributes from ten types of logs from different network devices. Each type of log has its own attributes. For instance, the firewall log has attributes of data, time, module, level, type, and reason, while the traffic logs have attributes of data, time, action, source, destination, translated source, translated destination, duration, bytes sent,

bytes received, application, and reason. Although different logs reflect different characteristics of the device status, the shared attributes, such as time, date, and reason, can be effectively used to infer the status of one event in different logs.

3.3. Anomaly Detection

3.3.1. Feature Integration through Association Rules. In daily networks, there are no direct correlations between the log data and the traffic data. However, they can be correlated through the shared attributes like time and date. Therefore, we need to model the mutual mappings between traffics and logs by effectively leveraging the correlated attributes. With that, we can obtain the classification boundaries of the log attributes based on the corresponding attributes of traffics.

The discretization of traffic features, which is useful for boundary division, plays an important role in detecting anomaly traffics. Specifically, we use the Fuzzy-C Means (FCM) algorithm to divide the traffic characteristics (including quantitative attributes and Boolean attributes) into several fuzzy sets. Note that the elements and nonelements of each fuzzy set can be mutually transformed, in order to achieve the goal of softening features. In the process of highly skewed data, FCM algorithm can effectively model the actual distribution of data and clearly reveal the boundary between normal data and anomalies.

In our method, we first extract 29 basic attributes of the traffics, including the five-tuple (source IP address, destination IP address, source port, destination port, and protocol number), the total number of uplink and downlink packets, the total number of uplink and downlink payload packets, the total amount of uplink and downlink load, flow duration, average load, the maximum load, the minimum load, average time interval between the uplink and downlink data packets, the minimum time interval, the maximum time interval, and so on. Then, we extract 16 basic attributes of the logs, including data, time, module, PID, type, action, source, destination, translated source and destination, duration, bytes sent and received, application, reason, and so on. We assume that each feature comes from a Gaussian distribution. Then according to the membership function of fuzzy recognition, we can determine the fuzzy numbers of the maximum fuzzy set. Denoting the center of the maximum fuzzy set as μ, the membership degree as r_i (i = 1, 2, 3, ..., n), and σ as the parameter, the Gaussian fuzzy expression can be represented as follows:

$$\mathrm{y} = \exp\left[-\frac{(x-\mu)^2}{\sigma^2}\right]. \tag{1}$$

To approximate the maximum membership degree, we design the objective function as

$$g(\sigma) = \sum_{i=1}^{n}\left\{\exp\left[\frac{-(x_i-\mu)^2}{\sigma^2}\right] - r_i\right\}^2. \tag{2}$$

The corresponding membership function is expressed as

$$A_{ij}\left(x_j\right) = \begin{cases} 0, & \left|x_j - \overline{x}_j\right| > 2s_j \\ 1 - \left(\dfrac{x_j - \overline{x}_j}{2s_j}\right)^2, & \left|x_j - \overline{x}_j\right| \le 2s_j, \end{cases} \tag{3}$$

where $\overline{x_j}$ is the center and $2s_j$ is the standard deviation σ. Finally, we can identify whether a sample is an anomaly based on the principle of the maximum membership.

3.3.2. Anomaly Detection. The key point in anomaly detection is to detect anomalies from benign data according to the extracted features. To achieve this, we adopt supervised learning methods, such as K-Nearest Neighbor (KNN), Support Vector Machine (SVM), neural networks, or decision trees, to design the detection module. Basically, supervised learning first needs to establish a training set and then train a classification model over the training set. For this anomaly detection task, our goal is to learn a classifier that can effectively detect out the anomalies. In this work, we adopt Gradient Boosting Decision Tree (GBDT), which is an advanced machine learning technique and models the data with an ensemble of decision trees. To finally evaluate the performance of our method, 10-fold cross validation is adopted in our work.

4. Experiments

4.1. Dataset. In our experiments, we evaluate our method over 4 types of network attack (XSS, HTTP botnet, P2P botnet, and phishing). These cyberattacks are carefully injected into the normal business and will not bring undesirable effects for other business. Both the traffic data and the log data are collected from university servers and enterprise servers. To obtain the traffic data, we have monitored the university-enterprise network for one month. Specifically, we simulate the P2P botnet and HTTP botnet attack according to the Contagio blog [21] and white paper [22], which provide guidance about how to make botnet evade intrusion detection techniques. To simulate the XSS attacks, we inject malicious code into the web pages of university servers. The simulated phishing emails are sent to both university servers and enterprise servers. Note that in our simulation, the anomaly traffics only account for 0.1% of the total traffic flows, which is close to the real situations. As displayed in Table 3, the collected data include 30 normal traffic datasets, 6 traffic datasets for XSS injection attacks, 5 traffic datasets for phishing emails, and 20 ones for botnets (13 P2P botnets and 7 HTTP botnets). On the other hand, as displayed in Table 4, the log data are collected from 1 switch, 2 routers, 2 firewalls, and 3 servers.

4.2. Experimental Results. To validate the performance of integrating traffics with logs for anomaly detection, we conduct comparison experiments through only leveraging the traffic data (or the log data). As displayed in Tables 5–8 and Figures 3–6, it is clear that neither traffics nor logs can independently achieve desirable results in detecting

Table 3: The collection details for traffic data.

Type	Traffic Amount	Name
Normal	30	N/A
XSS	6	N/A
Phishing	5	N/A
HTTP botnets[20]	7	Virut, Sogou
P2P botnets [21]	13	NSIS.ay, SMTP Spam, Zeus (C&C), UDP Storm, Zeus, Zero access, Weasel

Table 4: The collection details for log data.

Device name	Quantity	Brand
Switch	1	Huawei
Router	2	Cisco,Huawei
Firewall	2	Juniper
Server	3	Cisco

Table 5: The detection results over XSS attack.

XSS	FP	FN
10-fold KNN for traffics	8.2%	5.6%
10-fold SVM for traffics	8.6%	5.8%
10-fold KNN for logs	9.1%	9.9%
10-fold SVM for logs	9.0%	8.6%
10-fold SVM for logs-and-traffics	5.2%	6.3%
10-fold KNN for logs-and-traffics	6.2%	3.6%
TLCD (GBDT)	4.3%	2.5%

Table 6: The detection results over phishing email.

Phishing	FP	FN
10-fold KNN for traffics	7.1%	7.3%
10-fold SVM for traffics	6.5%	7.3%
10-fold KNN for logs	8.8%	8.3%
10-fold SVM for logs	7.9%	8.2%
10-fold SVM for logs-and-traffics	5.0%	6.0%
10-fold KNN for logs-and-traffics	5.5%	4.8%
TLCD (GBDT)	5.3%	4.9%

Table 7: The detection results over HTTP botnet.

Http Botnet	FP	FN
10-fold KNN for traffics	5.5%	4.8%
10-fold SVM for traffics	5.3%	5.0%
10-fold KNN for logs	6.3%	5.9%
10-fold SVM for logs	6.3%	5.8%
10-fold SVM for logs-and-traffics	3.6%	2.9%
10-fold KNN for logs-and-traffics	3.8%	2.7%
TLCD (GBDT)	2.5%	2.8%

cyberattacks (both the False Negative (FN) and False Positive (FP) values decrease significantly), which is consistent with [16]. On the contrast, when we integrate the traffic flows with network device logs, the detection performance can be significantly improved. Additionally, we also compare

Table 8: The detection results over P2P botnet.

P2P botnet	FP	FN
10-fold KNN for traffics	4.5%	4.6%
10-fold SVM for traffics	5.2%	5.0%
10-fold KNN for logs	6.4%	6.0%
10-fold SVM for logs	6.0%	5.9%
10-fold SVM for logs-and-traffics	2.9%	2.9%
10-fold KNN for logs-and-traffics	3.3%	2.9%
TLCD (GBDT)	2.8%	2.6%

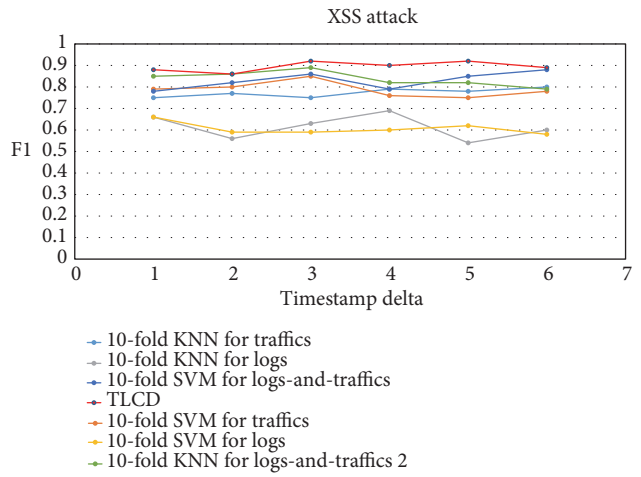


Figure 3: The F1 value obtained by each method over the XSS attack.

the detection performance of different supervised learning methods, including SVM, KNN, and GBDT. As we can see, these compared methods can achieve very similar results, with GDBT slightly better than the others. This effectively demonstrates that the features obtained through integrating traffics with logs are robust for our cyberdetection task.

4.3. Attack Reconstructions. In our experiments, we also evaluate the performance of TLCD on attack reconstruction. In particular, for these detected attacks, we first obtain their time horizon and communication address according to the information of the corresponding anomaly traffics, such as data, time, IP, and so on. With that, we can get the corresponding log features, and then the concrete network devices are determined. Finally, we reconstruct the original attack paths based on the abnormal information above.

Figures 7–10 display our attack reconstruction results for the four simulated cyberattacks. Generally, the XSS attack

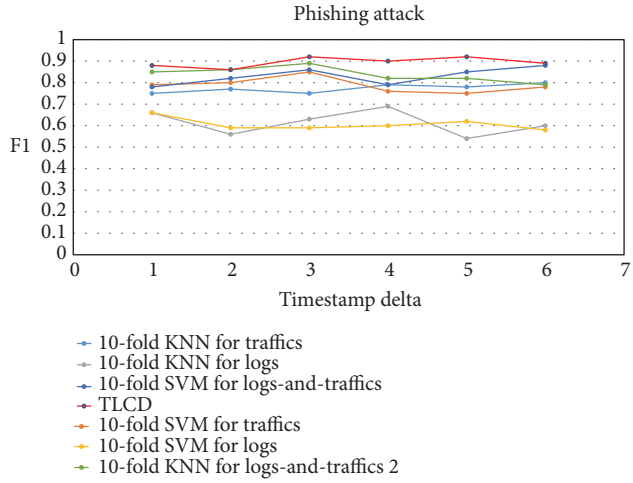


Figure 4: The F1 value obtained by each method over the phishing email.

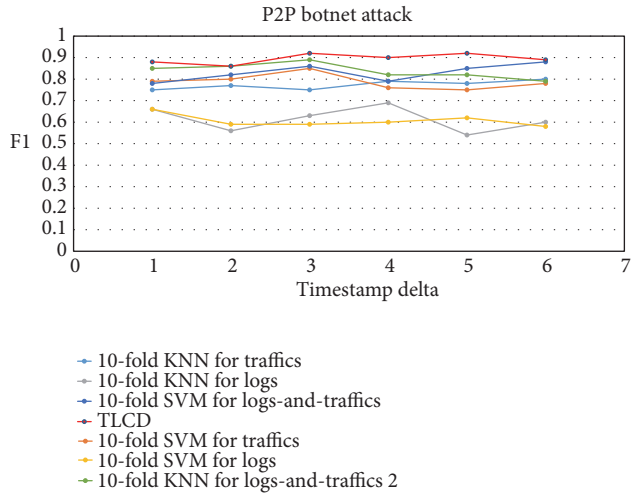


Figure 5: The F1 value obtained by each method over the P2P botnet.

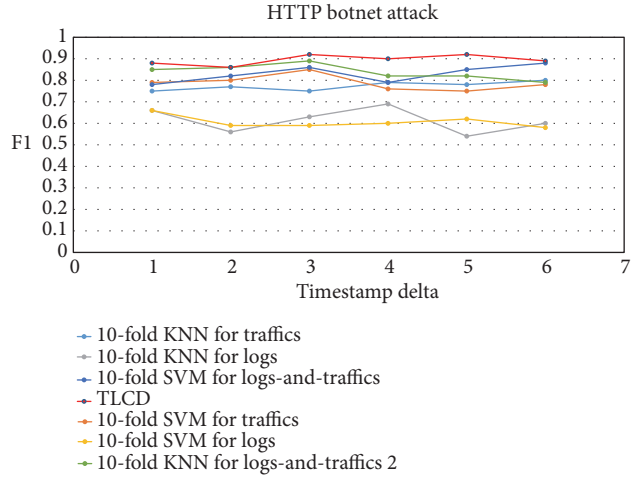


Figure 6: The F1 value obtained by each method over the HTTP botnet.

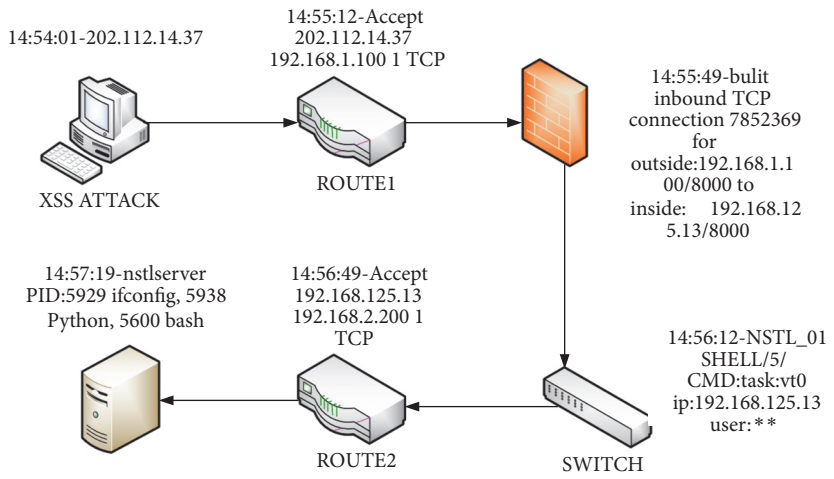


Figure 7: XSS attack reconstruction.

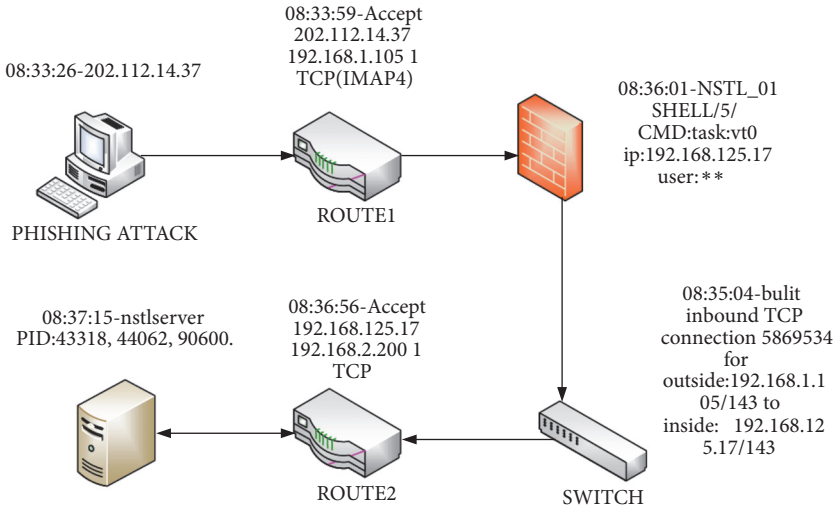


Figure 8: Phishing attack reconstruction.

damages the web server through passing several network devices, including routers, firewalls, switches, and so on. The phishing attack shares similar attack process to XSS, except that it adopts the IMP4 protocol and a fixed port. Different from XSS and phishing, the HTTP botnet does not aim to attack the servers, but the hosts through passing the servers. Note that the HTTP botnet attack is completed by web pages and the ICMP protocol. The P2P botnet attack is implemented not only through the servers, but also directly over the hosts. As displayed in Figures 7–10, our reconstructed results have accurately revealed the attack paths of the corresponding cyberattacks, which demonstrates that our method can effectively reconstruct the attack process.

5. Conclusion

In this paper, we propose to integrate traffics with network device logs for detecting cyberattacks. Specifically, we use fuzzy association rules to integrate the device logs with traffics to obtain the features for attack detection and reconstruction. The experiments over four common network attacks clearly demonstrate that our TLCD method can effectively detect diverse cyberattacks and reconstruct their event process.

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 paper is supported by National Natural Science Foundation of China under grant no. 6157211.

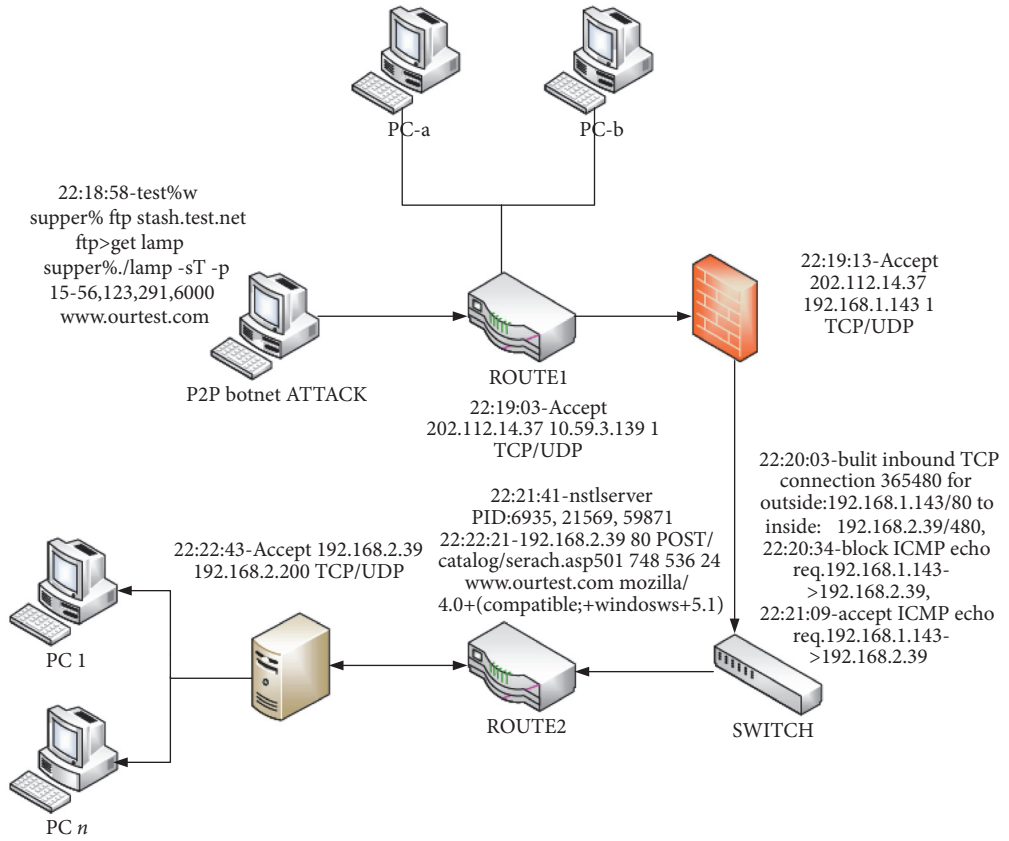


Figure 9: HTTP botnet attack reconstruction.

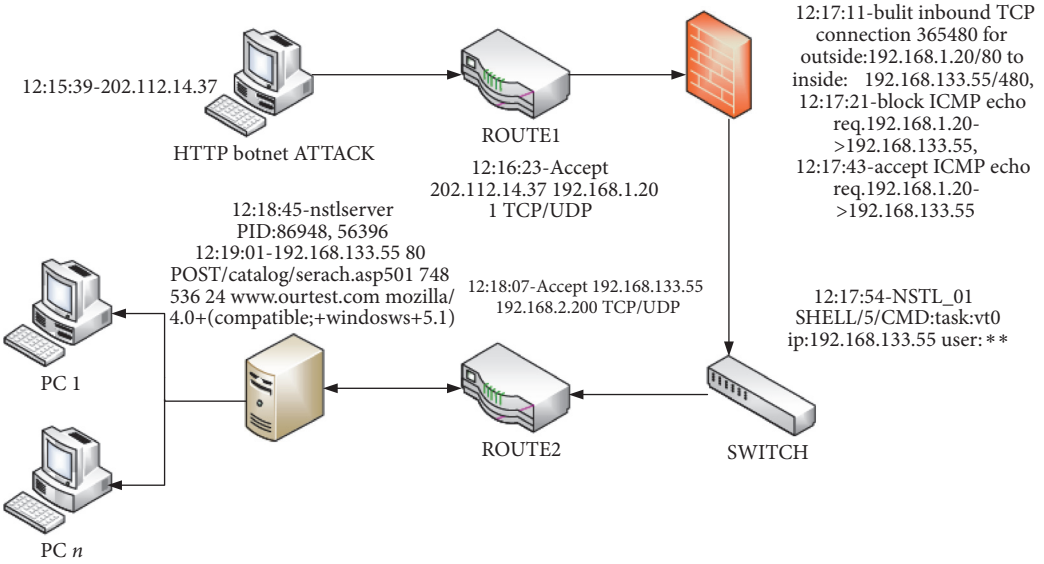


Figure 10: P2P botnet attack reconstruction.

References

[1] A. Sood and R. Enbody, *Targeted Cyber Attacks: Multi-Staged Attacks Driven by Exploits and Malware*, Syngress, 2014.

[2] K. L. Chiew, K. S. C. Yong, and C. L. Tan, "A survey of phishing attacks: Their types, vectors and technical approaches," *Expert Systems with Applications*, vol. 106, pp. 1–20, 2018.

[3] B. Caswell, J. Beale, and A. Baker, *Snort Intrusion Detection and Prevention Toolkit*, Syngress, 2007.

[4] A. Yaar, A. Perrig, and D. Song, "Pi: a path identification mechanism to defend against DDoS attacks," in *Proceedings of the 2003 Symposium on Security and Privacy, SP 2003*, pp. 93–107, USA, May 2003.

[5] H. Wenhua and Y. Geng, "Identification method of attack path based on immune intrusion detection," *Journal of Networks*, vol. 9, no. 4, pp. 964–971, 2014.

[6] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu, "Portcullis: protecting connection setup from denial-of-capability attacks," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 289–300, 2007.

[7] K. Kleemola, M. Crete-Nishihata, and J. Scott-Railton, Targeted Attacks against Tibetan and Hong Kong Groups Exploiting CVE-2014-4114, Citizen Lab, June 2015.

[8] J. R. Crandall, G. Wassermann, D. A. de Oliveira, Z. Su, S. F. Wu, and F. T. Chong, "Temporal search: detecting hidden malware timebombs with virtual machines," *ACM SIGARCH Computer Architecture News*, vol. 34, no. 5, pp. 25–36, 2006.

[9] Z. Gu, K. Pei, Q. Wang, L. Si, X. Zhang, and D. Xu, "LEAPS: detecting camouflaged attacks with statistical learning guided by program analysis," in *Proceedings of the 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 57–68, Rio de Janeiro, Brazil, June 2015.

[10] K. Pei, Z. Gu, B. Saltaformaggio et al., "Hercule: attack story reconstruction via community discovery on correlated log graph," in *Proceedings of the the 32nd Annual Conference on Computer Security Applications*, pp. 583–595, ACM, Los Angeles, Calif, USA, December 2016.

[11] H.-S. Chen, W. Gao, and D. G. Daut, "Signature based spectrum sensing algorithms for IEEE 802.22 WRAN," in *Proceedings of the 2007 IEEE International Conference on Communications, ICC'07*, pp. 6487–6492, IEEE, UK, June 2007.

[12] J. Zhang and M. Zulkernine, "Anomaly based network intrusion detection with unsupervised outlier detection," in *Proceedings of the 2006 IEEE International Conference on Communications, ICC 2006*, pp. 2388–2393, IEEE, Turkey, July 2006.

[13] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández et al., "Anomaly-based network intrusion detection: techniques, systems and challenges," *Journal of Computers and Security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[14] F. Gong, "Deciphering detection techniques: Part ii anomaly-based intrusion detection," White Paper, McAfee Security, 2003.

[15] Y. Yu, "A survey of anomaly intrusion detection techniques," *Journal of Computing Sciences in Colleges*, vol. 28, no. 1, pp. 9–17, 2012.

[16] F. Sönmez, M. Zontul, O. Kaynar, and H. Tutar, "Anomaly detection using data mining methods in it systems: a decision support application," *Sakarya University Journal of Science*, vol. 22, no. 4, pp. 1109–1123, 2018.

[17] N. Kamiyama and T. Mori, "Simple and accurate identification of high-rate flows by packet sampling," in *Proceedings of the Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pp. 1–13, IEEE, Barcelona, Spain, April 2006.

[18] G. Carl, G. Kesidis, R. R. Brooks, and S. Rai, "Denial-of-service attack-detection techniques," *IEEE Internet Computing*, vol. 10, no. 1, pp. 82–89, 2006.

[19] J. Lu, K. Chen, Z. Zhuo, and X. Zhang, "A temporal correlation and traffic analysis approach for APT attacks detection," *Cluster Computing*, pp. 1–12, 2017.

[20] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Journal of Computers and Security*, vol. 45, pp. 100–123, 2014.

[21] M. Parkour, Contagio malware database, https://www.mediafire.com/folder/c2az029ch6cke/TRAFFIC_PATTE%20RNS_COLLECTION, 2013.

[22] S. Siddiqui, M. S. Khan, K. Ferens, and W. Kinsner, "Detecting advanced persistent threats using fractal dimension based machine learning classification," in *Proceedings of the 2nd ACM International Workshop on Security and Privacy Analytics (IWSPA '16)*, pp. 64–69, ACM, 2016.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 8085303, 15 pages
https://doi.org/10.1155/2019/8085303

Research Article

RMMDI: A Novel Framework for Role Mining Based on the Multi-Domain Information

Wei Bai, Zhisong Pan, Shize Guo, and Zhe Chen

Command & Control Engineering College, Army Engineering University of PLA, Nanjing 210014, China

Correspondence should be addressed to Zhisong Pan; hotpzs@hotmail.com

Received 20 November 2018; Revised 14 March 2019; Accepted 24 April 2019; Published 11 June 2019

Guest Editor: Rohit Ranchal

Role-based access control (RBAC) is widely adopted in network security management, and role mining technology has been extensively used to automatically generate user roles from datasets in a bottom-up way. However, almost all role mining methods discover the user roles from existing user-permission assignments, which neglect the dependency relationships between user permissions. To extend the ability of role mining technology, this paper proposes a novel role mining framework based on multi-domain information. The framework estimates the similarity between different permissions based on the fundamental information in the physical, network, and digital domains and attaches interdependent permissions to the same role. Three simulated network scenarios with different multi-domain configurations are used to validate the effectiveness of our method. The experimental results show that the method can not only capture the interdependent relationships between permissions, but also detect user roles and permissions more reasonably.

1. Introduction

Access control is a fundamental concern in network security management. Role-based access control (RBAC) has become the dominant model for both commercial and research fields [1, 2]. The key point of RBAC is to determine proper roles to capture business needs, which is named as role engineering. There are mainly two kinds of approaches to find user roles: top-down and bottom-up. The top-down approaches always perform a deep analysis of business processes and identify user roles manually [3], while the bottom-up approaches always discover the user roles from existing datasets automatically, which are also named as role mining as they usually resort to data mining techniques [4, 5].

Existing role mining approaches mainly discover a proper user-role assignment relation $UA \subseteq USERS \times ROLES$ and a proper role-permission assignment relation $PA \subseteq ROLES \times PERMS$ from an existing user-permission assignment relation $UPA \subseteq USERS \times PERMS$. In the process, user-permission assignments are considered to be independent. However, considering a typical service authorization process, users are authorized by multiple policy control points, including gate machines, firewalls, or identity authentication systems. Those systems are always configured separately and may grant users with more permissions than they deserve. For example, users, who are authorized to enter certain space, may have the opportunity to use the terminals belong to other users in the same space; users can connect the server behind the firewall remotely to bypass the access control lists and access unauthorized services; users can use the assigned passwords to crack similar passwords for other unauthorized services, etc. In a word, if the interdependent relationships are not taken into consideration, users with certain roles would get extra permissions, introducing security vulnerabilities into network systems.

To address the above-mentioned issues, this paper proposes a novel role mining framework named as RMMDI from the perspective of network security management. Instead of mining user roles from user-permission assignments, the framework discovers user roles from the fundamental information in multiple domains, including the physical domain, network domain, and digital domain. The framework is aimed at outputting a flat RBAC state that divides user permissions into several disjoint subsets. The user permissions in one set tend to be interdependent while the permissions in different sets tend to be independent. If a permission

set is assigned to a user role, a user assigned some roles is unlikely to get extra permissions assigned to other roles. As such, potential security risks involved in the user-permission assignments process can be avoided.

The rest of this paper is organized as follows. In Section 2, some general works are briefly reviewed. Section 3 presents the proposed framework in detail. Section 4 shows the experimental setup and results, and Section 5 presents a comprehensive discussion. At last, Section 6 provides concluding remarks.

2. Related Work

2.1. Role Mining. RBAC has become a dominating model for access control in network security. Instead of assigning permissions to the user directly, RBAC introduces the concept of roles to make access control system more compact and comprehensive [6]. A role is defined as a collection of permissions. The key point of RBAC is to generate proper roles. In this process, the bottom-up approach named as role mining gets much more attention than the top-down approach as the latter is time-consuming and human-intensive [3].

Kuhlmann et al. first proposed the concept of role mining for finding roles from user-permission assignment data [7]. Traditional role mining approaches are mainly divided into two classes based on their output [5, 8, 9]. The first class is to output a prioritized list of candidate roles, each of which is assigned a priority value. A larger priority value means the role is more important or useful. Complete Miner (CM) and Fast Miner (FM) are two typical algorithms of the first class, which identify overlapping clusters by analyzing the subset enumeration in an unsupervised way [10]. The second class is to output a complete RBAC state under a certain cost. There are also a lot of classic algorithms in the class, for example, OFFIS Role mining tool with Cluster Analysis (ORCA) [11], Hierarchical Miner (HM) [12], Graph Optimization (GO) [13], HP Role Minimization (HPr) [14], and HP Edge Minimization (HPe) [14].

Besides those traditional role mining algorithms, there are also many important approaches that emerged in recent years. For example, Frank et al. proposed a probabilistic approach to improve the role mining process by taking account of the business information. The approach utilized the similarity between user-permission relations to detect exceptional assignments and wrong assignments [15]. Besides, entropy-based methods were used in this approach to analyze the impact of business knowledge on role mining [16]. Alessandro et al. presented an approach that allowed role engineers to leverage business information. In the role mining process, the access data was divided into smaller subsets from a business perspective firstly and then traditional methods can be used to discover roles with business meanings [5]. Iran et al. proposed a method based on formal concept lattices to discover roles with semantic meanings [12] as well as a method based on logistic PCA (Principal Component Analysis) to eliminate data noises [17]. Du X and Change X proposed two algorithms based on artificial intelligence, i.e., the genetic algorithm and ant colony optimization algorithm [18]. Dong et al. proposed both fast exact and heuristic methods based on biclique network cover to minimize role number or edge number [19].

With regard to goodness measure, several metrics have been proposed in the literature, including minimizing the number of roles [10, 20], minimizing the number of edges [13, 14, 19], minimizing the number of user-role assignment and permission-role assignment relations [13], minimizing both the number of roles and edges [21], and minimizing the administrative cost [22]. These optimization goals can be uniformly represented by the Weighted Structural Complexity (WSC) [8, 9].

Although there are a lot of effective role mining approaches, most of them neglect the relationships between user permissions. From the perspective of network security management, user permissions are not independent. A user or potential attacker may get extra permissions from the preassigned permissions, which may introduce fatal risks to network security. Hence, in the framework RMMDI, we model the interrelationships between user permissions from multi-domain configuration information and get more reasonable user roles, mitigating the vulnerabilities and strengthening network security.

2.2. Multi-Domain Information Modeling. Traditional network security analysis mainly concentrates on the network domain, with a few concerns on other domains. However, with the deepening of research on insider threat, an increasing number of studies have shown that the attacker will attack the network not only in digital ways, but also through the physical domain and social domain.

The existing methods of joint modeling of network multi-domain information mainly define multi-domain information by using the formalized methods and then make inference based on the logical rules to judge whether the system can reach the unsafe state. Probst et al. proposed a formal model for describing scenarios that span the physical and digital domain [23, 24]. Kotenko et al. proposed a model for describing attacks that use social engineering and physical access based on the preconditions and postconditions of atomic actions [25]. Scott et al. built a security model that adds a spatial relationship between the elements in the ambient calculus [26]. Dimkov presented a security model named as Portunes graph to abstract the environment of an organization into a stratified graph, which involved the information in physical, digital, and social domain information [27]. Kammuller and Probst combined formal modeling and analysis of infrastructures of organizations with a sociological explanation to provide a framework for insider threat analysis [28].

In this paper, we take possible interaction effects among multi-domain permissions into consideration, which are the basis of similar permission finding and role mining based on multiple domain information.

2.3. Multi-View Community Detection. The community is a universal property in many complex networks, which means that network nodes can be divided into small groups [26]. Traditional community detection methods only utilize single network information. And several multiview community

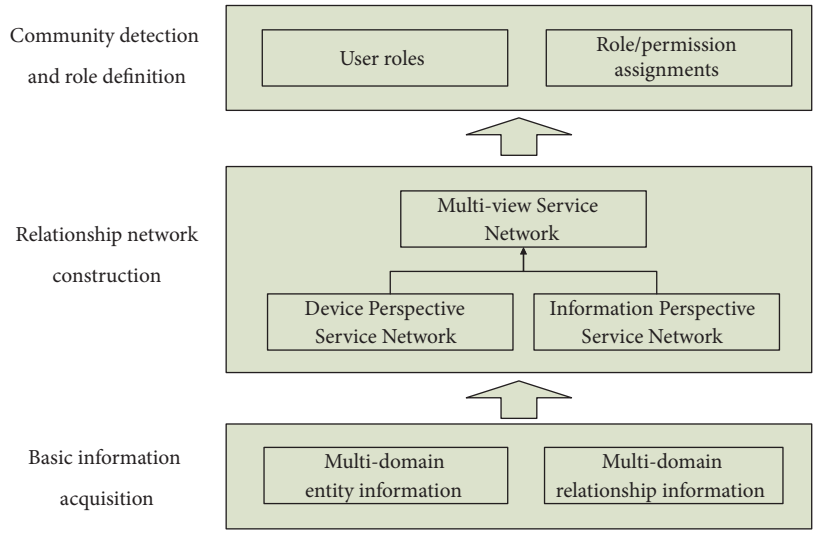


Figure 1: Role mining framework based on multi-domain information.

detection methods have been proposed, which utilize more information and achieve better performance.

Nonnegative Matrix Factorization (NMF) [29] is a classic clustering method, and several multi-view community detection methods based on NMF are proposed. Akata et al. proposed a method to jointly factorize multiple data matrices through a shared coefficient matrix [30]. Liu et al. proposed MultiNMF that regularizes the coefficient matrices learned from different views towards a common consensus for clustering [31]. He et al. extended NMF for multiview clustering by jointly factorizing the multiple matrices through coregularization [32]. Pei et al. proposed a nonnegative matrix tri-factorization (NMTF) based clustering framework with three types of graph regularization [33]. Li et al. proposed a framework based on regularized joint nonnegative matrix factorization (RJNMF) to utilize link and content information jointly to enhance the community detection accuracy [34].

In the framework RMMDI, we use the Pairwise Coregularized NMF clustering algorithm proposed in [32] to merge the information from two service networks (views). Experiments show that it can get more information than from a single view and make the role mining results more reasonable.

3. Role Mining Framework Based on Multi-Domain Information

In this paper, we proposed a role mining framework based on the multi-domain information, which is named as RMMDI. The framework is aimed at dividing possible user permissions into several disjoint subsets and assigning each subset to a user role. Then users are assigned with one or more necessary roles according to the permission they deserve. The structure of RMMDI is shown in Figure 1. The framework can be divided into three modules: basic information acquisition, relationship network construction, and community detection and role definition.

The basic information acquisition module obtains the necessary basic information from the target network, including multi-domain entity information and relationship information. The relationship network construction module constructs eight networks based on the obtained basic information, including the intermediate networks and ultimate networks. The community detection and role definition module detects permission communities on the ultimate networks by a multi-view community detection method and defines possible user roles.

3.1. Basic Information Acquisition. The basic information acquisition module is to collect network basic information, including the entities and entity relationships in the physical domain, network domain, and information domain, which are the foundation of relationship network construction.

3.1.1. Entity. There are five kinds of entities involved in the framework, i.e., space, object, service, info, and user.

Entity space represents specific physical space such as city, campus, building, or room, which is in the physical domain. All the space entities are represented as a set NS. Entity object is also located at the physical domain and represents network device like router, switch, or terminal. All the object entities are represented as a set NO. Entity service is in the network domain and represents network service like HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), and Email. All the service entities are represented as a set NV. Entity info is in the digital domain and represents the information like password, data, or digital file. All the info entities are represented as a set NI. Entity user represents network user. All the user entities are represented as a set NU.

3.1.2. Relationships. There are seven kinds of relationships involved in the framework, i.e., spatial similarity relationships, containment relationships, service access relationships, local management relationships, remote management relationships, service domination relationships, and info domination relationships.

Spatial similarity relationships are described by the matrix $M^{SS} \in A^{S\times S}$, where $A = \{0, 1\}$ and $S = |NS|$. $M^{SS}(i, j)$ is determined by

$$M^{SS}(i,j) = \begin{cases} 1, & \text{if } (i = j) \text{ or } \mathrm{u}(i,j) + \mathrm{u}(j,i) > \varepsilon \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where $u(i, j)$ is the number of users who can move from space ns_i to space ns_j and ε is the threshold value, ranging from 0 to $2|NU|$.

Device containment relationships are described by the matrix $M^{OS} \in A^{O\times S}$, where $A = \{0, 1\}$, $O = |NO|$, and $S = |NS|$. $M^{OS}(i, j)$ is determined by the following.

$$M^{OS}(i,j) = \begin{cases} 1, & \text{if object } no_i \text{ locates in space } ns_j \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Service access relationships are described by the matrix $M^{VO} \in A^{V\times O}$, where $A = \{0, 1\}$, $V = |NV|$, and $O = |NO|$. $M^{VO}(i, j)$ is determined by the following.

$$M^{VO}(i,j) = \begin{cases} 1, & \text{if service } nv_i \text{ can be reach by device } no_j \\ 0, & otherwise \end{cases} \quad (3)$$

Local management relationships are described by the matrix $M^{OV\_L} \in A^{O\times V}$, where $A = \{0, 1\}$, $O = |NO|$, and $V = |NV|$. $M^{OV\_L}(i, j)$ is determined by the following.

$$M^{OV\_L}(\mathrm{i},\mathrm{j}) = \begin{cases} 1, & \text{if device } no_i \text{ can be managed by service } nv_j \text{ locally} \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

Remote management relationships are described by the matrix $M^{OV\_R} \in A^{O\times V}$, where $A = \{0, 1\}$, $O = |NO|$, and $V = |NV|$. $M^{OV\_R}(i, j)$ is determined by the following.

$$M^{OV\_R}(i,j) = \begin{cases} 1, & \text{if device } no_i \text{ can be managed by service } nv_j \text{ remotely} \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

Service domination relationships are described by the matrix $M^{VI} \in A^{V\times I}$, where $A = \{0, 1\}$, $V = |NV|$, and $I = |NI|$. $M^{VI}(i, j)$ is determined by the following.

$$M^{VI}(i,j) = \begin{cases} 1, & \text{if the password of service } nv_i \text{ is } ni_j \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

Info domination relationships are described by the matrix $M^{II} \in A^{I\times I}$, where $A = \{0, 1\}$ and $I = |NI|$. $M^{II}(i, j)$ is determined by

$$M^{II}(i,j) = \begin{cases} 1, & \text{if } \left(ni_j \longrightarrow ni_i\right) \text{ or } \left(ni_i \longrightarrow ni_j\right) \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

where symbol $a \longrightarrow b$ indicates that information a is dominated by information b. It means there is a service $v \in NV$, whose password is b and from which the users can get information a.

3.2. Relationship Network Construction. The relationship network construction module is to construct basic relationship networks based on the obtained basic information. As shown in Figure 2, there are eight networks to be constructed in total. The ultimate goal of this module is to form the Device View Service Network (DVSN), Information View Service Network (IVSN), and Multiview Service Network (MVSN). These three ultimate networks are used in community detection and role definition. Besides the three ultimate networks, there are other five networks involved in user-role mining, which are named as Local Management View Device Network (LMVDN), Remote Management View Device Network (RMVDN), Local Information View Device Network (LIVDN), Remote Information View Device Network (RIVDN), and Multiview Device Network (MVDN). The five intermediate networks are the foundation to construct ultimate networks. The meanings of intermediate networks and ultimate networks are described as follows.

3.2.1. Intermediate Networks. The five intermediate networks are described as undirected weighted graphs, whose adjacency matrices are constructed from the seven basic relationship matrices.

LMVDN. The LMVDN represents the similarity between devices from a spatial (local management) perspective, which means the devices located at similarity spaces are more similar than others. The network is represented by the adjacency matrix A^{OO_S}, whose values represent the similarity of two devices. The matrix is determined by the following.

$$A^{OO\_S} = M^{OS} M^{SS} \left(M^{OS}\right)^T \quad (8)$$

RMVDN. The RMVDN represents the similarity between devices from a remote management perspective, which means the devices that can be managed by similar management services are more similar than others. The network is represented by the adjacency matrix A^{OO_R}, whose values represent the similarity of two devices. The matrix is determined by the following.

$$A^{OO\_R} = M^{O\_VR} M^{VO} + \left(M^{O\_VR} M^{VO}\right)^T \quad (9)$$

LIVDN. The LIVDN represents the similarity between devices from the perspective of local management service password, which means the devices with similar local management service password are more similar than others. The network is represented by the adjacency matrix A^{OO_IL}, whose values represent the similarity of two devices. The matrix is determined by the following.

$$A^{OO\_IL} = M^{OV\_L} M^{VI} M^{II} \left(M^{OV\_L} M^{VI}\right)^T \quad (10)$$

RIVDN. The RIVDN represents the similarity between devices from the perspective of remote management service

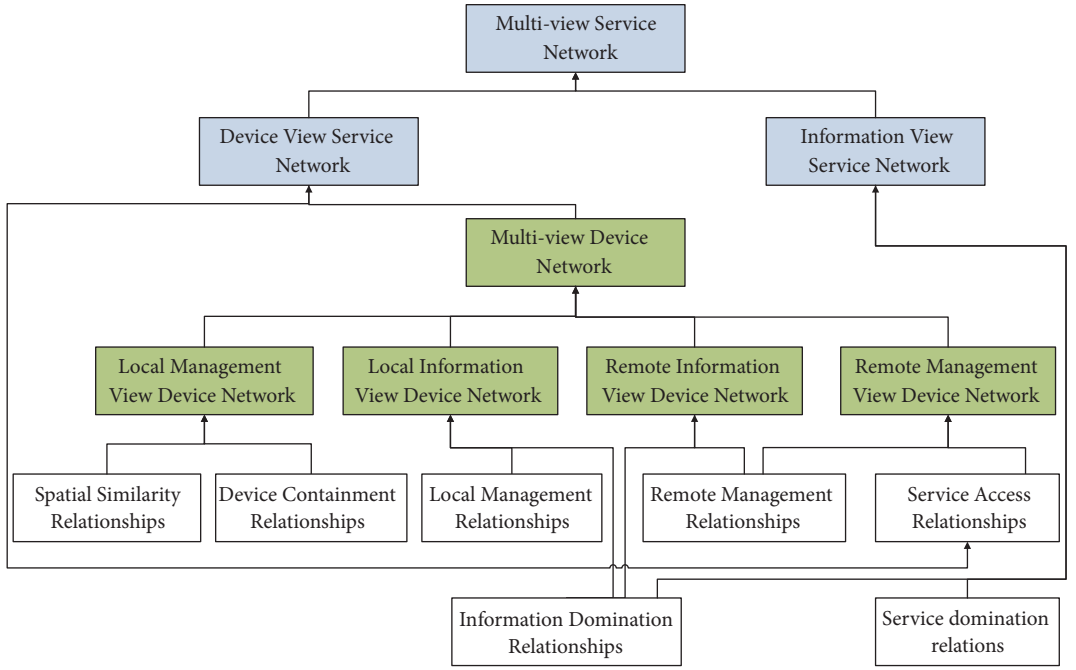


Figure 2: Relationship networks constructed in RMMDI.

password, which means the devices with similar remote management service password are more similar than others. The network is represented by the adjacency matrix A^{OO_IR}, whose values represent the similarity of two devices. The matrix is determined by the following.

$$A^{OO\_IR} = M^{OV\_R} M^{VI} M^{II} \left(M^{OV\_R} M^{VI}\right)^T \tag{11}$$

MVDN. The MVDN represents the similarity between devices from multiple perspectives, which merges the relationships from the local management perspective and the remote management perspective. The network is represented by the adjacency matrix A^{OO}, whose values represent the similarity of two devices. The matrix is determined by

$$A^{OO} = A^{OO\_S} \cdot A^{OO\_IL} + A^{OO\_R} \cdot A^{OO\_IR} \tag{12}$$

where the symbol $\cdot$ means dot product of two matrices.

3.2.2. Ultimate Networks. The three ultimate networks are also described as undirected weighted graphs, whose adjacency matrices are constructed from the seven basic relationship matrices and five intermediate networks.

DVSN. The DVSN represents the similarity of service permissions from a device perspective, which means the services accessed by similar devices are more similar than others. The network is represented by the adjacency matrix A^{VV_D}, whose values represent the similarity of two devices. The matrix is determined by

$$A^{VV\_D^*} = M^{VO} A^{OO} \left(M^{VO}\right)^T \tag{13}$$

$$A^{VV\_D} = \text{relationFilter}\left(A^{VV\_D^*}, \lambda\right) \tag{14}$$

where relationFilter(G, λ) is the function of filtering edges from the original graph, whose parameter G is the original graph and λ is the ratio of edges to be reserved.

As the matrix $A^{VV\_D^*}$ is a fully connected matrix in which the edges with small weight have negative impacts on community results, we use a function relationFilter(G, λ) to filter the low weight edges from the network. In function *relationFilter*(G, λ), for any node n in the graph G, we only reserve the top $\lambda \times \text{edgeNum}(n)$ edges with the largest weight. If two edges have the same weight, we reserve the edge between the node n and the neighbor node with a higher degree.

IVSN. The IVSN represents the similarity of service permissions from an information perspective, which means the services with a similar password are more similar than others. The network is represented by the adjacency matrix A^{VV_I}, whose values represent the similarity of two devices. The matrix is determined by

$$A^{VV\_I} = \text{relationFilter}\left(M^{VI} A^{II} \left(M^{VI}\right)^T, \lambda\right) \tag{15}$$

where relationFilter(G, λ) is the same edges filtering function in formula (14).

MVSN. The MVSN represents the similarity of service permissions from multiple perspectives, which merges the similarity relationships from the device perspective and the information perspective. The network is represented by the adjacency matrix A^{VV}, whose values represent the similarity of two devices. The matrix is determined by the following.

$$A^{VV} = A^{VV\_D} + A^{VV\_I} \tag{16}$$

Input: nonnegative matrices A^{VV_D}, A^{VV_I}. number of communities k, parameters $\lambda_D, \lambda_I, \lambda_{DI}$
Output: Service Community Division $C = \{c_1, c_2, \cdots, c_k\}$.
(1) Initialize $W^{VV\_D} \geq 0$, $H^{VV\_D} \geq 0$, $W^{VV\_I} \geq 0$, $H^{VV\_I} \geq 0$
(2) **While** Objective function does not converge and the Number of iterations is less than Threshold **do**
(3) Update H^{VV_D} according to Formula (18)
(4) Update H^{VV_I} according to Formula (19)
(5) Update W^{VV_D} according to Formula (20)
(6) Update W^{VV_I} according to Formula (21)
(7) **end while**
(8) Divide nodes to communities division $C = \{c_1, c_2, \cdots, c_k\}$ according to the coefficient matrix W^{VV_D}
(9) **return** $C = \{c_1, c_2, \cdots, c_k\}$

Algorithm 1: Permission community detection algorithm (PCDA).

3.3. Community Discovery and User-Role Definition. After building the ultimate networks, services can be divided into community relations through multi-view clustering algorithm, where all service permissions are divided into a community division $C = \{c_1, c_2, \cdots, c_k\}$. Then, for each $c_i \in C$, a role can be defined correspondingly. In this way, all network service permissions can be naturally assigned to k classes, where the possible values of k can be determined through algorithms such as maximum module degree.

In multiview service community discovery, we use the Pairwise Coregularized NMF clustering algorithm (PCoNMF) proposed in [32], which is based on regularized joint NMF. The objective function of service community discovery is formulated as follows.

$$\begin{aligned} J = {} & \lambda_D \left\| A^{VV\_D} - W^{VV\_D} \left(H^{VV\_D}\right)^{\mathrm{T}} \right\|_F^2 \\ & + \lambda_I \left\| A^{VV\_I} - W^{VV\_I} \left(H^{VV\_I}\right)^{\mathrm{T}} \right\|_F^2 \\ & + \lambda_{DI} \left\| W^{VV\_D} - W^{VV\_I} \right\|_F^2 \\ & s.t.\ H^{VV\_D} \geq 0,\ H^{VV\_I} \geq 0 \end{aligned} \tag{17}$$

The hypothesis behind PCoNMF is to regularize the coefficient matrices of the different views to a common consensus, which is then used for clustering. PCoNMF also adopts alternating optimization to minimize the objective function. The optimization works as follows: (1) fix the value of W^{VV_D} and W^{VV_I} while minimizing J over H^{VV_D} and H^{VV_I}; then (2) fix the value of H^{VV_D} and H^{VV_I} while minimizing J over W^{VV_D} and W^{VV_I}. We repeat the two steps until the iteration threshold is achieved.

According to [32], the update rules are as follows.

$$H^{VV\_D} \longleftarrow H^{VV\_D} \cdot \frac{\left(W^{VV\_D}\right)^T A^{VV\_D}}{\left(W^{VV\_D}\right)^T W^{VV\_D} A^{VV\_D}} \tag{18}$$

$$H^{VV\_I} \longleftarrow H^{VV\_I} \cdot \frac{\left(W^{VV\_I}\right)^T A^{VV\_I}}{\left(W^{VV\_I}\right)^T W^{VV\_I} A^{VV\_I}} \tag{19}$$

$$W^{VV\_D} \longleftarrow W^{VV\_D} \cdot \frac{\lambda_D A^{VV\_D} \left(H^{VV\_D}\right)^T + \lambda_{DI} W^{VV\_I}}{\lambda_D W^{VV\_D} H^{VV\_D} \left(H^{VV\_D}\right)^T + \lambda_{DI} W^{VV\_D}} \tag{20}$$

$$W^{VV\_I} \longleftarrow W^{VV\_I} \cdot \frac{\lambda_I A^{VV\_I} \left(H^{VV\_I}\right)^T + \lambda_{DI} W^{VV\_D}}{\lambda_I W^{VV\_I} H^{VV\_I} \left(H^{VV\_I}\right)^T + \lambda_{DI} W^{VV\_I}} \tag{21}$$

Hence, the permission community detection algorithm is shown as Algorithm 1.

4. Experiments and Results

In this section, we evaluate our role mining method based on the multi-domain information of a simulated network, which is the simplification of the inner network of Corporation M.

4.1. Experiment Environment. We built a simulation network for experiments, including a router, a firewall, an Intrusion Prevention System (IPS), 3 switches (Switch1, Switch2, and Switch3), 6 servers (WServer, DServer, FServer, GServer, OServer, and IServer), 3 gate machines (GM1, GM2, and GM3), and 13 terminals (T1, T2, T3, ..., T13). We used a HUAWEI S7706 as the core router, three HUAWEI S5700 as switches, a TOPSEC NGFW 4000-UF as the firewall, a TOPSEC IDP 3000 as IPS, and computers from Dell and HP as the servers or terminals. The router enabled 3-layer routing and the firewall were configured with bidirectional access control lists. All the servers and terminals were installed with different versions of Windows, including Windows 2003 Server, Windows XP, and Windows 7. We deployed an entrance guard system including 3 gate machines and a server (GServer). The gate machines used face recognition technology to determine whether a person can pass or not. An office automation system was deployed on the OServer, whose database was deployed on the DServer. We also deployed two websites and an FTP using IIS (Internet Information Services) on WServer, IServer, and FServer. Similarly, the websites depended on the same database deployed on

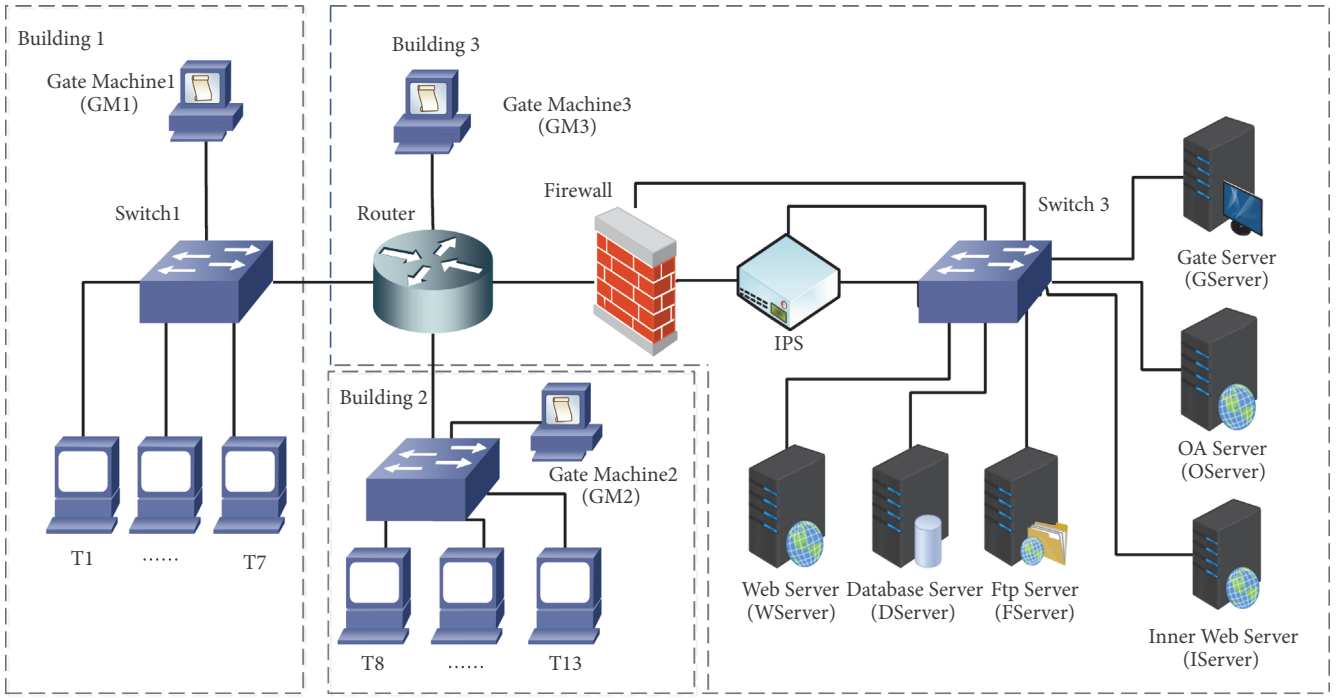


Figure 3: An example network.

DServer. The physical link relationships among devices are shown in Figure 3.

All the devices are distributed in 12 rooms in 3 buildings. 10 devices are located in building 1: terminal T1, T2, and T3 are in room 1-1; T4 and T5 are in room 1-4; T6 and T7 are in room 1-5; Switch1 is in room 1-2; and GM1 is in the hall of building 1 (room 1-3). 8 devices are located in building 2: terminals T8 and T9 are in room 2-1; T10 and T11 are in room 2-4; T12 and T13 are in room 2-5; Switch2 is in room 2-2; and GM2 is in the hall of building 2 (room 2-3). 10 devices are located in building 3: router, firewall, IPS, Switch3, and all servers are in room 3-1, and GM3 is in the hall of building 3 (room 3-2).

There were 34 services in the network, including 28 management services and 6 business services. The management services were used for device management, while the business services were used for corporation business. Each device was managed by a management service. The router and switches enabled SSH service. The servers and terminals enabled the Remote Desktop Service. In addition, the gate machines enabled web-based management interfaces. The website deployed on WServer provided a web service on port 80 named as WS_W, which was used to publish public information. The FServer provided an FTP service on port 21 named as FS_F, which was used by Network Administrators to share information. The GServer provided a data transmission service on port 8080 named as GS_T, which was used to synchronize data between GM machine and GServer. The OServer provided a web service on port 80 named as OS_W, which was used to document circulation for all users. The IServer provided a web service on port 80 named as IS_W, which was used by Server Administrators to share information. The DServer provided a database service on port 1433 named as DS_D, which was used to provide underlying support for WS_W, OS_W, and IS_W.

There were 33 passwords in the analysis. Each service, except for WS_W and OS_W, has a password. Besides, *NULL* was added to represent the empty password. All the information involved is shown in Table 1.

There were 13 users involved in analysis named from User1 to User13, who used terminals T1 to T13 and knew passwords T1_M_P to T13_M_P, respectively. Using the top-down approaches, the network security administrators had gotten 5 user roles for the business information, which were named as Ordinary User, Server Administrator, Database Administrator, Network Administrator, and Security Administrator. The role-permission assignments are listed in Table 2.

4.2. Baseline Methods. To demonstrate the effectiveness of our method, we compare our approach with two groups of baselines. The first group comprises 5 clustering methods: 2 single view methods and 3 multiview methods. The second group comprises 4 traditional role mining methods: ORCA (OFFIS Role mining tool with Cluster Analysis), CM (Complete Miner), HPr (HP Role Minimization), and HPe (HP Edge Minimization)

4.2.1. Clustering Methods. SP (Spectral Clustering). SP [35] is a classical single view clustering algorithm, which makes use of the eigenvalues of the data similarity matrix to perform dimensionality reduction before clustering in fewer dimensions. The similarity matrix is provided as an input, consisting of a quantitative assessment of the relative similarity of each pair of points in the dataset.

SymNMF. SymNMF [36] is a clustering algorithm based on NMF, which takes a nonnegative and symmetric matrix as

Table 1: Device related information.

Device	Location	Services	Password	Device	Location	Services	Password
T1	R1-1	T1_M	T1_M_P	GM2	R2-3	G2_M	G2_M_P
T2	R1-1	T2_M	T2_M_P	GM3	R3-2	G3_M	G3_M_P
T3	R1-1	T3_M	T3_M_P	Switch1	R1-2	S1_M	S1_M_P
T4	R1-4	T4_M	T4_M_P	Switch2	R2-2	S2_M	S2_M_P
T5	R1-4	T5_M	T5_M_P	Switch3	R3-1	S3_M	S3_M_P
T6	R1-5	T6_M	T6_M_P	Router	R3-1	R_M	R_M_P
T7	R1-5	T7_M	T7_M_P	Firewall	R3-1	F_M	F_M_P
T8	R2-1	T8_M	T8_M_P	IPS	R3-1	IPS_M	IPS_M_P
T9	R2-1	T9_M	T9_M_P	WServer	R3-1	WS_W WS_M	- - WS_M_P
T10	R2-4	T10_M	T10_M_P	DServer	R3-1	DS_D DS_M	DS_D_P DS_M_P
T11	R2-4	T11_M	T11_M_P	FServer	R3-1	FS_F FS_M	FS_F_P FS_M_P
T12	R2-5	T12_M	T12_M_P	GServer	R3-1	GS_T GS_M	GS_T_P GS_M_P
T13	R2-5	T13_M	T13_M_P	OServer	R3-1	OS_W OS_M	– OS_M_P
GM1	R1-3	G1_M	G1_M_P	IServer	R3-1	IS_W IS_M	IS_W_P IS_M_P

Table 2: User role-permission assignments by top-down methods.

Roles	Service Permissions
Ordinary User	WS_W, OS_W
Server Administrator	WS_M, FS_M, GS_M, OS_M, IS_M, DS_M, IS_W
Database Administrator	DS_D
Network Administrator	S1_M, S2_M,S3_M, R_M, FS_F
Security Administrator	F_M, IPS_M, G1_M, G2_M, G3_M, GS_T

an input. The matrix contains pairwise similarity values of a similarity graph and is approximated by a lower rank matrix instead of the product of two lower rank matrices.

PCoSpec (Pairwise Coregularized Spectral clustering) and CCoSpec (Center-wise Coregularized Spectral clustering). Two coregularization schemes are adopted in spectral clustering framework [37], PCoSpec utilizes a pairwise coregularization to enforce the eigenvectors of each pair to be similar, and CCoSpec employs the centroid-based coregularization to enforce the eigenvectors to be similar with a common center.

CCoNMF (Cluster-wise Coregularized NMF clustering). CCoNMF extends NMF for multiview clustering by jointly factorizing the multiple matrices through cluster-wise coregularization [32], which enforces the cluster similarity matrices to be similar.

RMSC (Robust Multiview Spectral Clustering). RMSC [38] is a multiview spectral clustering method based on Markov chain, which explicitly handles the possible noise in the transition probability matrices associated with different views.

4.2.2. Mining Baseline Methods. ORCA. ORCA [11] is the first role mining algorithm, which uses the hierarchical clustering technology to discover user roles. The algorithm defines each permission as an initial cluster first, then merges the clusters, and forms a role hierarchy.

Complete Miner (CM). CM [10] is another classic role mining algorithm proposed in 2006. It starts by creating an initial set of roles for the distinct user-permission sets, then computes all possible intersection sets of the initial roles, and outputs a list of candidate roles.

HP Role Minimization and HP Edge Minimization. HP Role Minimization (HPr) and HP Edge Minimization (HPe) [14] are the role mining algorithms based on minimum biclique coverage. HPr tries to find a minimal set of roles that override the user-permission assignment relationship, while HPe uses a heuristic method to find the smallest number of edges of an RBAC system.

4.3. Experiments Setup

4.3.1. Scenarios Construction. To validate our framework and method, we built 3 scenarios named as Scenario1 (S1), Scenario2 (S2), and Scenario3 (S3) based on the basic experimental environment shown in Figure 3 and assigned users one or more user roles, which is shown in Table 3. We can find out that each user in S1 was assigned only 1 user role, while they were assigned 2 user roles in S2. In S3, users working

Table 3: User-role assignments in different scenario.

Scenario	Ordinary User	Server Administrator	Database Administrator	Network Administrator	Security Administrator
S1	User1, User2, User3	User4, User5	User6, User7	User8, User9, User10,User11	User12, User13
S2	All Users	User7, User8	User11, User12, User13	User4, User5	User9, User10
S3	User1, User2, User5, User6, User9, User10, User13	User4, User7	User8	User11	User12

in one room may be assigned different user roles, which may introduce more vulnerabilities to network security.

For each scenario, we first configured the gate machines and firewall according to Tables 2 and 3. Spatial access control lists were added on gate machines, making users have physical access to devices they managed or used, while network access control lists were added on the firewall, making the terminals have network access to the target services.

It should be noted that there were potential conflicts among multi-domain configurations on the semantic level. Take the user User4 in S1 as example. User4 was a Server Administrator and should not access service DS_M and the firewall had forbidden T4 to access service DS_D directly, but T4 was permitted to access service WS_M and there was no firewall between WServer and DServer. Thus, User4 can use T4 to log in WServer remotely first and then access service DS_D (he can get the password DS_D_P from the configuration files on WServer). This is a typical semantic conflict between the network access control lists. Similarly, as User4 had the ability to access DS_D physically by entering the room 3-1, there is another conflict between the network access control list and the spatial access control list. Those conflicts may result in extra permissions for users.

Then, we extracted basic information from the network and established the necessary relationship matrices. Note that there were 16 space entities, 28 object entities, 34 service entities, 32 information entities, and 13 user entities in all the 3 scenarios. The relationship matrices $M^{OS} \in \mathbb{R}^{28\times16}$, $M^{VI} \in \mathbb{R}^{34\times32}$, $M^{OV\_L} \in \mathbb{R}^{28\times34}$, and $M^{OV\_R} \in \mathbb{R}^{28\times34}$ were the same in all three scenarios, where $|M^{OS}|_0 = |M^{OV\_L}|_0 = |M^{OV\_R}|_0 = 28$ and $|M^{VI}|_0 = 34$. The matrices $M^{SS} \in \mathbb{R}^{16\times16}$ and $M^{VO} \in \mathbb{R}^{34\times28}$ varied from scenario to scenario, depending on the configurations of gate machines or firewall. For each space pair (S_1, S_2), we counted the users who can move from S_1 to S_2 under each scenario and set the parameter $\varepsilon = 14$ when constructing the matrix M^{SS}. The results showed that $|M^{SS}|_0 = 46$ in S1, $|M^{SS}|_0 = 46$ in S2, and $|M^{SS}|_0 = 42$ in S3. We used the scanner NMAP to get the accessibility relationships between devices and services, establishing the matrix M^{VO}. The results showed that $|M^{VO}|_0 = 549$ in S1, $|M^{VO}|_0 = 566$ in S2, and $|M^{SS}|_0 = 548$ in S3.

Finally, we detected the user roles by RMMDI and compared the results with the two groups of baseline methods. On the one hand, we performed the role mining baseline methods based on the user-permission assignment (UPA) matrices constructed from the firewall configurations and compared the results with RMMDI. On the other hand, we studied the best parameters for each clustering method and then compared the effectiveness of RMMDI with the clustering baseline methods. Accuracy and normalized mutual information (NMI) [31–33, 36] were adopted to evaluate the community detection effectiveness of different parameters, whose values both range from 0 to 1 and a higher value means better effectiveness. We calculated the accuracy and NMI of different clustering methods with ground truth after studying the best parameters for each clustering method. In the experiments, we constructed two ground truths manually. In one ground truth, we divided 21 service permissions into 5 roles according to Table 2. In the other ground truth, we combined the roles "Database Administrator" with "Server Administrator" and classified 21 service permissions into 4 roles. For one community detection result, we compared it with the two ground truths and calculated metrics separately.

4.4. Result

4.4.1. Role Mining Results. Firstly, we performed the baseline role mining methods based on the firewall configurations. As the firewall only conducted the network access control lists, it can only reflect the accessibility between devices and services. Since each terminal was assigned to a user, we can get the 3 different UPA matrices from it. As we wanted to find disjoint service subnets, we used $W = \langle 1, 1, 1, \infty, \infty\rangle$ as the optimization objective. In order to save space, only the permission divisions are shown in Table 4. We found that the permission divisions were almost the same as those in Table 2, which meant that the role mining methods can find no errors from the top-down approach.

Then, we also performed the RMMDI on all scenarios with the role number k = 5,100 times for each scenario. The majority of the results were different from the result shown in Table 2. The most frequent result (222 in 300 times) is listed as Table 5. Comparing with Table 2, we counted the number of inconsistent classification results of each service. All 18 services were classified inconsistently for 572 times in total. And the top 2 services with the most inconsistent

Table 4: Role mining results of baselines on all scenarios.

Scenario	Method	Role	Permissions
S1	ORCA CM HPr HPe	Role1	WS_W, OS_W
		Role2	WS_M, FS_M, GS_M, OS_M, IS_M, DS_M, IS_W
		Role3	DS_D
		Role4	S1_M, S2_M,S3_M, R_M, FS_F
		Role5	F_M, IPS_M, GS_T
S2	HPr	Role1	WS_W, OS_W
		Role2	WS_W, OS_W, WS_M, FS_M, GS_M, OS_M, IS_M, DS_M, IS_W
		Role3	WS_W, OS_W, DS_D
		Role4	WS_W, OS_W, S1_M, S2_M, S3_M, R_M, FS_F
		Role5	WS_W, OS_W, F_M, IPS_M, GS_T
S2	ORCA CM HPe	Role1	WS_W, OS_W
		Role2	WS_M, FS_M, GS_M, OS_M, IS_M, DS_M, IS_W
		Role3	WS_W, OS_W, DS_D
		Role4	S1_M, S2_M, S3_M, R_M, FS_F
		Role5	F_M, IPS_M, GS_T
S3	ORCA CM HPr HPe	Role1	WS_W, OS_W
		Role2	WS_M, FS_M, GS_M, OS_M, IS_M, DS_M, IS_W
		Role3	DS_D
		Role4	S1_M, S2_M,S3_M, R_M, FS_F
		Role5	F_M, IPS_M, GS_T

Table 5: The most common result of RMMDI when k=5.

Method	Role	Permissions
RMMDI	Role1	WS_W, OS_W
	Role2	WS_M, FS_M, GS_M, OS_M, IS_M, DS_M, IS_W, DS_D
	Role3	GS_T
	Role4	S1_M, S2_M,S3_M, R_M, FS_F
	Role5	F_M, IPS_M

classification times were DS_D (282 times) and GS_T (258 times).

Finally, we changed the role number $k = 4$ and performed the experiments 100 times under each scenario. The most common results are shown in Table 6.

4.4.2. Parameter Study Results. We studied the parameters used in RMMDI as well as the baseline clustering methods. We performed a series of experiments for a series of different parameters and tried to find out the optimal parameters. The experiments were conducted under S2 with role number $k = 4$.

We first studied the parameters used in baseline methods, including λ_{p_spec} in PCoSpec, $\lambda_{c\_spec\_D}$ and $\lambda_{c\_spec\_I}$ in CCoSpec, and λ_{rmsc} in RMSC. With each parameter, we set $\lambda = 0.05$ and studied 30 different values between 0.005 and 100. When $\lambda_{p\_spec} = 0.15$, $\lambda_{c\_spec\_D} = 8$, $\lambda_{c\_spec\_I} = 1$, and $\lambda_{rmsc} = 0.15$, the methods had relatively better effectiveness on the dataset.

Then, we studied the parameters in PCoNMF and CCoNMF. There are 3 parameters: λ_D, λ_I, and λ_{DI}. λ_D and λ_I represent the weights of view A^{VV_D} and A^{VV_I}, while λ_{DI} is the regularization parameter. We studied 27 different ratios of λ_D to λ_I between 10 and 0.02, as well as 10 different values of λ_{DI} between 0.1 and 10. The experiments were conducted 50 times for each pair. The results are shown in Figures 4 and 5. We found that the parameters λ_D and λ_I had little impact on both the accuracy and NMI when $\lambda_D/\lambda_I < 1$ and $0.5 < \lambda_{DI} < 1.5$, so we used $\lambda_D = 1$, $\lambda_I = 3$, and $\lambda_{DI} = 0.9$ in the study of λ and the clustering experiments shown in Section 4.4.3.

Finally, we studied the parameter λ used in RMMDI. We performed an experiment with a series of λ from 0.05 to 1 with step size 0.5 and observed their impacts on the clustering

Table 6: The most common result of RMMDI when k=4.

Method	Role	Permissions
RMMDI	Role1	WS_W, OS_W
	Role2	WS_M, FS_M, GS_M, OS_M, IS_M, DS_M, IS_W, DS_D
	Role3	S1_M, S2_M,S3_M, R_M, FS_F
	Role4	F_M, IPS_M, GS_T

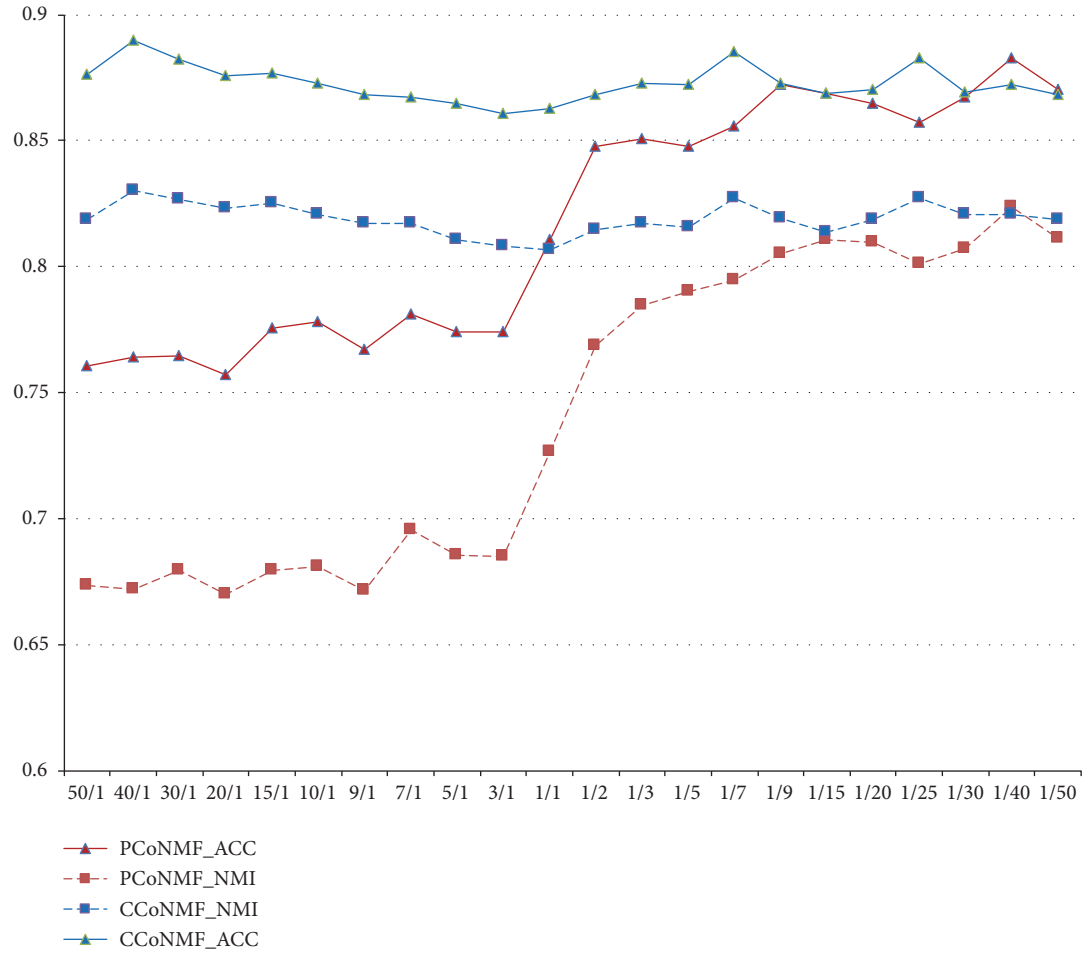


Figure 4: Evaluating the accuracy and NMI of PCoNMF and CCoNMF on varying λ_D/λ_I.

effectiveness (shown in Figure 6). The other parameters were set as mentioned in the previous paragraph.

We found that the curves showed a downward trend in whole, and the accuracy and NMI got greater values when λ was around 0.3, which was used for the experiments shown in Section 4.4.3.

4.4.3. Clustering Results. We also conducted experiments to compare the effectiveness of RMMDI with the clustering baseline methods. We performed all algorithms 200 times on each scenario and compared results with the ground truth shown in Table 4. All the other parameters were set as the optimal values mentioned in Section 4.4.2. The results are listed in Tables 7–9, in which the results of SP and SymNMF were the larger result of 3 different inputs, A^{VV}, A^{VV_D}, and A^{VV_I}. Most of those values were from the input A^{VV_I}.

5. Discussion

We propose a novel user role framework, which uses multiple domain information to mine user roles other than the preassigned user-permission assignment matrix.

It is proved that the framework is suitable for role mining. For the three scenarios used in the experiment, different users are assigned to different user roles. One user may be assigned one or more user roles, and one user role may be assigned to several users. For the results listed in Tables 7 and 8, we can find that the accuracy of the proposed framework is greater

Table 7: Accuracy for different methods on 3 scenarios.

Scenario	SP	SymNMF	PCoSpec	CCoSpec	PCoNMF	CCoNMF	RMSC
S1	0.9010	0.9229	0.7121	0.9057	*0.9381*	0.9190	0.6952
S2	0.8981	0.9276	0.6675	0.9072	*0.9428*	0.9365	0.5762
S3	0.9210	0.9181	0.7070	0.9035	*0.9365*	0.9333	0.6333

Table 8: NMI for different methods on 3 scenarios.

Scenario	SP	SymNMF	PCoSpec	CCoSpec	PCoNMF	CCoNMF	RMSC
S1	0.8605	0.8571	0.5827	0.8414	*0.8738*	0.8579	0.6382
S2	0.8506	0.8703	0.4869	0.8497	*0.8879*	0.8744	0.479
S3	0.8692	0.8538	0.5958	0.8464	*0.8774*	0.8719	0.4890

Table 9: Runtime for different methods on 3 scenarios (s).

Scenario	SP	SymNMF	PCoSpec	CCoSpec	PCoNMF	CCoNMF	RMSC
S1	0.0178	0.0109	0.1367	0.1247	0.8533	*1.1568*	0.0580
S2	0.0145	0.0091	0.0973	0.1108	0.8518	*1.0866*	0.0440
S3	0.0127	0.0085	0.0929	0.1031	0.8494	*1.2394*	0.0230

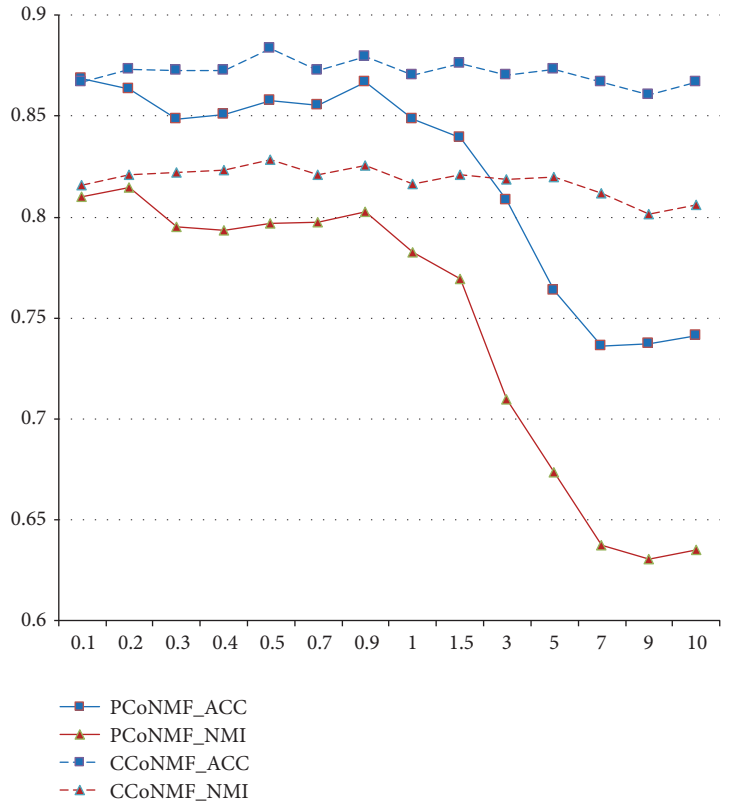
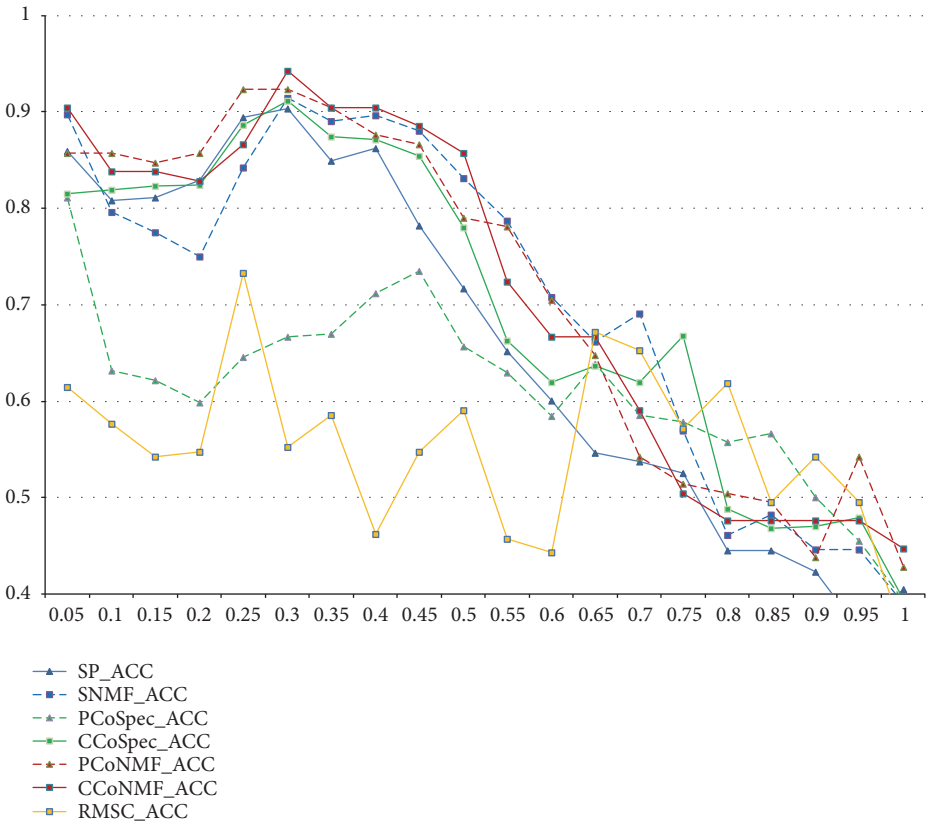


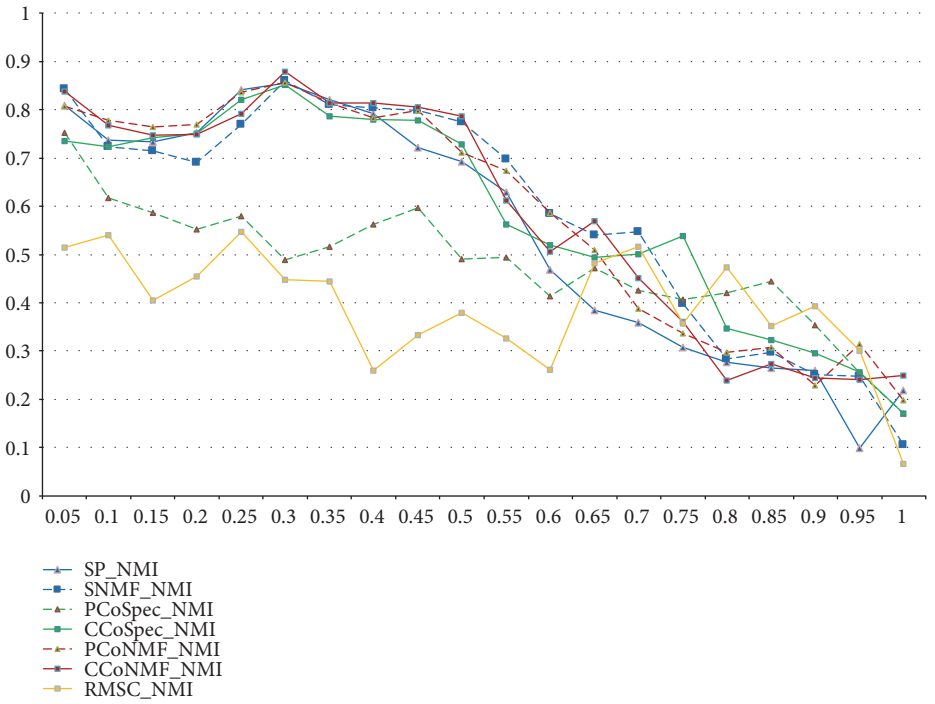
Figure 5: Evaluating the accuracy and NMI of PCoNMF and CCoNMF on varying λ_{DI}.

than 93.5% in all the three scenarios, while the NMI is greater than 87.0%. It means that the framework can detect user roles from the multiple domain configuration information successfully.

More importantly, it is also demonstrated that the framework has the ability to find interdependent relationships between permissions, avoiding potential errors. From the experimental results in Section 4.4.1, we find that RMMDI tends to integrate user roles "Database Administrator" and "Server Administrator". Analyzing user potential permissions, it can be found that all Server Administrators can access service DS_D as they can both reach the service from



(a) Evaluating the accuracy of RMMDI on varying λ



(b) Evaluating the NMI of RMMDI on varying λ

Figure 6: Evaluating the accuracy and NMI of RMMDI on varying λ.

other servers and get its password from configuration files in WServer. Therefore, it may be more appropriate to integrate user roles "Database Administrator" and "Server Administrator". This trend cannot be found by traditional methods.

It is also proved that the performances of different clustering methods vary in the framework. As shown in Tables 7 and 8, the accuracy and NMI of PCoNMF are always the best among all the three scenarios, 30% better than the worst method. It means that a reasonable clustering method will promote the effectiveness of the framework significantly. Compared with the single view clustering methods, PCoNMF promotes the accuracy and NMI by more than 1%, which means the reasonable utilization of information from multiple views will get more structure information than from single view. Both the PCoSpec and RMSC have a lower accuracy or NMI, which means the multiview methods based on spectral clustering may not be suitable to the datasets.

There are 4 parameters involved in the RMMDI in total and it is important to select proper values to the parameters. The first two parameters λ_D and λ_I are the weights of views A^{VV_D} and A^{VV_I}. From the results in Figure 4, we find that the view A^{VV_I} has more structure information than A^{VV_D} in the experiments and it is reasonable to set a larger λ_I and a smaller λ_D. The third parameter λ_{DI} is the regularization parameter that indicates the degree of community proximity between the two perspectives. A too low λ_{DI} will not establish the connection between two views. Nevertheless, as the view A^{VV_I} has more structure information than A^{VV_D}, a too big λ_{DI} will reduce the accuracy of the algorithm. Therefore, a moderate parameter λ_{DI} ($0.5 < \lambda_{DI} < 1.5$) will make the algorithm perform better. The last parameter λ is used in the function relationFilter(G, λ), which means to reserve the top $\lambda \times edgeNum(n)$ edges with the largest weight. From the results shown in Figure 6, we find that it is vital to reserve an appropriate proportion of links. A big λ will reserve more low weight links and the existences of low weight links will impact the effectiveness of the framework. However, a low value of λ will lost a lot of useful structure information, which will have an impact on the effectiveness of the algorithm too.

6. Conclusion

In this paper, a novel framework for role mining based on multi-domain information named as RMMDI is proposed. The key idea of the framework is to mine user roles from multiple domain information rather than existing user-permission assignment matrices. In the framework, information from the physical domain, network domain, and digital domain is used to find the relationships between user permissions, and multi-view community detection methods are used to integrate information from different domains. Experiments on 3 simulated network scenarios demonstrate that RMMDI can capture the interdependent relationships between permissions and perform user-role mining more effectively and reasonably.

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 is supported by the grants from the National Key R&D Program of China (Project No. 2017YFB0802800).

References

[1] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *The Computer Journal*, vol. 29, pp. 38–47, 1996.

[2] A. Colantonio, R. D. Pietro, A. Ocello, and N. V. Verde, "A formal framework to elicit roles with business meaning in RBAC systems," in *Proceedings of the 14th ACM Symposium on Access Control Models And Technologies*, pp. 85–94, ACM, Stresa, Italy, 2009.

[3] A. Baumgrass, M. Strembeck, and S. Rinderle-Ma, "Deriving role engineering artifacts from business processes and scenario models," in *Proceedings of the 16th ACM Symposium on Access Control Models and Technologies*, pp. 11–20, ACM, Innsbruck, Austria, 2011.

[4] A. Colantonio, R. Di Pietro, A. Ocello, and N. V. Verde, "Taming role mining complexity in RBAC," *Computers & Security*, vol. 29, pp. 548–564, 2010.

[5] A. Colantonio, R. Di Pietro, and N. V. Verde, "A business-driven decomposition methodology for role mining," *Computers & Security*, vol. 31, no. 7, pp. 844–855, 2012.

[6] S. Hachana, F. Cuppens, N. Cuppens-Boulahia, and J. Garcia-Alfaro, "Semantic analysis of role mining results and shadowed roles detection," *Information Security Technical Report*, vol. 17, pp. 131–147, 2013.

[7] M. Kuhlmann, D. Shohat, and G. Schimpf, "Role mining - revealing business roles for security administration using data mining technology," in *Proceedings of the Eighth ACM Symposium on Access Control Models and Technologies*, pp. 179–186, ACM, Como, Italy, 2003.

[8] I. Molloy, N. Li, T. Li, Z. Mao, Q. Wang, and J. Lobo, "Evaluating role mining algorithms," in *Proceedings of the ACM Symposium on Access Control Models and Technologies*, pp. 95–104, 2009.

[9] J. Jiang, X. Yuan, and R. Mao, "Research on role mining algorithms in RBAC," in *Proceedings of the 2018 2nd High Performance Computing and Cluster Technologies Conference*, pp. 1–5, ACM, 2018.

[10] J. Vaidya, V. Atluri, and J. Warner, "RoleMiner: mining roles using subset enumeration," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pp. 144–153, ACM, Alexandria, Va, USA, 2006.

[11] J. Schlegelmilch and U. Steffens, "Role mining with ORCA," in *Proceedings of the Tenth ACM Symposium on Access Control Models and Technologies*, pp. 168–176, ACM, Stockholm, Sweden, 2005.

[12] I. Molloy, H. Chen, T. Li et al., "Mining roles with semantic meanings," in *Proceedings of the ACM Symposium on Access Control Models and Technologies, SACMAT '08*, pp. 21–30, Estes Park, Colo, Usa, 2008.

[13] D. Zhang, K. Ramamohanarao, and T. Ebringer, "Role engineering using graph optimisation," in *Proceedings of the 12th ACM*

Symposium on Access Control Models and Technologies, pp. 139–144, ACM, Sophia Antipolis, France, June 2007.

[14] A. Ene, W. Horne, N. Milosavljevic et al., *Fast Exact and Heuristic Methods for Role Minimization Problems*, 2008.

[15] M. Frank, D. Basin, and J. M. Buhmann, "A class of probabilistic models for role engineering," in *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pp. 299–310, ACM, Alexandria, Va, USA, 2008.

[16] M. Frank, A. P. Streich, D. A. Basin et al., "A probabilistic approach to hybrid role mining," in *Proceedings of the Acm Conference on Computer & Communications Security*, 2009.

[17] I. Molloy, N. Li, Y. Qi et al., "Mining roles with noisy data," in *Proceedings of the ACM Symposium on Access Control MODELS and Technologies*, pp. 45–54, 2010.

[18] X. Du and X. Chang, "Performance of AI algorithms for mining meaningful roles," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2070–2076, 2014.

[19] L. Dong, Y. Wang, R. Liu, B. Pi, and L. Wu, "Toward edge minability for role mining in bipartite networks," *Physica A: Statistical Mechanics and its Applications*, vol. 462, pp. 274–286, 2016.

[20] L. Wu, L. Dong, Y. Wang et al., "Uniform-scale assessment of role minimization in bipartite networks and its application to access control," *Physica A: Statistical Mechanics and its Applications*, vol. 507, pp. 381–397, 2018.

[21] Q. Guo, J. Vaidya, and V. Atluri, "The role hierarchy mining problem: discovery of optimal role hierarchies," 2008.

[22] A. Colantonio, R. D. Pietro, and A. Ocello, "A cost-driven approach to role engineering," in *Proceedings of the Acm Symposium on Applied Computing*, 2008.

[23] C. W. Probst, R. R. Hansen, and F. Nielson, *Where Can an Insider Attack?* Springer, Berlin, Germany, 2007.

[24] C. W. Probst and R. R. Hansen, "An extensible analysable system model," in *Elsevier Advanced Technology Publications*, vol. 13, pp. 235–246, 2008.

[25] I. Kotenko, M. Stepashkin, and E. Doynikova, "Security analysis of information systems taking into account social engineering attacks," in *Proceedings of the 2011 19th International Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pp. 611–618, 2011.

[26] D. Scott, A. Beresford, and A. Mycroft, "Spatial policies for sentient mobile applications," in *Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks*, p. 147, IEEE Computer Society, 2003.

[27] T. Dimkov, *Alignment of Organizational Security Policies: Theory and Practice*, University of Twente, Enschede, Netherlands, 2012.

[28] F. Kammüller and C. W. Probst, "Invalidating policies using structural information," in *Proceedings of the 2013 IEEE Security and Privacy Workshops*, pp. 76–81, 2013.

[29] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[30] A. Wendel, S. Sternig, M. Godec et al., "Non-negative matrix factorization in multimodality data for segmentation and label prediction," *Computer Vision Winter Workshop*, 2011.

[31] J. Liu, C. Wang, J. Gao et al., "Multi-View clustering via joint nonnegative matrix factorization," in *Proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 252–260, 2013.

[32] X. He, M.-Y. Kan, P. Xie, and X. Chen, "Comment-based multi-view clustering of web 2.0 items," in *Proceedings of the 23rd International Conference on World Wide Web*, pp. 771–782, ACM, Seoul, Korea, 2014.

[33] Y. Pei, N. Chakraborty, and K. Sycara, "Nonnegative matrix tri-factorization with graph regularization for community detection in social networks," in *Proceedings of the 24th International Conference on Artificial Intelligence*, pp. 2083–2089, AAAI Press, Buenos Aires, Argentina, 2015.

[34] Z. Li, Z. Pan, Y. Zhang, G. Li, and G. Hu, "Efficient community detection in heterogeneous social networks," *Mathematical Problems in Engineering*, vol. 2016, Article ID 5750645, 15 pages, 2016.

[35] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[36] D. Kuang, S. Yun, and H. Park, "SymNMF: nonnegative low-rank approximation of a similarity matrix for graph clustering," *Journal of Global Optimization*, vol. 62, pp. 545–574, 2015.

[37] A. Kumar, P. Rai, and H. Daumé, "Co-regularized multi-view spectral clustering," *Advances in Neural Information Processing Systems*, 2011.

[38] R. Xia, Y. Pan, L. Du et al., "Robust multi-view spectral clustering via low-rank and sparse decomposition," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 2149–2155, AAAI Press, Québec, Canada, 2014.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 5483918, 9 pages
https://doi.org/10.1155/2019/5483918

Research Article

HeteMSD: A Big Data Analytics Framework for Targeted Cyber-Attacks Detection Using Heterogeneous Multisource Data

Ankang Ju [1], Yuanbo Guo [1], Ziwei Ye [1], Tao Li [1], and Jing Ma [2]

[1] *Zhengzhou Institute of Information Science and Technology, 450001, China*
[2] *Science and Technology on Information Assurance Laboratory, 100071, China*

Correspondence should be addressed to Ankang Ju; jusissp@yeah.net

Received 25 January 2019; Accepted 10 April 2019; Published 2 May 2019

Guest Editor: Pelin Angin

In the current enterprise network environment, multistep targeted cyber-attacks with concealment and advanced characteristics have become the main threat. Multisource security data are the prerequisite of targeted cyber-attacks detection. However, these data have characters of heterogeneity and semantic diversity, and existing attack detection methods do not take comprehensive data sources into account. Identifying and predicting attack intention from heterogeneous noisy data can be meaningful work. In this paper, we first review different data fusion mechanisms of correlating heterogeneous multisource data. On this basis, we propose a big data analytics framework for targeted cyber-attacks detection and give the basic idea of correlation analysis. Our approach will offer the ability to correlate multisource heterogeneous security data and analyze attack intention effectively.

1. Introduction

In the current network environment, network attacks are more covert and systematic. Advanced Persistent Threat (APT) brings huge economic losses and security risks to governments, enterprises, and other institutions [1]. Network security administrators do not realize that their network has been compromised until weeks, months, or even years later. Traditional detection techniques cannot ideally handle targeted cyber-attacks with characteristics of complexity and customization. Multistep targeted cyber-attacks have become the most critical factor affecting network security [2].

The significance of timely detection and analysis of targeted cyber-attacks lies in two aspects [3]. First, timely detection of intrusion behaviors: illegal system services can be cleaned up and recover from a crash in time after the system has been attacked. That can reduce losses caused by less normal service hours. Second, targeted cyber-attacks are often implemented in multiple stages, and the abnormal behaviors detected currently may not rebuild the whole attack sequence. Timely detection of preattack steps can help us understand the intention of an attack. Detecting and predicting follow-up attacks in time can help us take protective measures to prevent further damage caused by follow-up attacks.

In the current research of intrusion detection technology, it can be divided into host detection and network detection [4]. Host-based intrusion detection method analyzing procedure behavior to find payload is represented by the malicious code. Network-based intrusion detection mainly analyses network flow to identify unknown network attacks. Threat intelligence, combined with big data analysis method of massive logs and traffic data, provides a more comprehensive security perspective.

However, the intrusion detection method based on single source data cannot reflect the relationship of seemingly irrelevant events. Besides, traditional security solutions (such as intrusion detection system, antivirus software, etc.) generate a lot of intrusion alerts. These alerts still need to be examined and cross-checked with other available data (such as host log and network communication data) in order to eliminate false positives and identify any legitimate attacks [5].

Illegal events affected by attack behavior are hidden in the dispersive log. Existing detection methods are insufficient in recognizing anomaly events to support sophisticated attacks detection. Targeted cyber-attacks detection based on heterogeneous multisource data has become a consensus. At present, the limitations of targeted cyber-attacks detection fall in three major categories as follows.

(1) Heterogeneous multisource security data is complicated and the semantics of expression is more abundant. There are various attack detection methods for different data sources. But the combinatorial relationship between research questions is not clear enough. There is a lack of systematic discussion in academic research.

(2) Existing methods are unable to quickly and efficiently locate anomalies related to network attacks. The problem of data correlation has not been well solved. Data association method in heterogeneous multisource still constrains the development of targeted cyber-attacks detection.

(3) The intelligence and automatization of existing detection technology are not satisfactory. Semantic information about secure data is not effectively expressed. Manual analysis is still the main way in attack recognition.

In order to address the limitations discussed above, we propose a novel layered framework for targeted cyber-attacks detection based on the integration of heterogeneous multisource data. The proposed framework utilizes attack investigation through correlation analysis, which can efficiently cope with targeted cyber-attacks detection in a big data environment. Based on this framework, inner-layer and cross-layer analysis approach for targeted cyber-attacks is proposed. Follow-up researchers can carry out further research on this basis.

This paper concentrates on the issue of the targeted cyber-attacks detection from the practical perspective. We propose a novel framework that uses big data correlation analysis techniques to analyze cyber incidents for enterprise network in order to improve attack detection ability. The proposed framework is named as *HeteMSD: A Big Data Analytics Framework for Targeted Cyber-Attacks Detection* that uses heterogeneous multisource data. We aim to develop an efficient framework that can assist security analyzers to reduce the blindness of data analysis from heterogeneous data sources without reducing the level of digital security guarantees.

Currently, both of experimental and theoretical works for targeted cyber-attacks detection have been in a groping stage, and there are still some significant discrepancies among the research of targeted cyber-attacks detection based on heterogeneous multisource data. Our proposed framework has a great significance as it intends to solve the conflicts between advanced network attack protection and big heterogeneous data burdens. The main contributions of our work are as follows.

(1) This paper comprehensively summarizes the existing data sources and points out the existing problems from the view of expression difference. Then we summarize and classify the multisource heterogeneous security detection data from three aspects: nonsemantic data, semantic data, and security knowledge data. Thus, we can have a more intuitive understanding of heterogeneous multisource security data.

(2) This paper proposed a novel framework designed for eliminating data redundancy while enhancing data relevance. We divide the research problem into five layers: sensing layer, event layer, alert layer, context layer, and scenario layer. We've made classification and integration of data sources and provide a dataflow diagram of the data process.

(3) This paper combines cyber-attacks analysis methods with big data correlation techniques to improve security levels and realize the comprehensive network security situation awareness. The research issues in related fields were investigated from perspectives at different levels.

The remainder of this paper is organized as follows. Section 2 gives a definition of targeted cyber-attacks and analyses existing targeted cyber-attacks detection technologies. Section 3 summarizes heterogeneous data sources and gives a novel classification perspective. Then we present the proposed framework based on the integration of heterogeneous multisource data and explain the brief dataflow in practical application. Section 4 discusses the application of correlation analysis for targeted cyber-attacks detection. Finally, the last section concludes the paper and describes the direction of future work.

2. Related Work

Targeted cyber-attacks are a class of dedicated attacks that aim at a specific user, company, or organization to realize specific intention, such as stealing sensitive data from a back-end database or paralysis system service. Targeted cyber-attacks have a characteristic of discrimination and are not random in nature. It means attackers involved in targeted attacks differentiate the targets and wait for the appropriate opportunity to realize the attack plan. Targeted cyber-attacks usually require several stages to achieve the goal. The lifestyle of a successful implemented targeted cyber-attack process usually includes gathering, infecting targets, system exploitation, data exfiltration, and maintaining control. [6]. Each of these steps is the key factor of targeted cyber-attacks. To successfully implement targeted cyber-attacks, all the above stages must be successful.

The difference between the targeted cyber-attack and a traditional attack is that targeted cyber-attacks are more complex and usually with strong intrusion motivation. Attackers spend more time choosing the target, find vulnerabilities, and customize malware. Targeted cyber-attacks are usually implemented by professionals instead of simply using attack tools.

The example attack scenario described in Figure 1 is as follows.

(i) Step 1: Attacker utilizes social engineering tools such as phishing mail to bypass firewall and infiltrates application server (Server1).

(ii) Step 2: Attacker takes Server1 as a springboard to identify and penetrate hosts in the internal network.

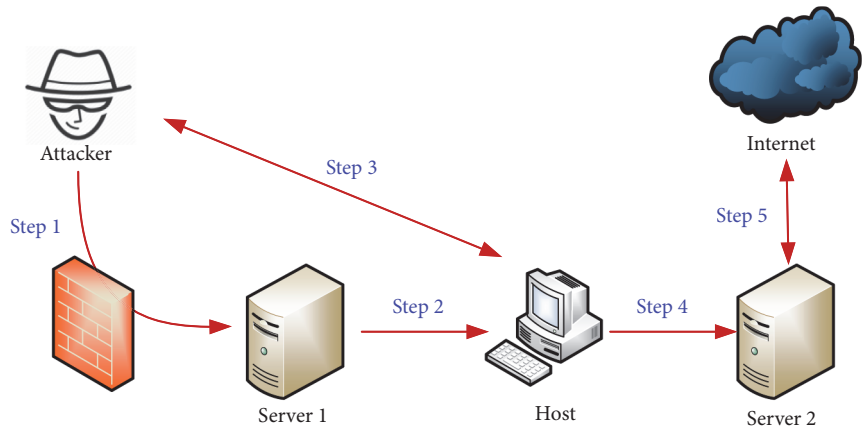


Figure 1: An example of targeted cyber-attack.

(iii) Step 3: Attacker establish C&C channel between the Attacker and host to control the host of the intranet to make further intranet penetration.

(iv) Step 4: Attacker uses the controlled host to log into the internal data server Server2 to collect sensitive data.

(v) Step 5: Send the collected sensitive data to the Internet to achieve the purpose of data theft.

When talking about targeted cyber-attacks, another concept Advanced Persistent Threat must be involved. APT can be seen as a subset of targeted cyber-attack [7]. Commonly speaking, APTs are targeted cyber-attacks with higher-level attack means. APT is implemented through a variety of different attack paths. It exists for a long time without being discovered in a real network environment. Generally, when talking about advanced complex and targeted network attacks, targeted cyber-attacks and APTs are interchangeable. There are few differences between targeted cyber-attacks and APTs [6].

The implementation process of targeted cyber-attacks can be described by the Kill Chain model [8]. As shown in Figure 2, the Kill Chain model summarizes the attack process as target reconnaissance, weapon customization, delivery, exploitation, installation, C&C channel establishment, action implementation, and other attack steps. Kill Chain attack model is as follows.

Targeted cyber-attacks have brought enormous threat to enterprise network security. In recent years, targeted cyber-attacks detection represented by APTs detection has attracted great attention from academic researchers and security industry. A variety of detection schemes have been proposed. Generally, it can be divided into host-based detection and network-based detection. At present, more detection methods are combined with big data analysis method and machine learning technology. Researchers also proposed correlation analysis method of audit log and network data based on threat information [9].

Host-based detection method: the representatives are antivirus software and HIDS. The main idea of malware detection is to detect malicious programs through a monitoring system call, network access, file operation, process creation, and memory modification. Through static analysis and dynamic analysis, malicious programs can be detected and APT attacks can be prevented [10]. The HIDS method based on pattern matching can effectively identify known attacks. But unknown attacks cannot be detected and the rule base needs to be updated regularly. Then behavior-based detection method is proposed. In recent years, with the rise of data mining and machine learning technology, researchers [11] have proposed a variety of detection technologies based on abnormal behavior recognition.

Network-based detection method: the pattern of command and control channel by malware has certain regularity (such as attack payload signature, sequential characteristics of network communications, and the generated domain name). Network traffic generated by targeted cyber-attack is different from those in normal business environments. Therefore, it is feasible to find the attack load of attack process by the network detection method.

Due to increase in number of sophisticated threats and the great increase in the volume of security data, the landscape of analyzing heterogeneous security data has drastically changed, as now working with security data has entered in the category of Big Data problem [4].

Parth Bhatt [12] proposed a research framework to handle complex attacks and it was tested with simulated attack data. The central basis of the framework consists of an Intrusion Management System and a multistage attack model. The multistage attack model is used to identify prevention and detection controls that provide logs used by the Intrusion Management System, and it is also used as a guide to logs correlation activities.

Mirco Marchetti [13] designed and evaluated a novel framework that is tailored to support security analysts in detecting APTs. The proposed framework uses multifactor approaches where big data analytics methods are applied to internal and external information to support human specialists so that those specialists can focus their security and intelligence analyses on the subset of hosts that are most likely to have been compromised. The proposed approach represents a step forward with respect to the state of the art

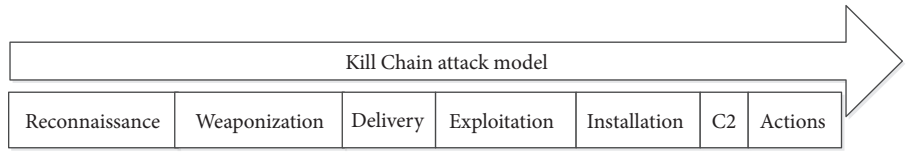


Figure 2: Kill chain attack model.

and paves the way to novel methods for early detection and mitigation of APTs

At present, traditional detection schemes focus on one or more stages. They cannot achieve comprehensive attack detection and have a high false negative rate and false positive rate. Although there have been a lot of research and great progress is made, targeted cyber-attacks are still occurring constantly. The shortcoming is that existing method still depends on manual analysis. It cannot identify and respond quickly. Here we want to reduce human participation and make the detection process as intelligent as possible. Similar attacks should be detected according to the attack pattern. Data parsing should be as automatic as possible and be detected in an adaptive way. The intelligence of attack detection is mainly embodied in several research points.

(1) Accuracy. A large number of false alerts pose a big challenge in intrusion detection. Even if the false alarm rate is merely 1%, a large amount of data can bring a lot of alerts, which will bring enormous burdens to security managers. Reducing false positive and false negative is an important content in intrusion detection.

(2) Efficiency. Fast identification of attack behavior is also important in practical application. Shortening attack detection time will minimize the damage caused by the attack. Thus it can reduce security risk for the system and handle security accidents in time.

(3) Intelligence. Automated attack inference process. Reducing manual labor costs on irrelevant information can help human analysts put artificial judgment in the key part.

The ability to cross-correlate events or alerts from various sources in the network is key for detecting sophisticated multistage attacks at an early stage, which would allow a reasonable time to stop the attack or mitigate the damage.

In summary, existing targeted cyber-attacks detection technologies for network security threats recognition are not fully adapted. These methods cannot meet the needs for network security situational awareness. Current utilization of multisource heterogeneous data is inadequate. The purpose of this paper is to better correlate security data from different sources and improve situational awareness ability. The related research issues and the next research directions are discussed in the following content.

3. Concepts and the Proposed Framework

3.1. Heterogeneous Multisource Data. Data is the premise of attack detection and threat analysis. However, security data come from diverse sources, such as IDS alerts, firewall alerts, NetFlow data, Linux syslog, Event Tracing for Windows, etc. Heterogeneous data formats from various sources have diverse semantic meanings. It is difficult to identify the relationship in practical application. Firstly, we investigate and summarize the heterogeneous multisource security data from threat hunting aspect. Security data has the following characteristics.

(1) Heterogeneous Format. Comprehensive security data come from local hosts, servers, routers, firewalls, IDSs, and other security devices. Data acquisition and storage from different sources are quite different. Data formats include structured, semistructured, and unstructured.

(2) Diverse Semantic. Different types of data represent different levels of security knowledge. For example, network data and host logs represent information in different fields. IDS alerts, firewall logs, and other security logs represent information at different semantic levels from host logs and network data. Different security data calls for different treatment in practical application.

(3) Correlation across Data Sources. A complex targeted cyber-attack hides its behavior in multisource heterogeneous data. A single attack will bring records of data from multiple sources. These data can be linked together with the hosts, users, location, etc. as the associated elements. And all these data as a whole can be restored to a complete attack scenario.

It is an important research issue to classify security data from various sources. This paper summarizes and analyses the multisource heterogeneous security data in targeted cyber-attack detection. Firstly, we classify them into three categories according to the different semantic levels of security data expression.

(1) Nonsemantic Data. Nonsemantic data includes detailed descriptions of the running process and logs reflecting the attacks, but it does not contain security semantic information. Nonsemantic data includes operation logs, system calls, NetFlow data, user behavior logs, etc. Analysts can analyze malicious behavior from their data and generate semantic data.

(2) Semantic Data. Based on pattern matching and other technologies, we can get security alerts from the nonsemantic data. This kind of data has security semantics information,

which indicates the violation of security rules or the abnormal operation of the system such as IDS alerts, firewall logs, OS security logs, and so on. There are many different security systems deployed in the current network environment, including firewall, antivirus software, intrusion detection system, and traffic analysis system. The key technologies corresponding to these security systems have made great progress in the past decades. But these independent systems can only solve local problems but cannot reflect the overall situation of network security. The limitations of traditional security systems are becoming more and more obvious.

(3) Security Knowledge Data. At present, most network attacks rely on specific vulnerabilities and services, such as operating system vulnerabilities, software vulnerabilities, or protocol vulnerabilities. If vulnerability information is integrated into the process of attack detection, the accuracy of attack detection can be improved. Besides, with the development of security technology, threat intelligence technology has become an important data source for detection [14]. Security knowledge data includes vulnerability database, attack pattern database, and another external threat intelligence.

3.2. Framework Design. In the previous section, we analyze and summarize three kinds of security data. But how to carry out targeted cyber-attack detection based on multisource heterogeneous data? In this section, we present a novel framework for targeted cyber-attacks detection after an in-depth study of targeted cyber-attack detection technology, According to the different analytical perspective, we divide the detection into five levels: sensing layer, event layer, alert layer, context layer, and scenario layer. Figure 3 shows a high-level architecture of our framework.

(1) Sensing Layer: sensing layer includes the initial data source of targeted cyber-attack detection. In this layer, heterogeneous multisource security data acquisition and data sensing is an important basic research issue. Summarizing all kinds of data sources comprehensively and systematically and collecting and transmitting security data efficiently from many collectors is still a challenge to the current security research. In addition, as an important source of perceived data, how to quickly and timely response to new security threats combined with security knowledge is also a problem that needs to be considered. The main research area in the sensing layer includes data acquisition, data aggregation, data parsing and preprocessing, data fusion, feature extraction, and feature selection.

(2) Event Layer: event layer handles with nonsemantic data. As mentioned above, the nonsemantic data of network security are various and the attributes are quite different (for example, time sequential data, spatial data, trajectory data, and NetFlow data). How to extract specific features from these data and express large-scale heterogeneous data will be a big challenge. Especially in targeted cyber-attack detection, only by establishing the relationship between different data can the later analysis process be conducted efficiently. In this layer, the main research area includes event aggregation, content security, event correlation, anomaly detection, etc.

(3) Alert Layer: data input of alert layer includes anomaly detection results and semantic data. The semantic data include alerts coming from event layer data after analysis and detection and alerts generated by firewall, intrusion detection system, antivirus software, and other security devices. The difference from the event layer is that the alert layer contains all kinds of alert data with security semantics, such as exception score, alarm level, etc. In this layer, the main research area includes alert expression, alert clustering, alert fusion, security information, and event management.

(4) Context Layer: due to the wide range of alert data sources, the result of alert correlation is only attack fragments or preliminary attack steps. How to obtain reinforcing knowledge from security data is still a challenging problem. Traditional machine learning technologies focus on a single data source rather than heterogeneous multisource data. It is still a challenging problem in targeted cyber-attack detection to extract attack context and attack fragments from security data of different sources. Human participation is an important factor in this layer. Reducing manual analysis is a problem to be solved at this level. In this layer, the main research area includes attack modelling, attack pattern mining, attack scenario reconstruction, attack inference, etc.

(5) Scenario Layer: In this layer, the attack scenario is reconstructed based on attack modelling and attack scenario reconstruction. Attack scenarios are expressed in a way consistent with human cognition. In this layer, the main research area includes threat intelligence expression, attack causality analysis, visual analysis, and attack planning.

Other existing research issues can be introduced into our framework. The research contents of each layer and across layers are analyzed respectively. Current research issues in this field are shown in Figure 4.

Discovering targeted cyber-attacks from heterogeneous multisource data is a systematic analysis process. The problem is more complicated because of the diversity of security data. As data continues to be added, data processing will provide a deeper understanding of attack scenarios. As shown in Figure 4, there are data dependencies across layers. The lower layer provides input data for the upper layer, while the upper layer analyses the input data for further analysis. More specifically, the data flow process is shown in Section 3.3

3.3. Dataflow Diagram. In the current research, intrusion detection technology needs further refinement and modification. The key issue is to analyze the relationship between different levels of security data. However, even if the detection framework is given, it is difficult to find the behavior traces of attacks from heterogeneous data. This paper presents a general attack detection method. The dataflow diagram discussed in Figure 5 gives the description of the process

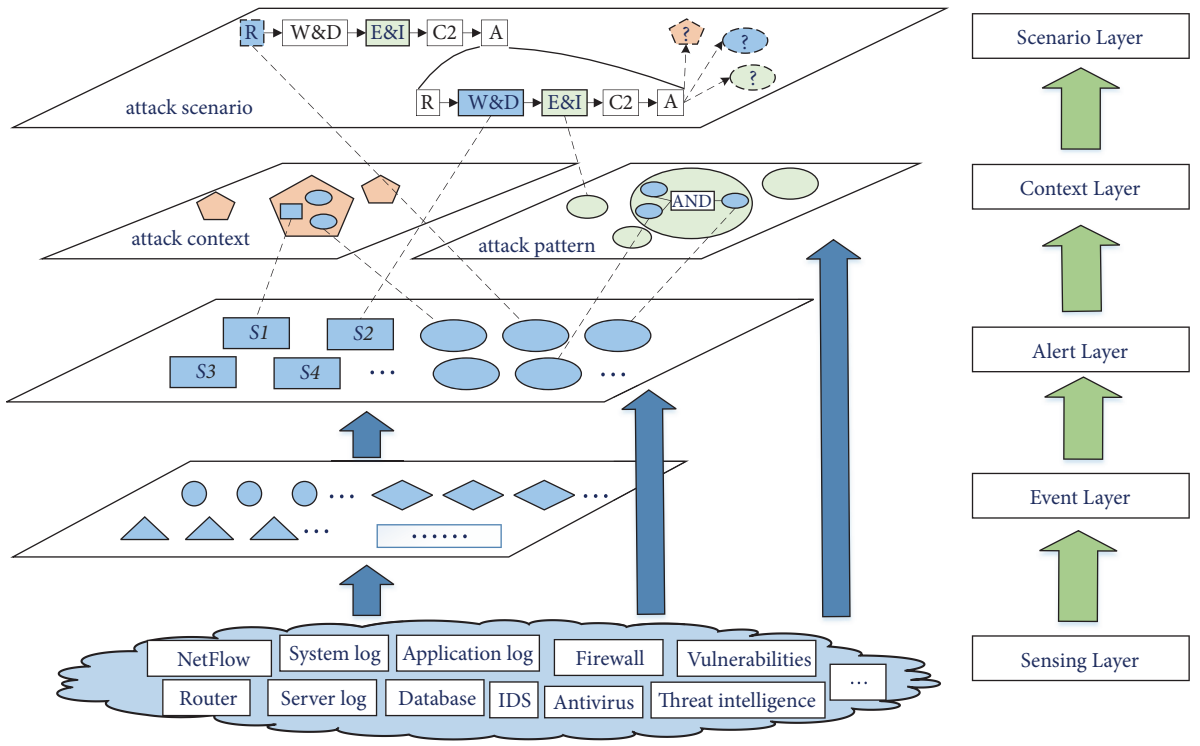


Figure 3: HeteMSD: a big data analytics framework for targeted cyber-attacks detection.

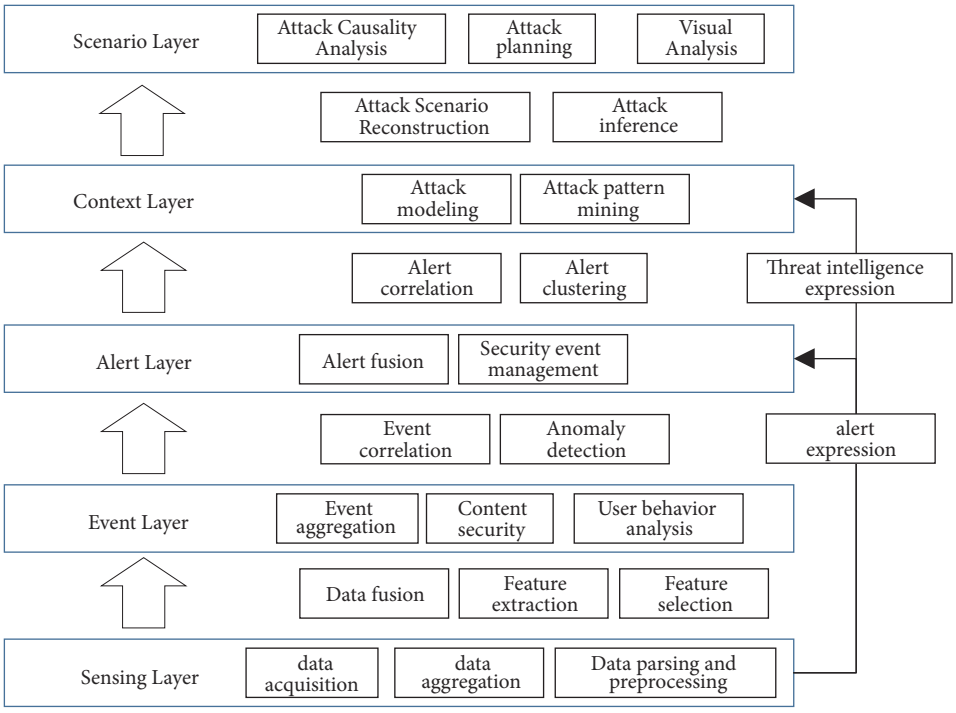


Figure 4: Research issues in this field.

from the original security data to the generated attack scenario.

Three kinds of data summarized in Section 3 are inputs for different stages in the dataflow diagram. The initial input data is nonsemantic data. Raw data is preprocessed to generate security event with security semantics. Preprocessed security events, combined with semantic data, are the input of the data analysis module. After data fusion analysis, these data are analyzed and generate security alerts with higher confidence. Furthermore, security data supplemented by

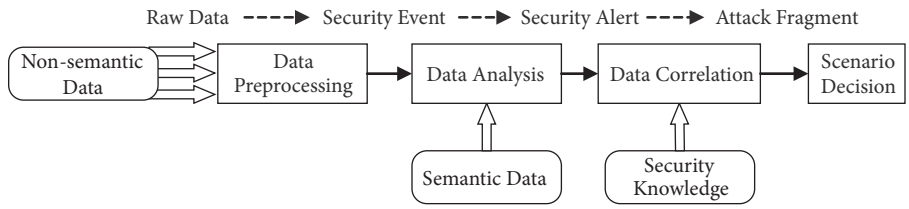


Figure 5: Dataflow diagram.

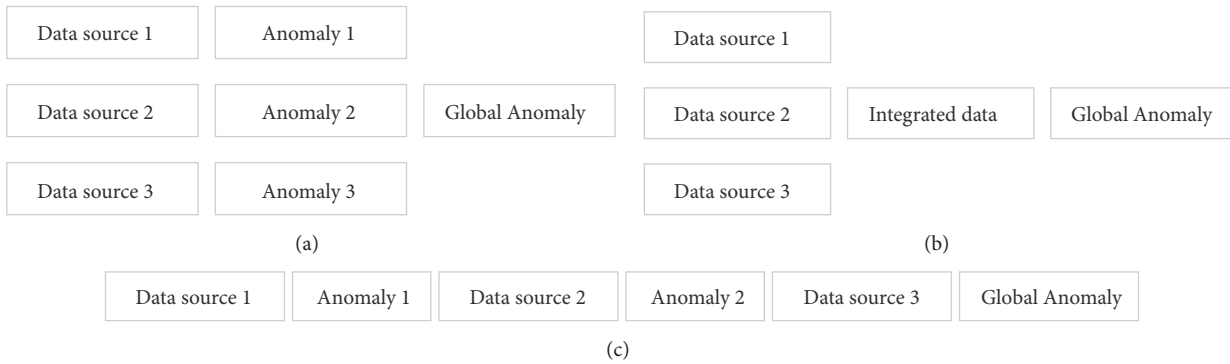


Figure 6: Anomaly detection technology based on data fusion.

security knowledge will generate attack fragment with more cognitive information. Finally, these fragments are filled into the attack inference model to form a more complete attack scenario. Attack scenario reconstruction is implemented with attack correlation and attack investigation. Specifically, the data analysis process is shown in Figure 5.

The main functions of each module in Figure 5 are as follows.

Data preprocessing module: the input of data preprocessing module is nonsemantic data. It mainly processes raw data into security events for subsequent analysis, where security events are with certain security semantics. This module is the key to find abnormal traces of targeted cyber-attacks. The main methods used in this module are feature extraction, feature selection, and anomaly detection.

Data analysis module: the function of this module is extracting security alerts with security semantic data from basic security incidents such as low-level alarms and anomalies. And that can eliminate false alarms of alarms produced by security devices to a large extent. By combining analysis of alarms and anomaly detection results generated by security equipment, a more reliable result is obtained. Ranking algorithms are commonly used in this module.

Data correlation module: this module generates fragmented information that expresses attack patterns based on multiple correlation analysis methods. That can establish the relationship between different levels of data and enhance security situation awareness. This module is designed to aggregate alerts and security knowledge representing human cognition. Attack scenarios will eventually be formed through knowledge extracting from the original data.

In summary, we introduced the framework and explain the flow of data processing in the application. The following section will give the main correlation analysis methods used in this framework.

4. Correlation Analysis

The key to targeted cyber-attack detection lies in how to integrate and establish the correlation between multisource heterogeneous security data. In the previous section we have discussed the framework design. Furthermore, there is a need to correlate security measures with attack phases for better understanding. In this section technical aspects of correlation analysis and defence measures are discussed, more concretely, mainly in the following aspects.

4.1. Event-Event Correlation. The correlation between event and event can be achieved by multisource data fusion. For raw input data, abnormal detection results represent event correlation. Data fusion can reduce data redundancy and collaborative information acquisition through complementarity [15]. The features of original heterogeneous data need to be extracted as the formats of the original data are inconsistent. The global abnormal results are obtained by utilizing the comprehensiveness association of multisource data. At present, the main method of heterogeneous event fusion is to extract features from different security data and form a global feature vector. The relationship between different sources of data can better reflect the global anomaly that cannot be expressed by a single data source.

The anomaly detection method of multisource heterogeneous data is different from that of single source data. Existing multisource heterogeneous data anomaly detection technologies based on data fusion can be divided into three categories (as shown in Figure 6). First, anomaly detection

algorithms are applied to different data sources, and the global anomaly is obtained after aggregation analysis of anomaly detection results. Second, different data sets are merged into a unified data source with the same data pattern. In this way, the multisource data anomaly detection is transformed into the traditional single-source data anomaly detection problem. Thirdly, more data sources are added in the process of anomaly detection to supplement and strengthen the anomaly detection results.

In this field, extracting feature vectors from multisource heterogeneous data is a key factor. And anomaly detection of multisource heterogeneous data based on ensemble learning needs further research.

4.2. Alert-Alert Correlation. An attack behavior may trigger many different alert events, so it is necessary to associate the original security events to higher-level alerts. Alert correlation is an important technology to cope with large number of alerts generated by intrusion detection systems. And it has become a new research trend in the field of attack detection. In current attack analysis methods, manual analysis is still a key factor in identifying attack scenarios. The automatic method is to establish a link between alerts to simplify the analysis of security experts. At present, there are mainly three kinds of alerts correlation analysis methods.

(1) Similarity-based method. Alerts are clustering by calculating the similarity of attribute values between alerts.

(2) Condition-based method. Alerts are correlated based on the precondition and postresult of an attack. The attack correlation is established by matching the premise and result of different attack types.

(3) Scenario-based method. The attack scenarios are established by matching the alarm with the known attack pattern.

Through alert correlation, redundant information of alert events can be deleted. On the other hand, the higher-level semantic information of alarm events can be extracted and improve the accuracy of attack final detection results. It reduces the large number of alerts generated by intrusion detection system and can help analysts to better grasp and analyze the attacker's motivation. Establishing alarm correlation model by adjusting probability inference results with optimization algorithm needs further study.

4.3. Pattern-Knowledge Correlation. In recent years, representation learning technology has gained wide attention in the fields of speech recognition, image analysis, and natural language processing. Representation learning aims at representing the semantic information of the object as a dense low-dimensional real-valued vector. In this low-dimensional vector space, the closer the two vectors are, the higher their semantic similarity is. Knowledge representation learning is a representation learning method oriented to entities and relationships in the knowledge base. Recently, a series of important advances have been made in this field [16]. It can efficiently calculate the semantic relationship between entities and relationships in low-dimensional space. This method can effectively solve the problem of data sparsity and significantly improve the performance of knowledge reasoning.

The vectors learned by the representation learning algorithm can find the similarity between the descriptive text data by calculating the distance or tag the document further. They are usually used in the field of emotional analysis. In the field of cyber security, the result of the representation learning algorithm is used to express the similarity between attack pattern and vulnerabilities. By matching the evaluation results with the attack context detection results, we can find the closest attack pattern and reduce the cost of manual analysis.

4.4. Alert-Context Correlation. After discovering attack steps or fragments by correlation analysis, the reconstruction of attack scenario often relies on manual analysis of security experts. However, manual analysis cannot cope with the change of targeted cyber-attacks and the dramatic increase of big security data. Therefore, it is necessary to remove irrelevant factors from fragmented attack data by attack modeling method. Correlating alerts and attack patterns can simplify the judgment process by using the intelligent method. That will extract more cognitive attack scenarios from the results and leave them to the analysts to judge, so as to improve the detection efficiency and shorten the attack detection time.

Attack modeling is an important technology to analyze attacks in cyberspace and to guide the detection and investigation of targeted cyber-attacks. Attack modeling can further help security analysts in dealing with targeted cyber-attacks. Researchers have proposed a variety of targeted cyber-attack analysis models, such as attack graph, attack chain, diamond model, pyramid model, and so on. Targeted cyber-attack scenarios can be reconstructed based on cognitive analysis model.

5. Conclusions and Perspectives

This paper discusses cyber-attacks detection from an attacker's perspective to help security professionals to have in-depth understanding of targeted cyber-attack. In this paper, we proposed a novel framework, HeteMSD, in order to enhance data relevance using heterogeneous multisource data. The proposed research work was a novel attempt in targeted cyber-attacks detection field and provided the theoretical fundamental for future research. HeteMSD is explained with technical components and corresponding methodologies. Moreover, correlation analysis against attack investigation is discussed with effective implementation using existing solutions. Various techniques for mitigation, prevention, and detection at different layers are also discussed in detail to help the defender in implementing effective defenses.

This work represents a first step in the definition of a comprehensive framework for the investigation of targeted cyber-attacks. HeteMSD still needs to be complemented with more features for the integration of the human expert, who, beyond being a simple observer, also has the knowledge required to enrich the preliminary analyses proposed by the framework. Further work must be done to anomaly detection based on multisource data fusion. More research will be done

on the extension of proposed correlation method. Lastly, security knowledge reasoning is likely to become a major topic of interest for security investigation in the near future.

Data Availability

The relevant data related to this paper is in https://github.com/kbandla/APTnotes.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Authors thank the support from The National Natural Science Foundation of China no. 61501615 and no. 61602515. The paper is also supported from the Foundation of Science and Technology on Information Assurance Laboratory (no:. 614211203010417).

References

[1] Y. Li, W. Dai, J. Bai, X. Gan, J. Wang, and X. Wang, "An intelligence-driven security-aware defense mechanism for advanced persistent threats," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 646–661, 2019.

[2] J. Navarro, A. Deruyver, and P. Parrend, "A systematic survey on multi-step attack detection," *Computers & Security*, vol. 76, pp. 214–249, 2018.

[3] Y. Liu, M. Zhang, D. Li et al., "Towards a timely causality analysis for enterprise security," in *Proceedings of the Network and Distributed System Security Symposium*, 2018.

[4] R. Zuech, T. M. Khoshgoftaar, and R. Wald, "Intrusion detection and Big Heterogeneous Data: a Survey," *Journal of Big Data*, vol. 2, no. 1, pp. 1–41, 2015.

[5] I. Herwono and F. A. El-Moussa, "A system for detecting targeted cyber-attacks using attack patterns," in *Proceedings of the International Conference on Information Systems Security and Privacy*, Communications in Computer and Information Science, pp. 20–34, Springer, 2017.

[6] A. Sood and R. Enbody, *Targeted Cyber Attacks: Multi-staged Attacks Driven by Exploits and Malware*, Syngress, 2014.

[7] A. K. Sood and R. J. Enbody, "Targeted cyberattacks: a superset of advanced persistent threats," *IEEE Security & Privacy*, vol. 11, no. 1, pp. 54–61, 2013.

[8] M. S. Khan, S. Siddiqui, and K. Ferens, "A cognitive and concurrent cyber kill chain model," *Computer and Network Security Essentials*, pp. 585–602, 2017.

[9] L. Qiang, Y. Zeming, L. Baoxu, J. Zhengwei, and Y. Jian, "Framework of cyber attack attribution based on threat intelligence," in *Proceedings of the International Conference on Interoperability in IoT*, pp. 92–103, Springer, 2016.

[10] P. Gao, X. Xiao, Z. Li et al., "AIQL: enabling efficient attack investigation from system monitoring data," *USENIX Security*, pp. 113–126, 2018.

[11] S. R. Snapp, J. Brentano, G. Dias et al., "DIDS (distributed intrusion detection system)-motivation," *Architecture, and an Early Prototype*, pp. 167–176, 2017.

[12] P. Bhatt, E. T. Yano, and P. Gustavsson, "Towards a framework to detect multi-stage advanced persistent threats attacks," in *Proceedings of the 8th IEEE International Symposium on Service Oriented System Engineering, SOSE 2014*, pp. 390–395, IEEE, UK, April 2014.

[13] M. Marchetti, A. Guido, F. Pierazzi, and M. Colajanni, "Countering Advanced Persistent Threats through security intelligence and big data analytics," in *Proceedings of the 8th International Conference on Cyber Conflict, CyCon 2016*, pp. 243–261, Estonia, June 2016.

[14] Z. Syed, A. Padia, T. Finin, M. L. Mathews, and A. Joshi, "UCO: a unified cybersecurity ontology," *AAAI Workshop: Artificial Intelligence for Cyber Security*, 2016.

[15] Y. Zheng, "Methodologies for cross-domain data fusion: an overview," *IEEE Transactions on Big Data*, vol. 1, no. 1, pp. 16–34, 2015.

[16] J. Navarro, V. Legrand, S. Lagraa et al., "HuMa: A multi-layer framework for threat analysis in a heterogeneous log environment," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 10723, pp. 144–159, 2018.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 4656480, 10 pages
https://doi.org/10.1155/2019/4656480

Research Article

Optimizing Computer Worm Detection Using Ensembles

Nelson Ochieng,[1] **Waweru Mwangi**,[2] **and Ismail Ateya**[1]

[1]*Strathmore University, Kenya*
[2]*Jomo Kenyatta University of Agriculture and Technology, Kenya*

Correspondence should be addressed to Nelson Ochieng; nochieng@strathmore.edu

Received 13 December 2018; Revised 26 February 2019; Accepted 3 March 2019; Published 11 April 2019

Guest Editor: Pelin Angin

The scope of this research is computer worm detection. Computer worm has been defined as a process that can cause a possibly evolved copy of it to execute on a remote computer. It does not require human intervention to propagate neither does it attach itself to an existing computer file. It spreads very rapidly. Modern computer worm authors obfuscate the code to make it difficult to detect the computer worm. This research proposes to use machine learning methodology for the detection of computer worms. More specifically, ensembles are used. The research deviates from existing detection approaches by using dark space network traffic attributed to an actual worm attack to train and validate the machine learning algorithms. It is also obtained that the various ensembles perform comparatively well. Each of them is therefore a candidate for the final model. The algorithms also perform just as well as similar studies reported in the literature.

1. Introduction

Malware includes computer virus, Trojan horse, spyware, ad-ware, computer worms among many others. In survey by [1], a malware event occurs in organizations every 3 minutes and attacks many sectors with alarming losses to intellectual property, compromised customer records and even destruction of data. This research has as its scope computer worm detection in a network. Reference [2] defines a computer worm as a "process that can cause a (possibly evolved) copy of it to execute on a remote computational machine". Worms self-propagate across computer networks by exploiting security or policy flaws in widely used network services. Unlike computer viruses, computer worms do not require user intervention to propagate nor do they piggy-back on existing files. Their spread is very rapid [3, 4] with the ability to infect as many as 359,000 computers in under 14 hours, or even faster. Computer worms therefore present unique challenges to security researchers hence motivating this study.

Defense against computer worm attacks may be through prevention of worm attacks, detection of worms, containment of worm spread and removal of worm infections. Prevention is not always wholly possible because of the inherent vulnerabilities found in all software. Detection is therefore the better approach.

A number of computer worm detection approaches have been explored in the research environment. Content-based fingerprinting captures a worm's characteristics by deriving the most representative content-sequence as the worm's signature. Anomaly-detection leverages the fact that worms are likely to exhibit anomalous behavior such as port-scanning and failed connection attempts, which are distinct from normal behavior. Behavioral foot-printing makes use of the fact that each worm exhibits a definite communication pattern as it propagates between hosts in a network and these patterns can be used to uniquely identify a worm. Intelligent detection approaches that use machine learning have also been proposed. While each of these approaches has its strengths, a number of weaknesses have also been noted. For example, content-signature schemes, while an established dimension to detect worms, fail to detect novel worms and are expensive on the system. In anomaly detection, profiling normal network behavior is impossible and establishing detection threshold is also difficult. Behavioral foot-printing is prone to behavior camouflaging attacks. Approaches that leverage machine learning have generated high false positive

and false negative rates. This has been partly because of poor characterization of worm traffic and also because of the lack of sound datasets for training and validation of the algorithms.

This paper presents an approach that attempts to provide better performance. The feature set used for the machine learning algorithms are selected network packet header fields as reported in an earlier paper by the authors [5]. The rest of the paper is organized as follows. Section 2 reviews existing literature on computer worm detection using machine learning. Section 3 discusses the methodology for the research. Section 4 discusses the results. The paper concludes with a summary in Section 5.

2. Related Work

A number of approaches for computer worm detection have been reviewed in the literature. These include content-based signature schemes, anomaly-detection schemes, and behavioral-signature detection schemes, a summary and analysis of which has been presented by the authors in an earlier paper [6]. For this present work, only approaches that utilize machine learning are emphasized. Reference [7] was one of the seminal works in using machine learning techniques for malware detection. It used static program binary properties and achieved a detection rate of 97.76%. Reference [8] used n-grams extracted from the executable to form training examples. They apply several learning methods such as Nearest Neighbors, Naïve Bayes, Support Vector Machines, Decision Trees, and Boosting. Boosted Decision Trees performed the best with an Area under Curve (AUC) of 0.996. Win32 Portable Executables (PE) as a feature is used by [9–11]. Paper [12] achieves a True Positive Rate of 98.5% and a False Positive Rate of 0.025 using Windows Application Programming Interface (API) calls as the features. Other feature types used include Operation Code (OPcode) [13], sequence of instructions that capture program control flow information [14], and binary images [15]. Reference [15] obtains an accuracy of 98%. Paper [16] uses a restricted Boltzmann machine, a neural network, to create a new set of features from existing ones. These are then used to train a one-side perceptron (OSP) algorithm. The work gets very close to obtaining a zero false positive classifier. Paper [17] uses Logistic Model Trees, Naïve Bayes, Support Vector Machines (SVM), and k Nearest Neighbors (kNN) and obtains an accuracy of 98.3% with the Linear Model Tree algorithm. A deep neural network that uses byte entropy histogram, PE import features and PE metadata features is deployed by [17] and achieves a detection rate of 95% and a false positive rate of 0.1%. Reference [18] also uses deep learning.

Most of the reviewed works utilize a single parameter for the detection. The present work will utilize many features as reported by the authors in [5].

3. Methods

The main aim of this work is to investigate various machine learning ensembles on computer worm detection using unidirectional network traffic to a dark space. The methodology adopted follows the standard procedure in machine learning: (1) collecting data, (2) exploring and preparing the data, and (3) training a model on the data and evaluating model performance.

3.1. Dataset. The datasets used for the experiments were obtained from the University San Diego California Center for Applied Data Analysis (USCD CAIDA). The center operates a network telescope that consists of a globally rooted /8 network that monitors large segments of lightly used address space. There is little legitimate traffic in this address space; hence, it provides a monitoring point for anomalous traffic that represents almost 1/256th of all IPv4 destination addresses on the Internet.

Two sets of datasets were requested and obtained from this telescope. The first is the Three Days of Conficker Datasets [19] containing data for 3 days between November 2008 and January 2009 during which Conficker worm attack [20] was active. This dataset contains 68 compressed packet capture (pcap) files each containing one hour of traces. The pcap files only contain packet headers with the payload having been removed to preserve privacy. The destination IP addresses have also been masked for the same reason. The other dataset is the Two Days in November 2008 dataset [21] with traces for the 12th and 19th November 2008, containing two typical days of background radiation just prior to the detection of Conficker which has been used to differentiate between Conficker-infected traffic and clean traffic.

The datasets were processed using the CAIDA Corsaro software suite [22], a software suite for performing large-scale analysis of trace data. The raw pcap datasets were aggregated into the FlowTuple format. This format retains only selected fields from captured packets instead of the whole packet, enabling a more efficient data storage, processing and analysis. The 8 fields are source IP address, destination IP address, source port, destination port, protocol, Time to Live, TCP flags, and IP packet length. An additional field, value, indicates the number of packets in the interval whose header fields match this FlowTuple key.

The instances in the Three Days of Conficker dataset have been further filtered to retain only instances that have a high likelihood of being attributable to Conficker worm attack of the year 2008. Reference [20] focuses on Conficker's TCP scanning behavior (searching for victims to exploit) and indicates that it engages in three types of observable network scanning via TCP port 445 or 139 (where the vulnerable Microsoft software Windows Server service runs) for additional victims. The vulnerability allowed attackers to execute arbitrary code via a crafted RPC request that triggers a buffer overflow. These include local network scanning where Conficker determines the broadcast domain from network interface settings, scans hosts nearby other infected hosts and random scanning. Other distinguishing characteristics include TTL within reasonable distance from Windows default TTL of 128, incremental source port in the Windows default range of 1024-5000, 2 or 1 TCP SYN packets per connection attempt instead of the usual 3 TCP SYN packets per connection attempt due to TCP's retransmit behavior.

This dataset solves the privacy challenge by removing the payload and also masking out the first octet of the destination IP address. It is also a more recent dataset than the KDD dataset that has been the one available for network security researchers. However, it only includes unidirectional traffic to the network telescope and therefore does not allow the researchers to include features of computer worms that would be available in bidirectional traffic and would deliver a more complete training for the classifiers.

3.2. Features. This section presents an analysis of the features to be used for detection and their contribution towards the detection capability of the learning algorithms. These features were obtained after performing feature selection experiments whose results were reported in [5]. The best features for the classification task were there identified as Time to Live (TTL), Internet Protocol (IP) packet length, value or number of packets in the packet capture interval whose header fields match the Flow Tuple key, well known destination ports or destination ports within the range 0-1024, and IP packet source country China. The above features are IP packet header fields. TTL is used to avoid looping in the network. Every packet is sent with some TTL value set, which tells the network how many network routers (hops) this packet can cross. At each hop, its value is decremented by one and when the value reaches zero, the packet is discarded. Different operating systems have default TTL ranges and since computer worms target vulnerabilities in particular operating systems, they will usually be associated with TTL within certain ranges. For example, Conficker worm packets have TTL within reasonable distance from Windows default TTL of 128. Packet length indicates the size of the packet. Particular computer worms are associated with particular packet length sizes. For example, the packet length for Conficker worm is around 62 bytes. The value feature referred to the number of packets with a unique packet header signature sequence. A number of flow tuples with a particular key would be suspicious. China originates most of the malicious software packets. Computer worms target well known ports where popular services run for maximum impact. Conficker worm, for example, targets port 445 or 139.

3.3. Ensembles. Various machine learning ensembles were explored and their detection capabilities investigated. Ensemble methods try to construct a set of learners and combine them. The ensemble methods investigated included averaging technique, GradientBoostingClassifier, AdaBoost, Bagging, Voting, Stacking, Random Forests, and ExtraTreesClassifier. The base classifiers used included SVM, Multilayer perceptrons, kNN, NB, Logistic Regression, and Decision Trees. Python programming language was used for the classification experiments and more especially the Scikit-learn library [22]. These ensemble techniques are described as follows.

3.3.1. ExtraTreesClassifier. This builds an ensemble of unpruned decision trees according to the classical top-down procedure [23].

The Extra-Trees splitting procedure for numerical attributes is given in Algorithm 1. It has two parameters: K, the number of attributes randomly selected at each node and nmin, the minimum sample size for splitting a node. It is used several times with the (full) original learning sample to generate an ensemble model (we denote by M the number of trees of this ensemble). The predictions of the trees are aggregated to yield the final prediction, by majority vote in classification problems.

3.3.2. Random Forests. Paper [24] defines a random forest as a classifier consisting of a collection of tree-structured classifiers h(x, θk), k = 1,..., where the θk are independent, identically distributed random vectors and each tree casts a unit vote for the most popular class at input x. This is as shown in Algorithm 2.

3.3.3. AdaBoost. Reference [25] explains AdaBoost as taking as input a training set $(x_1, y_1)\ldots(x_m, y_m)$ where each x_i belongs to some domain or instance space X, and each label y_i is in some label set Y. AdaBoost calls a given weak or base learning algorithm repeatedly in a series of rounds t = 1...T. One of the main ideas of the algorithm is to maintain a distribution or set of weights over the training set. The weight of this distribution on training example i on round t is denoted Dt (i). Initially, all weights are set equally, but on each round the weights of incorrectly classified examples are increased so that the weak learner is forced to focus on the hard examples in the training set. AdaBoost is shown in Algorithm 3.

3.3.4. Bagging. The name Bagging came from the abbreviation of Bootstrap AGGregatING [26]. The two key ingredients of Bagging are bootstrap and aggregation. Bagging applies boostrap sampling to obtain the data subsets for training the base learners. Given a training data set containing m number of training examples, a sample of m training examples will be generated by sampling with replacement. Each of these datasets is used to train a model. The outputs of the models are combined by averaging (in regression) or voting (in classification) to create a single output. Algorithm 4 shows Bagging.

3.3.5. Gradient Boosting. Gradient Boosting [27] is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. Gradient Boosting is shown in Algorithm 5.

3.3.6. Voting. Voting is the most popular and fundamental combination method for nominal outputs. In majority voting, every classifier votes for one class label, and the final output class label is the one that receives more than half of the votes; if none of the class labels receives more than half of

Split a node(S)
Input: the local learning subset S corresponding to the node we want to split
Output: a split $[a < a_c]$ or nothing
(i) If **Stop split**(S) is TRUE then return nothing.
(ii) Otherwise select K attributes $\{a_1, \ldots, a_K\}$ among all non-constant (in S) candidate attributes;
(iii) Draw K splits $\{s_1, \ldots, s_K\}$, where s_i = **Pick a random split**(S, a_i), $\forall i = 1, \ldots, K$;
(iv) Return a split s^* such that Score(s^*, S) = $\max_{i=1,\ldots,K}$ Score(s_i, S).
Pick a random split(S,a)
Inputs: a subset S and an attribute a
Output: a split
(i) Let $a_{S_{\max}}$ and $a_{S_{\min}}$ denote the maximal and minimal value of a in S;
(ii) Draw a random cut-point a_c uniformly in $[a_{S_{\min}}, a_{S_{\max}}]$;
(iii) Return the split $[a < a_c]$.
Stop split(S)
Input: a subset S
Output: a boolean
(i) If $|S| < n_{\min}$, then return TRUE;
(ii) If all attributes are constant in S, then return TRUE;
(iii) If the output is constant in S, then return TRUE;
(iv) Otherwise, return FALSE.

Algorithm 1: Extra Trees Algorithm.

Input: Learning set- S, Ensemble size B, Proportion of attributes considered f
Output: Ensemble E
(1) E = φ
(2) **for** i = 1 to B **do**
(3) S^i=Boostrap Sample(S)
(4) C^i=Build Random Tree Model (S^i, f)
(5) E = E ∪ {C^i}
(6) **return** E

Algorithm 2: Random forest.

the votes, a rejection option will be given and the combined classifier makes no prediction.

3.3.7. Stacking. Reference [28] explains that Stacked Generalization is a method for combining heterogeneous base models, that is, models learned with different learning algorithms such as the nearest neighbor method, DTs, NB, among others. The base models are not combined with a fixed scheme such as voting, but rather an additional model called meta model is learned and used for combining base models. First, the meta learning dataset is generated using predictions of the base models and then, using the meta learning set, the meta model is learned which can combine predictions of base models into a final prediction.

Input: Learning set- S, Ensemble size B.
Output: Ensemble E
(1) E = φ
(2) W=Assign Equal Weights (S)
(3) **for** i = 1 to B **do**
(4) C^i=Construct-Models (S, W)
(5) E_{rr}=Apply Model (C^i, S)
(6) **if** (E_{rr} = 0) ∪ (E_{rr} ≥ 0.5) **then**
(7) Terminate Model Generation
(8) **return** E
(9) **for** j = 1 to Number Of Examples (S) **do**
(10) if Correctly Classified (S_j, C^i) **then**
(11) $W_j = W_j \; E_{rr} / 1 - E_{rr}$
(12) W = Normalize Weights W
(13) E = E ∪ {C^i}
(14) **return** E

Algorithm 3: AdaBoost.

3.4. Ensemble Experiments. Ensemble experiments started with a comparison of the base classifiers. The base classifiers performed as shown in Figure 1.

Multilayer perceptron performed poorest and was therefore considered for elimination.

To build an ensemble of various models, the experiments started by benching a set of Scikit-learn classifiers on the

Input: Learning set- S, Ensemble size B.
Output: Ensemble E
(1) E = φ
(2) **for** i = 1 to B **do**
(3) S=Boostrap Sample(S)
(4) C=Construct-Base Model (S^i)
(5) E = E ∪ {C^i}
(6) **return** E

Algorithm 4: Bagging.

Inputs:
(i) input data (x, y) Ni=1
(ii) number of iterations M
(iii) choice of the loss-function (y, f)
(iv) choice of the base-learner model h (x, θ)
Algorithm:
(**1**) initialize f_0 with a constant
(2) for t = 1 to M do
(3) compute the negative gradient gt(x)
(4) fit a new base-learner function h(x, θt)
(5) find the best gradient descent step-size ρt : ρt = arg minρ N i=1 yi, ft−1(xi) + ρh(xi, θt)
(6) update the function estimate: ft ⟵ ft−1 + ρth(x, θt)
(**7**) **end for**

Algorithm 5: Gradient boosting.

dataset. The considered models performed as shown in Table 1 .

It is evident that the base classifiers performed almost equally well in terms of accuracy.

A way to understand what is going on in an ensemble when the task is classification is to inspect the Receiver Operator Curve (ROC). This curve shows the tradeoff between precision and recall or the rate of true positives versus true negatives. Typically, different base classifiers make different tradeoffs. An ensemble can adjust these. The ROC curve obtained for the various classifiers and how they compare to the ensemble averaging technique is shown in Figure 2.

Random Forest reported the results in Table 2 and Figures 3 and 4.

A Cohen Kappa score of 0.932 was obtained for Random Forest.

Experiments with ExtraTreesClassifier gave the results shown in Table 3 and Figures 5 and 6.

AdaBoost gave the results reported in Table 4 and Figure 7.

Bagging gave the results reported in Table 5 and Figure 8.

Voting reported the results as shown in Table 6 and Figure 9.

4. Discussion of Results

Before the construction of classifier ensembles, it was found out that errors were significantly correlated for the different

Table 1: Benchmarking models.

Model	Score
Logistic Regression	0.943
Decision Tree	0.975
Support Vector Machines	0.990
Naïve Bayes	0.985
K Nearest Neighbors	0.954

Table 2: Random Forest.

Accuracy	Precision	Recall	F1
0.896	0.907	0.888	0.895

Table 3: ExtraTreesClassifier scores.

AUC	Accuracy	Precision	Recall	F1
0.972	0.887	0.889	0.884	0.889

Table 4: AdaBoost scores.

AUC	Accuracy	Precision	Recall	F1
0.993	0.915	0.861	1.0	0.925

Table 5: Bagging scores.

AUC	Accuracy	Precision	Recall	F1
0.995	0.911	0.873	0.971	0.919

Table 6: Voting scores.

Accuracy	Precision	Recall	F1
0.919	0.872	0.992	0.927

classifiers, which is to be expected for models that perform well. Yet most correlations were in the 50-80% span, showing decent room for improvement could be realized by ensembles.

When ROC curves were plotted for the averaging ensemble technique and the base algorithms, the ensemble technique outperformed Logistic Regression, Decision Tree, and kNN. This was as shown in Figure 2 where the curve for the ensemble technique approached the left topmost corner the most. The ensemble technique performed almost as well as the NB and SVM classifiers. Trying to improve the ensemble by removing the worst offender (Logistic Regression in Figure 2) gave a truncated ensemble ROC-AUC score of 0.990, a further improvement.

Table 7 summarizes the performance of the ensemble classifiers.

The highest ROC-AUC score was achieved by GradientBoosting (0.997) while the lowest was achieved by Random Forest (0.970). The figures were however all high and not very different from one another indicating that all ensemble techniques perform well. Voting was removed from the comparison as it was slow, especially with more base classifiers integrated. Some of the ensemble classifiers however did not generalize well. These were ExtraTreesClassifier and

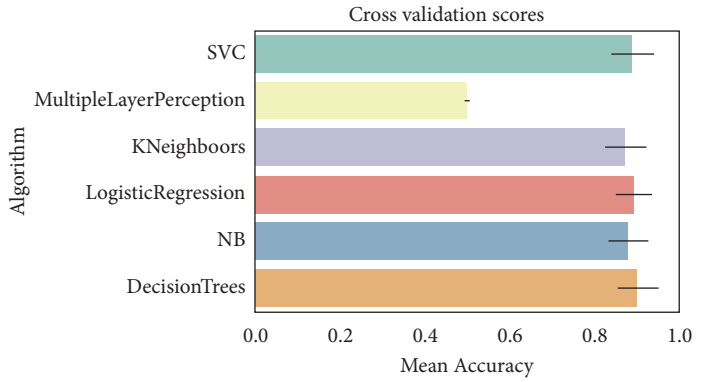


Figure 1: Comparison of base classifiers.

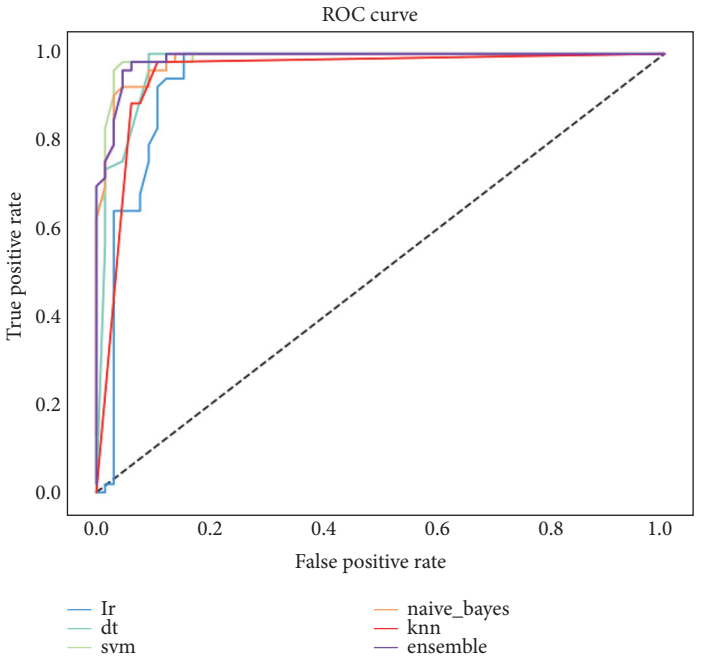


Figure 2: ROC comparison of simple average and base classifiers.

Table 7: Performance comparison of the ensemble classifiers.

Algorithm	AUC
Gradient Boosting	0.997
Random Forest	0.970
Extra Trees Classifier	0.972
AdaBoost	0.993
Bagging	0.995
Stacking	0.986
Simple Averaging	0.987

Random Forest. The rest of the ensemble techniques investigated generalized well including the slower voting ensemble technique.

It was evident that ensemble techniques improved obtained scores higher than some base learners though the performance difference was not significant as would be expected.

5. Conclusion

The study addressed the problem of detecting computer worms in networks. The main problem to be solved was that existing detection schemes fail to detect sophisticated computer worms that use code obfuscation techniques. In addition, many existing schemes use single parameter for the detection leading to a poorer characterization of the threat model hence a high rate of false positives and false negatives. The datasets used in many approaches is also outdated. The study aimed to develop a behavioral machine learning model to detect computer worms. The datasets used for the experiments were obtained from the University San Diego California Center for Applied Data Analysis (USCD CAIDA).

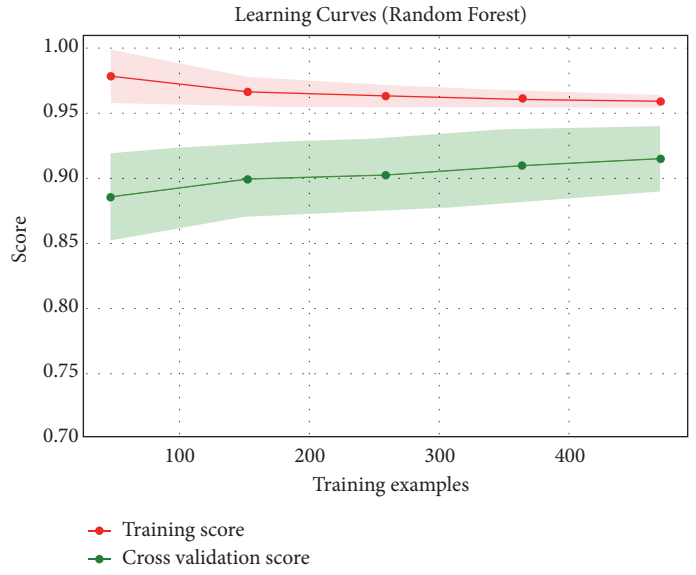


Figure 3: Learning curve for Random Forest.

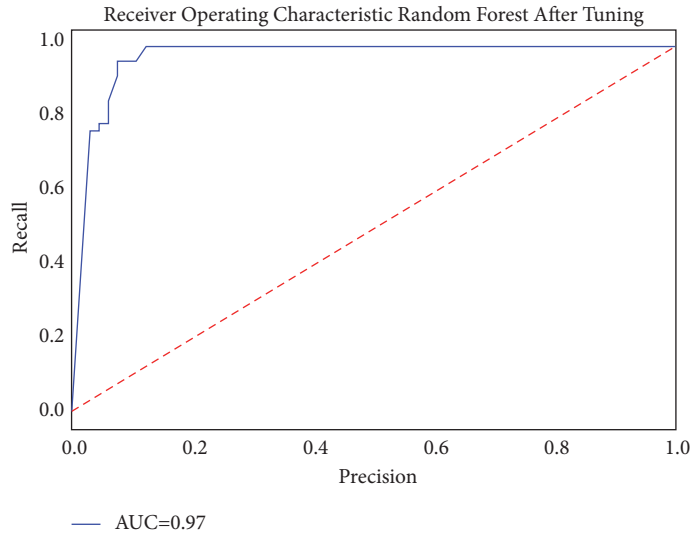


Figure 4: ROC for Random Forest.

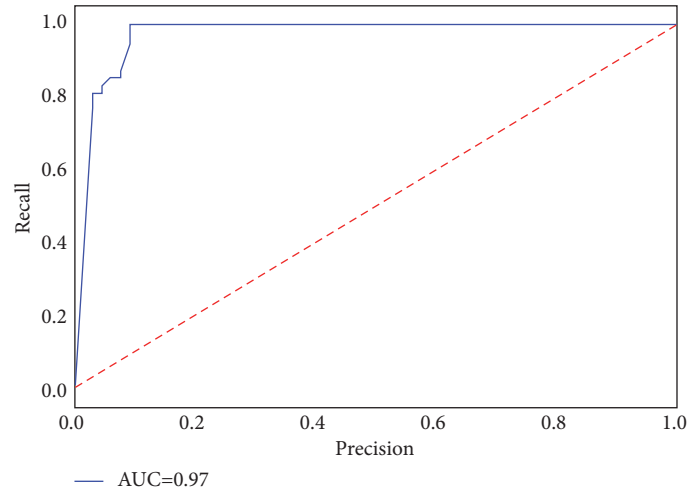


Figure 5: ROC for ExtraTreesClassifier.

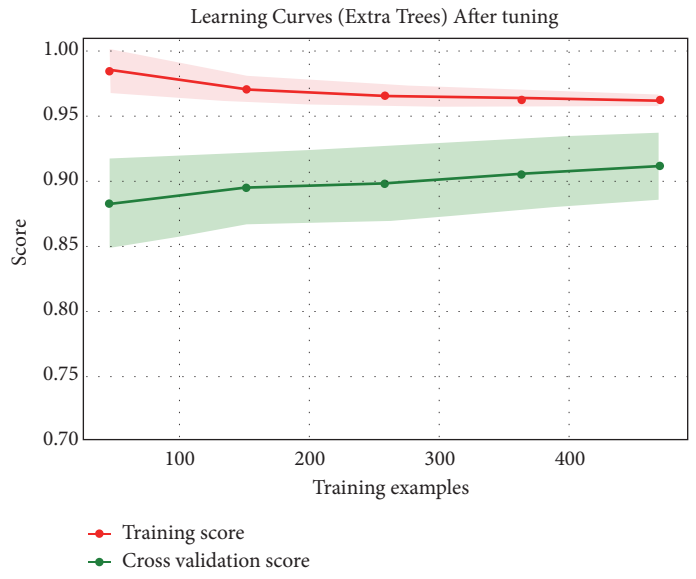


Figure 6: Learning curves for ExtraTreesClassifier.

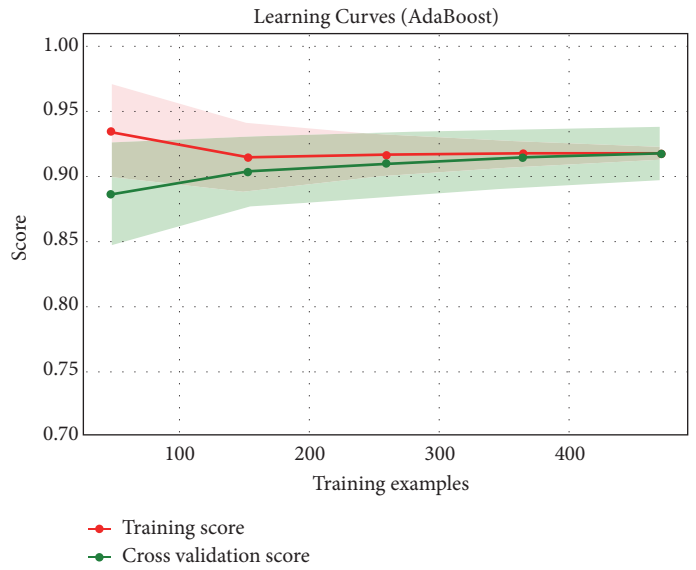


Figure 7: AdaBoost learning curve.

The results were promising in terms of accuracy and generalization to new datasets. There were no marked differences between the classifiers, especially when the datasets were standardized.

It is apparent that the particular classifier used may not be the determinant in classification in machine learning experiments but rather the choice of features. While this is largely consistent with other similar studies, it should be further confirmed by future research.

It is true that not all computer worms can be detected by a single method. In future, it is recommended that a combination of different detection approaches be combined to be able to detect as many types of computer worms as possible. Also, the span of features used for detection should be expanded to include even more features for the detection. The contribution of each feature to the detection ability should be documented.

Unsupervised learning has not been investigated in this research. Unlabeled traffic datasets are available to security researchers and practitioners. The cost of labeling them is high. This makes unsupervised learning useful for threat detection. The manual effort of labeling new network traffic can make use of clustering and decrease the number of labeled objects needed for the usage of supervised learning.

Data Availability

The packet capture (pcap) data used to support the findings of this study are provided by the UCSD, Center

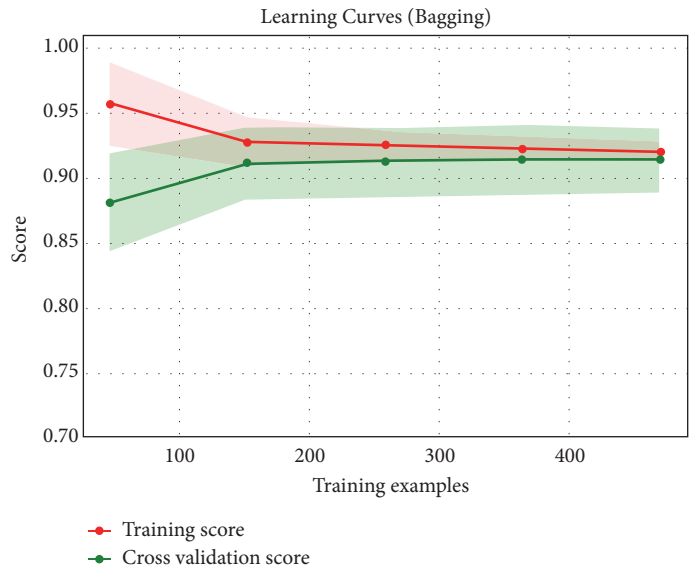


Figure 8: Bagging learning curve.

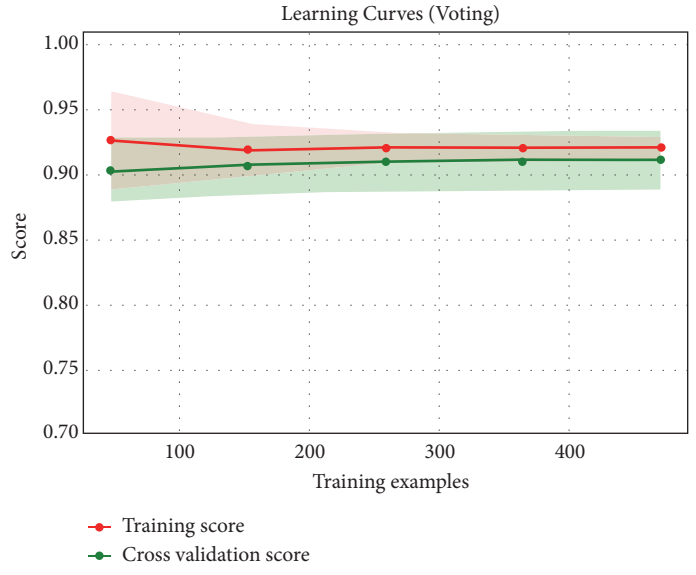


Figure 9: Learning curve for voting.

for Applied Internet Data Analysis. Two datasets were used: 1. the CAIDA UCSD Network Telescope "Two Days in November 2008" Dataset and 2. the CAIDA UCSD Network Telescope "Three Days of Conficker". They may be released upon application to IMPACT Cyber Trust, who can be contacted at the website address https://www.impactcybertrust.org/dataset_view?idDataset=382 and https://www.impactcybertrust.org/dataset_view?idDataset=383. The Corsaro tool that was used to process the pcap files is available as an open source tool at the Center for Applied Internet Data Analysis (CAIDA) website at https://www.caida.org/tools/measurement/corsaro/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] FireEye, The need for speed, https://www2.fireeye.com/ismg-incident-response-survey.html, 2018.

[2] D. Ellis, "Worm anatomy and model," in *Proceedings of the ACM Workshop on Rapid Malcode (WORM '03)*, pp. 42–50, ACM, October 2003.

[3] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the slammer worm," *IEEE Security & Privacy*, vol. 1, no. 4, pp. 33–39, 2003.

[4] D. Moore, C. Shannon, and K. Claffy, "Code-Red: a case study on the spread and victims of an internet worm," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment (IMW '02)*, pp. 273–284, Marseille, France, November 2002.

[5] O. Nelson, W. Mwangi, and I. Ateya, "A Hybrid Filter/Wrapper Method for Feature Selection for Computer Worm Detection

using Darknet Traffic," *International Journal of Computer Applications*, vol. 180, no. 44, pp. 12–17, 2018.

[6] N. Ochieng, W. Mwangi, and I. Ateya, "A Tour of the Computer Worm Detection Space," *International Journal of Computer Applications*, vol. 104, no. 1, pp. 29–33, 2014.

[7] M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 38–49, May 2001.

[8] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," *Journal of Machine Learning Research*, vol. 7, pp. 2721–2744, 2006.

[9] Z. Markel, "Machine learning based malware detection," https://apps.dtic.mil/dtic/tr/fulltext/u2/a619747.pdf, 2015.

[10] P. Vinod, V. Laxmi, M. S. Gaur, and G. Chauhan, "Detecting malicious files using non-signature-based methods," *International Journal of Information and Computer Security*, vol. 6, no. 3, pp. 199–240, 2014.

[11] J. Bai, J. Wang, and G. Zou, "A malware detection scheme based on mining format information," *The Scientific World Journal*, vol. 2014, Article ID 260905, 11 pages, 2014.

[12] M. Alazah, S. Venkatranan, and P. Watters, "Zero-day malware detection based on supervised learning algorithms of api call signatures," in *Proceedings of the 9th Australasian Data Mining Conference*, Data Mining and Analysis, 2011.

[13] A. Mamoun, H. Shamsul, A. Jemal et al., "A hybrid wrapper-filter approach for malware detection," *Journal of Networks*, vol. 9, no. 11, 2014.

[14] S. Muazzam, W. Morgan, and L. Joohan, "Detecting Internet worms using data mining techniques," *Journal of Systematics, Cybernetics and Informatics*, 2009.

[15] L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath, "Malware Images: visualization and automatic classification," in *Proceedings of the 8th International Symposium on Visualization for Cyber Security*, ACM, 2011.

[16] R. Benchea and D. T. Gavrilut, *Combining Restricted Boltzmann Machine adn One Side Perceptron for Malware Detection*, Springer International Publishing, 2014.

[17] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," in *Proceedings of the 10th International Conference on Malicious and Unwanted Software, (MALWARE '15)*, pp. 11–20, USA, October 2015.

[18] W. Huang and J. W. Stokes, "MtNet: a multi-task neural network for dynamic malware classification," in *Proceedings of the International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, Springer, 2016.

[19] The CAIDA UCSD Network Telescope, "Three Days of Conficker," http://www.caida.org/data/passive/telescope-3days-conficker_dataset.xml.

[20] E. Aben, "Conficker/Conflicker/Downadup as seen from the UCSD Network Telescope," Technical Report, CAIDA, February 2009, https://www.caida.org/research/security/ms08-067/conficker.xml.

[21] The CAIDA UCSD Network Telescope, "Two Days in November 2008," Dataset, http://www.caida.org/data/passive/telescope-2days-2008-dataset.xml.

[22] F. Pedregosa, G. Varoquaux, and A. Gramfort, "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[23] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.

[24] S. Dzeroski, P. Panov, and B. Zenko, *Ensemble Methods in Machine Learning*, Encyclopedia of Complexity and Systems Science, 2009.

[25] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *Journal of Japanese Society For Artificial Intelligence*, vol. 14, no. 771–780, p. 1612, 1999.

[26] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[27] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in Neurorobotics*, vol. 7, article 21, 2013.

[28] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 8485365, 9 pages
https://doi.org/10.1155/2019/8485365

Research Article

Malware Detection on Byte Streams of PDF Files Using Convolutional Neural Networks

Young-Seob Jeong, Jiyoung Woo, and Ah Reum Kang

SCH Media Labs, Soonchunhyang University, Asan 31538, Republic of Korea

Correspondence should be addressed to Ah Reum Kang; armk@arkang.net

Received 25 January 2019; Accepted 11 March 2019; Published 3 April 2019

Guest Editor: Pelin Angin

With increasing amount of data, the threat of malware keeps growing recently. The malicious actions embedded in nonexecutable documents especially (e.g., PDF files) can be more dangerous, because it is difficult to detect and most users are not aware of such type of malicious attacks. In this paper, we design a convolutional neural network to tackle the malware detection on the PDF files. We collect malicious and benign PDF files and manually label the byte sequences within the files. We intensively examine the structure of the input data and illustrate how we design the proposed network based on the characteristics of data. The proposed network is designed to interpret high-level patterns among collectable spatial clues, thereby predicting whether the given byte sequence has malicious actions or not. By experimental results, we demonstrate that the proposed network outperform several representative machine-learning models as well as other networks with different settings.

1. Introduction

Recently, as the file exchanges increase, intelligent attacks using documents with malicious code are increasing rapidly. Most Internet users are aware of the danger of files attached to mails or websites in forms of execution files. However, because people are not conscious at the documents, they become a good channel to deliver malware. Of the documented malware, PDF-based attack is one of the major attacks because of the flexibility of PDFs in contrast to other document formats. Most malicious PDF documents embed binary or JavaScript codes triggering specific vulnerabilities and perform malicious actions, as described in [1]. Various studies have been conducted to detect such malicious PDFs. The previous studies usually focused on feature extraction from the documents and applied the features to machine-learning models. Some of widely used features include the PDF structure information, entity property, metadata information, encoding method, content property, and lexicon-based features. Although such hand-crafted features have shown successful results, it requires much effort to design the features.

As the exponentially increasing amount of data, deep neural networks are drawing much attention in various fields such as image processing, natural language processing, sensor data processing, and speech recognition [2–6]. One of the main benefits of using the deep neural networks is that it is not necessary to define features because the networks automatically extract or compute features. In this work, we propose a novel approach using convolutional neural network (CNN) to tackle the malware detection. The contributions of this study can be summarized as follows: (1) we design a new CNN model well-suited to the malware detection on PDFs, (2) we demonstrate the performance of the proposed network by experiments using our manually labelled PDF dataset, and (3) we provide specific discussion about the experimental results.

The proposed approach does not require feature definition at all, but it is worth noting that it is still necessary to investigate or study the data structure in order to define better input data or design better network structures. As we target the PDFs in this work, we review the structure of the PDFs intensively and illustrate how we design the proposed network according to the characteristics of the input data (i.e., byte sequences). Although we conduct experiments only with the PDF documents, we expect that the proposed network is easily applicable to other formats (e.g., .rtf files).

In the following section, we review previous studies and the structure of PDF documents.

2. Background

2.1. Malware Detection on Stream Data. Malware is a program written to give an undesirable or harmful effect on a computer system. As the technology of malicious code generation by attackers becomes more intelligent, various researches have been conducted for detection and analysis of malicious codes. The malware can be divided into two categories: executables and nonexecutables. There have been many security programs to detect malicious actions in the form of portable executable files (e.g., Norton, Kaspersky). However, the nonexecutables (e.g., malicious actions in PDF documents) are easy to bypass some existing security programs and there is a high risk of false positives. Such document type malware is known to be more dangerous, as it is often considered as being insignificant by common users.

In order to detect the malicious documents, many studies focused on feature extraction based on the PDF structure analysis, where the features can be seen as a summary of the logical structure. In [7], a format-independent static analysis method, namely, a hierarchical document structure (hidost), was proposed. They adopted a structural multimap, where structural paths are represented by keys and the leaves are indicated by values. Several values of the multimap are reduced to a median value to constitute feature vectors that are used to train machine-learning algorithms. Cuan et al. [8] defined features using PDFiD Python script which verifies objects displayed in PDFs. As a simple trick called gradient-descent attack may bypass the proposed approach, they proposed two solutions: using a threshold for each feature and reducing the number of noncritical features. Smutz and Stavrou [9] extracted features from document metadata, such as the number of characters in titles and the size of images. Li et al. [10] developed a robust and secure feature extractor called FEPDF, which parses and extracts features from PDF documents. FEPDF consists of a matching method, detecting the PDF header, detecting all objects, detecting cross-reference, and detecting trailer. They emphasized that FEPDF can identify new malicious PDFs that have not been identified by existing feature extractors.

Note that these studies commonly require intensive feature engineering process, because it will almost determine the performance (i.e., accuracy) of the malware detection. In this paper, we designed a convolutional neural network to tackle the malware detection on nonexecutables. Although the proposed network allows getting results by just pushing the binary sequences into it without feature engineering, it is still important to investigate the structure of the target data. That is, the neural networks automatically capture features, but the features are obtained from input data which must be defined by an expert. Furthermore, understanding of the data structure helps to design better networks. In this paper, we target the PDF files because PDF-based attacks are known to be one of the major attacks recently. The following subsection provides detailed explanation about the structure of the PDFs.

2.2. Stream Data of PDF Files. The PDF document consists of header, body, cross-reference table, and trailer areas. The header contains document version information, and the body has objects that contain information about the actual document. The cross-reference table carries tables used to refer to objects. The trailer contains the root object and cross-reference table position information among the objects in the body region. The applications of PDFs (e.g., PDF reader) allow execution of JavaScript codes within the files, which means that any functions can be dynamically executed through JavaScript API.

The JavaScript code is contained in a stream, one of the PDF object types, where the object types include Boolean values, arrays, dictionaries, streams, and indirect objects. The stream is a collection of consecutive bytes of binary data that varies in length. The stream is normally supposed to contain large image files or page composition objects. A sample of stream object in a benign PDF file is shown in Figure 1. The object number is 141 and has 1,392 bytes. This stream can be decoded using FlateDecode filter, and the result is depicted in Figure 2. The function in this sample seems normal, but it is obvious that the users will be in danger if some malicious actions are contained in the stream.

In Figure 3, an example of a stream object obtained from a malicious PDF document is shown. The object number is 17 and its length is 1,279 bytes. The decoded beginning part of the malicious JavaScript code is shown in Figure 4, where it uses the 'unescape' function that decodes a string object encoded with 'escape' function. These two functions are intentionally employed to make the users confused, thereby making difficult to detect the malicious actions.

Figures 5 and 6 show another example of a malicious PDF stream object. The JavaScript code in Figure 6 calls the 'replace' function which uses regular expressions to replace certain characters with a desired string. As shown so far, the malicious JavaScript codes often appear obfuscated by encoding, and some functions (e.g., replace, unescape) trigger malicious actions.

There have been studies, of course, that focused on the JavaScript codes in the PDFs. Khitan et al. [12] defined features based on functions, constants, objects, methods, and keywords as well as lexical properties of JavaScript codes. Zhang [13] used features such as the number of objects, number of pages, and stream filtering information of JavaScript codes, together with the PDF structure information, entity characteristics, metadata information, and content statistics attributes. Liu et al. [1] proposed a context-aware approach based on the observation that the malicious JavaScript works differently from the normal JavaScript. This approach monitors suspicious activity based on JavaScript statements by putting the original code as an argument to 'eval' function and then opens the PDF document. The 'eval' function executes the delivered executable JavaScript code and returns a result. That is, it executes the code in a separated environment and finds suspicious codes running dropped malware.

In this study, we do not perform lexical analysis on JavaScript code or run PDF documents for dynamic analysis as in previous studies. We design a convolutional neural

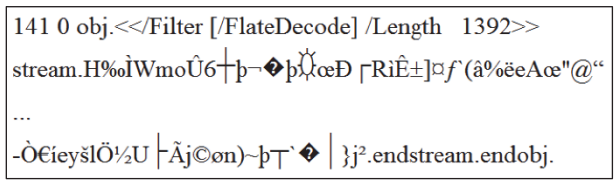


Figure 1: Example of a stream object in a positive PDF file.

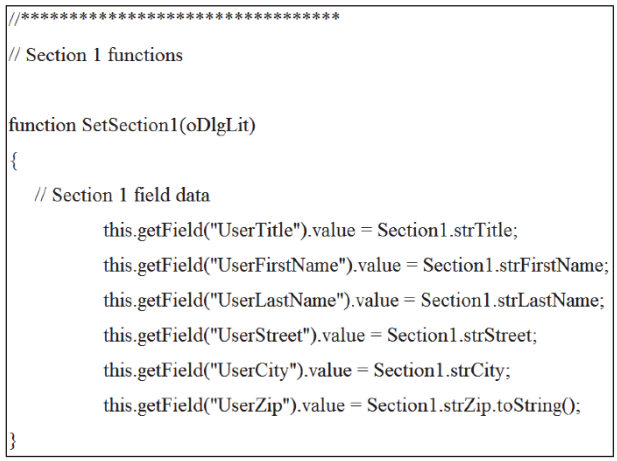


Figure 2: Example of a decoded stream content of a positive PDF file.

network that takes a byte sequence of a stream as an input and predicts whether the input sequence contains malicious actions or not. In next subsection, we briefly review the previous studies adopting neural networks to tackle the malware detection task.

2.3. Neural Networks for Malware Detection. There have been few studies thus far in applying neural networks to malicious software (malware) detection. Most recent works among them have used features extracted through dynamic analysis, so the features are extracted under the binary run in a virtualized environment. Kolosnjaji et al. [14] proposed a combination of convolutions and long short-term memory (LSTM) [15] to classify malware types based on the features of the API call sequences. Huang and Strokes [16] defined a manual 114 high-level features out of API calls as well as original function calls to predict malware types. This approach is essentially composed of two models, malware detection and malware type classification. The authors argued that the shared parameters of the two models contribute to improving the overall performance. These studies of dynamic analysis are performed on a certain nonpublic emulation environments, which makes it difficult to reproduce the works.

The other line of malware detection is the static analysis, in which features are obtained from the files without running them. Raff et al. [17] applied neural networks to the malware identification problem using the features in forms of 300 raw bytes of the PE-header of each file. This work showed

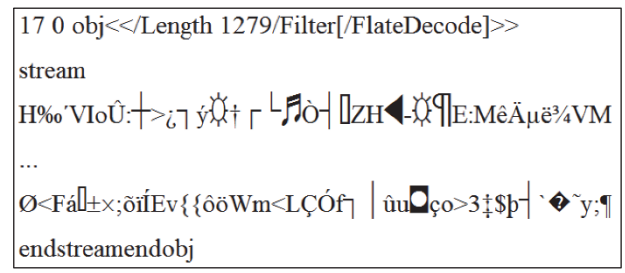


Figure 3: Example of a stream object in a malicious PDF file.

Figure 4: Example of a decoded stream content of a malicious PDF file.

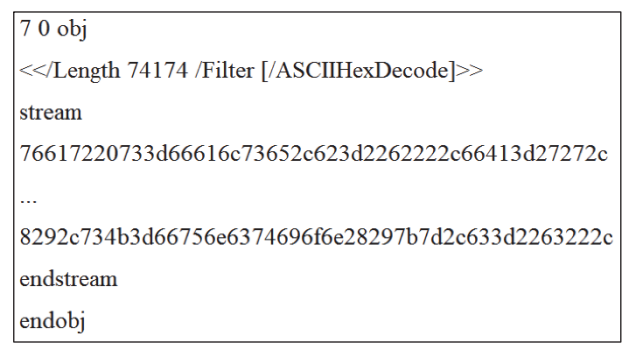


Figure 5: Another example of a stream object in a malicious PDF file.

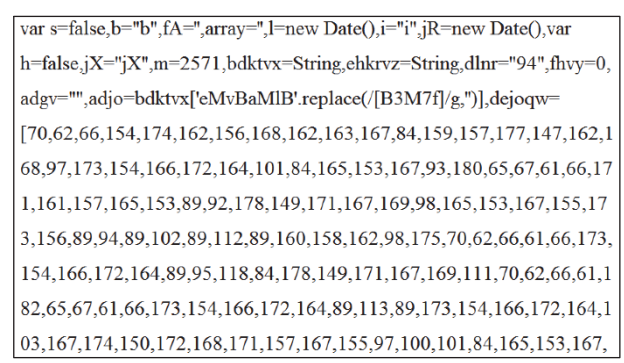


Figure 6: Another example of a decoded stream content of a malicious PDF file.

that the neural networks are capable of extracting underlying high-level interpretation from the raw bytes, which in turn makes it possible to develop malware detectors without hand-crafted features. Saxe and Berlin [18] utilized a histogram of byte entropy values from the entire files and defined a fixed length feature vector as the input of the neural networks. Le et al. [19] designed CNN-BiLSTM architecture, where the rational of taking bidirectional LSTM (BiLSTM) layer on top of CNN layer is that the BiLSTM layer may interpret sequential dependencies between different pieces generated by the CNN layer. They showed that the CNN layer is effective in representing local patterns of fixed length, and the BiLSTM has a potential to capture arbitrary sequential dependencies of executables. Raff et al. [20] derived a feature vector from the raw bytes and designed a shallow convolutional neural network with a gated convolutional layer, a global max-pooling layer, and a fully connected output layer. They insisted that their work is the first to define a feature vector from the entire binary, and it was hard to develop deeper networks due to the extraordinarily long byte strings (e.g., 1-2M length). Their biggest contribution is that the feature vector is obtained from the entire binary, so that it may grasp the global context of the entire binary. That is, the contents of a binary may have the high amount of positional variation or can be rearranged in arbitrary ordering, so they adopted the global-level feature vectors with a very large dimension. Although their network is designed to have ability to scale well with variable length of binary strings, it essentially will not be applicable to any longer binary strings (e.g., 3-4M length). All of these studies commonly utilized raw bytes of executables.

The proposed network in this paper is designed to take a byte sequence within the nonexecutables as an input and generates an output based on high-level patterns of collectable spatial clues, which implies that the proposed network is applicable to byte sequences with variable lengths. In the following section, we illustrate how we design the proposed network according to the characteristics of the input data (i.e., byte sequences).

3. Proposed Method

To detect malicious actions without heavy feature engineering, we designed a deep learning model against stream objects and discriminate the maliciousness in the object level. The stream objects have no size limitation, and a certain part of the stream exhibits malicious actions while the other part does not. Such high-level location invariance makes it difficult to detect the maliciousness of the object. Among deep learning models, convolutional neural networks (CNN) are known as successful in detecting locally contextual patterns. The CNN models have brought dramatic performance advance in the area of image processing [21, 22], and one of its benefits is that it works with smaller amount of data, compared to other deep learning models (e.g., recurrent neural networks, fullyconnected neural networks).

A graphical representation of our proposed network is depicted in Figure 7, where the network consists of an embedding layer, two convolutional layers, a pooling layer, a fully connected layer, and an output layer. Note that the figure just shows a structure of the network, and the dimensions of the layers and the number of channels must be much larger. The E denotes the embedding size and S means a sequence length. Rather than directly submitting the raw byte values into convolution (i.e., using a scaled version of a byte's value from 0 to 255), we adopt an embedding layer to map each byte to a E-dimensional feature vector because the byte values do not imply intensity but convey some contextual information. That is, given a byte sequence of length S, the $E \times S$ real-valued embedding matrix is computed during the training process, so that the matrix will help to grasp a wider breadth of input patterns.

The embedding layer makes it possible to represent meaning of each byte by incorporating all the byte sequences. As discussed in [20], the raw values of byte sequences do not simply represent intensity, and it will be better to find an alternative way to see the values. For example, a byte value 160 does not imply 'better' or 'stronger' intensity than 130, but the two values must convey different meaning. In the 'word embedding' concept in the natural language processing (NLP) field, similar words (e.g., 'hi' and 'hello') are close to each other in the embedding space, whereas opposite words are far from each other. Likewise, our embedding layer interprets the contextual meaning of byte values and represents them on the embedding space.

Several locally adjacent E-dimensional vectors generated from the embedding layer are then fed into the first convolutional layer followed by another convolutional layer. The first convolutional layer is designed to take a $C_1 \times E$ matrix which is supposed to carry spatial clues of malicious actions, in the hope that the collected clues will be enough for the entire network to make a wise decision. In the recent work [20], a convolutional neural network designed for analysing the entire sequence at once was proposed, but this network will not be applicable to longer sequences. In contrast, our network collects simple local clues and generates high-level representation, which means that the proposed network is available to all sequences with variable lengths. Each convolutional layer takes one or more adjacent vectors (or values) from its previous layer as an input and generates an output value by a summation of element-wise multiplication with a filter. This filter, also known as a kernel, is computed during the training process.

The two convolutional layers have K_1 and K_2 distinct channels (i.e., kernels), respectively, to find different spatial patterns. The convolutional layer slides or convolves from the top-left corner to the bottom-right corner with an arbitrary stride, thereby resulting matrix or vector will carry a compilation of spatial patterns throughout the input. Similar to many studies related to CNN [2, 23, 24], our network has consecutive convolutional layers because multiple stacked convolutional layers are known to have better (higher-level) representation than a single convolutional layer. We have two convolutional layers stacked along the network depth, where the first convolutional layer takes a $C_1 \times E$ matrix as an input, and the second convolutional layer takes a $C_2 \times 1$ vector as an input, respectively. The first layer is supposed to catch simple clues, which will be used for the second

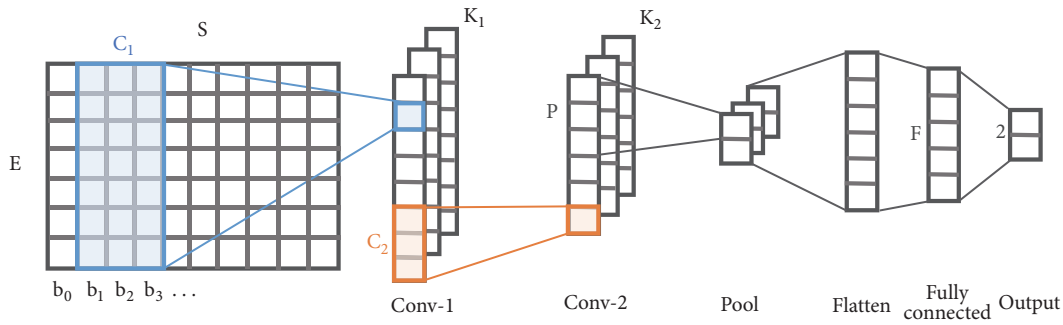


Figure 7: Graphical representation of the proposed convolutional neural network.

layer to understand underlying higher-level patterns. For example, the first layer may capture simple clues of JavaScript statements (e.g., replace), while the second layer represents contextual information about their types or arguments.

One may argue that stacking more convolutional layers might be better, because the deeper network is often known to be capable of extracting more complicated patterns. This might be true, but it should be noted that the deeper network is not always better than the shallow networks. The length of network must be considered carefully by investigating the data; too complicated network will probably overfit, whereas too simple network will underfit. By examination into the sequence data, we conclude that the two convolutional layers will be enough to capture the variety of the spatial patterns of malicious actions. We also, of course, show that this structure is indeed better than deeper networks through experiments.

The high-level representation obtained from the two consecutive convolutional layers is submitted to the pooling layer that helps to focus on some representative or primary patterns. Among several types of pooling layer such as average-pooling and L2-norm pooling, we adopt the max-pooling layer as it is generally known to be effective in various tasks. Similar to the convolutional layer, the pooling layer slides from the top-left corner to the bottom-right corner with an arbitrary stride, resulting in output vectors of much smaller size. The output vectors are flattened or concatenated to form a one-dimensional vector that will be passed to the F-dimensional fully connected (FC) layer. The FC layer collects the primary patterns from the pooled values, and the last output layer represents how likely the given byte sequence embeds malicious actions.

4. Experiments

4.1. Dataset. We collected PDF files from various antivirus or antimalware companies and constructed a dataset through manual labelling process. There often exist several stream objects in a malicious PDF file, and one of more streams contains malicious actions. As we observed that all malicious streams contain JavaScript codes, we mark the objects as 'malicious' if they contain JavaScript codes, otherwise 'benign'. The objects of benign PDF files are all marked as 'benign'. Note that we do not employ whether to contain JavaScript as a feature at all. The proposed network will see only the raw byte sequence. The object streams are hexadecimal representations of text streams, as shown in Figure 8, where each line of the figure is a part of the conversion of Figures 2, 4, and 6, respectively.

Table 1: Data statistics, where the positive and negative percentages imply the proportions of the malicious streams and the benign streams, respectively.

Size (# of instances)	1,978
Positive : Negative (%)	50.0 : 50.0
Length (dimension)	1,000

The data statistics are summarized in Table 1, where each data instance is defined as a byte-level stream object. As shown in Figure 9, for each data file, multiple byte streams are collected, each of which is manually labelled with a positive (malicious) or a negative (benign) tag. The total amount of originally collected byte sequences was 4,371, but randomly downsampled 989 negative instances.

Note that the byte streams have different lengths as described in Figure 10, but the proposed network cannot handle such sequences of variable lengths. As the network is designed to grasp high-level patterns from collectable spatial clues of malicious actions, it is not necessary to use the whole sequence of different length at a time. Rather, we padded short sequences and cut away the remaining part from long sequences, so that all sequences have the same length of 1,000. When the network is trained with these fixed length sequences, it can be applied to divided byte sequences with the same size in a target file, so that it will predict whether the target file contains malware or not.

4.2. Model. We compared the proposed network with several representative classification methods such as support vector machine [25], decision tree, naïve bayes, and random forest [26]. The brief description and parameter settings of the methods are summarized in Table 2.

We also examined various settings for our network, and the best setting was found as follows: (1) the embedding dimension $E = 25$, (2) C_1 and C_2 are set to 3, respectively, (3) pooling layer dimension $P = 100$, (4) K_1 and K_2 are 32 and 64, respectively, and (5) fully connected layer dimension $F = 128$, where the notations can be found in Figure 7. The two

```
2F 2F 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
76 61 72 20 75 6E 65 73 3D 75 6E 65 73 63 61 70 65 3B 0D 0A 76 61 72 20 70 47 76 52 49 4A 5A 70 71
76 61 72 20 73 3D 66 61 6C 73 65 2C 62 3D 22 62 22 2C 66 41 3D 27 27 2C 61 72 72 61 79 3D 27 27 2C
```

Figure 8: Byte streams.

Table 2: Description of comparative classifiers and parameter settings.

Model	Description
Naïve bayes (NB)	(i) Probabilistic classifier based on the bayes' theorem (ii) Assumes the independence between features
Decision tree (DT)	(i) C4.5 classifier using J48 algorithm (ii) Confidence factor for pruning: 0.25
Support vector machine (SVM)	(i) Non-probabilistic binary classification model that finds a decision boundary with a maximum distance between two classes (ii) Kernel: Poly (iii) Exponent=1.0, Complexity c=1.0 (iv) Trained via sequential minimal optimization algorithm [11]
Random forest (RF)	(i) Kind of ensemble model that generates final result by incorporating results of multiple decision-trees (ii) #trees=100 (iii) #features=log(#trees)+1 (iv) Each tree has no depth-limitation

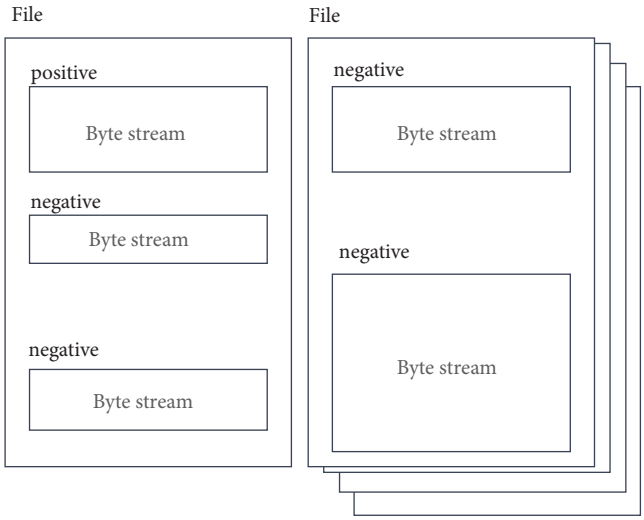


Figure 9: Positive or negative tags for each byte stream in the data files.

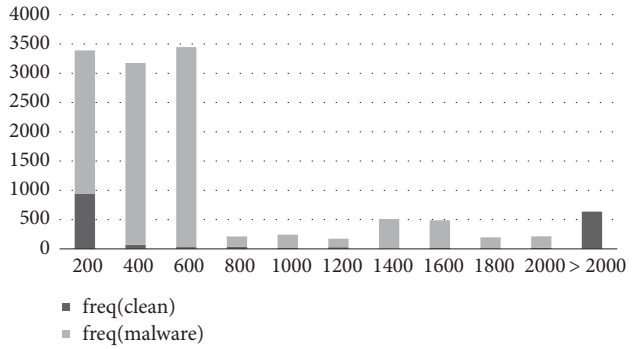


Figure 10: Frequencies of clean and malware PDF byte streams with different lengths, where the x-axis represents stream lengths and the y-axis shows frequencies.

convolutional layers have strides (1, 25) and (1, 1), respectively, and the pooling layer has a nonoverlapping stride (100, 1). The sequence length S must be 1,000. Every layer takes rectified linear unit (ReLU) as an activation function except for the output layer that takes a softmax function. The cost function is defined as a cross entropy over all nodes of the output layer. Consequently, the total numbers of trainable parameters are 89,371.

We adopt Adam's optimizer [27] with an initial learning rate of 0.001 to train our network. Our training recipe is as follows: (1) L2 gradient-clipping with 0.5, (2) drop-out [28] with a keeping probability 0.25 for the fully connected layer, and (3) batch normalization [29] for the two convolutional layers. We tried to use the regularization methods such as L2 regularization and decov [30] and observed no performance improvements, as the batch normalization and drop-out are known to have regularization effect themselves. The weight matrices of the convolutional layers, the FC layer, and the output layer are initialized by He's algorithm [31], and bias vectors are initialized with zeros. In the following

Table 3: Experimental results with the PDF dataset, where the two values of each cell are of 'benign' and 'malicious', respectively.

Model	Precision	Recall	F1
DT	96.00 / 90.30	89.70 / 96.30	92.70 / 93.20
NB	88.40 / 99.70	99.70 / 87.00	93.70 / 92.90
SVM	94.70 / 98.90	99.00 / 94.40	96.80 / 96.60
RF	93.50 / 99.40	99.40 / 93.10	96.40 / 96.10
Emb+Conv+Conv+Pool+FC	99.76 / 100.0	97.37 / 97.37	*98.48 / 98.65*
Conv+Conv+Pool+FC	99.78 / 100.0	92.62 / 97.27	95.71 / 98.61
Emb+Conv+Pool+FC	99.73 / 100.0	94.94 / 97.78	97.12 / 98.87
Emb+Conv+Conv+FC	99.67 / 100.0	97.78 / 92.32	98.55 / 96.00
Emb+Conv+Conv+Conv+Pool+FC	99.70 / 100.0	92.21 / 95.35	95.36 / 97.60

subsection, we demonstrate the performance of our network by experimental comparison with the other classifiers and different networks.

4.3. Result. One unfortunate aspect of the field of malware detection is that there is no available public dataset for various reasons. The dataset easily obtainable from public is often not of a sufficient quality, so previous studies could not compare performance (i.e., accuracy) across works because of different data characteristics and labelling procedures. It is inevitably hard to compare our performance with other state-of-the-art studies for the same reason. We compare our network with some comparative machine-learning methods and CNN models with different settings. The measurement is conducted using 10-fold cross validation, where the performance values (i.e., F1 score, precision, and recall) are averaged values of three distinct trials.

The experimental results are described in Table 3, where the two values for each cell correspond to the 'benign' and 'malicious' classes, respectively. For example, the F1 scores of random forest (RF) are 96.4 for 'benign' and 96.1 for 'malicious'. For the four traditional machine-learning models (e.g., DT, NB, SVM, and RF), the values of the input sequence are treated as nominal values. We tested five different structures of CNNs. The first network, Emb+Conv+Conv+Pool+FC, is the best structure which has an embedding layer, two consecutive convolutional layers, a pooling layer, and a fully connected layer followed by an output layer. The number of epoch differs from networks according to the complexity (i.e., the number of layers and parameters) of the networks and the dataset size. For instance, the first network is trained through 30 epochs, whereas training of the second network requires 25 epochs.

Among the traditional machine-learning models, the support vector machine (SVM) exhibits the best F1 scores and random forest (RF) has the comparable results. As also shown in Table 3, it seems obvious that the five CNNs outperform the traditional machine-learning models. From the results of the five networks, we can find two main observations. First, the embedding layer (Emb) seems to play a significant role in better representation of byte sequences, as the second network without the embedding layer exhibits much worse F1 scores than the other networks. Second, the stacked convolutional layers enable interpreting high-level patterns, and the optimal number of layers seems to be two. The third network having a single convolutional layer and the fifth network having three convolutional layers are worse than the first network. The fifth network especially has the worst F1 score among five networks. This indicates that more stacked layers are not always better than shallow networks.

4.4. Discussion. The experimental results can be summarized in two aspects. First, the convolutional neural networks showed superior performance than the traditional machine-learning models. The F1 score of the proposed network is almost 2% greater than the SVM, which can be explained that the convolutional neural networks have better comprehensive power to analyse the underlying spatial patterns of the byte sequences. Second, it seems that the embedding layer followed by the two convolutional layers is best suited to representing high-level patterns of malicious actions. Less or more stacked convolutional layers gave worse results.

Other than the two aspects, we need to discuss about the parameter settings for training. The results of Table 3 are obtained from the network trained with the dimensions and the parameters as aforementioned in Model part. We found the optimal parameter setting by grid search, and some remarkable results with different settings and dimensions are summarized in Table 4. The first two rows correspond to the embedding size E, and the last two rows are associated with the pooling size P. Other remaining rows are related to the training recipe, such as drop-out, gradient-clipping, and batch normalization. The drop-out together with the batch normalization was helpful to generalize the network, and we observed no improvements with additional regularization methods. The gradient-clipping made the network more robust, as it prevented from tripping over desirable points of the gradient space.

We also checked the training time of the comparable methods. Table 5 shows the elapsed seconds for training different methods. The training of the four traditional methods is performed on a machine of Xeon E5-2620 V4 with 128GB RAM, while the CNN models are trained using a machine of i7-9700K with 64GB RAM and two RTX 2080 Ti. The training time of the CNN models strongly depends on the performance of GPUs. All methods are trained using the 1,978 instances, and the number of epochs is equally 30 for the CNN models. Note that the fourth CNN model without a

Table 4: Experimental results with different settings, where the two values of each cell are of 'benign' and 'malicious', respectively.

Dimensions & Training options	Precision	Recall	F1
$E = 15$	99.68 / 100.0	92.52 / 95.75	95.34 / 97.82
$E = 35$	99.73 / 100.0	94.03 / 97.47	96.58 / 98.71
Drop-out with 0.5	99.70 / 100.0	93.53 / 97.37	96.23 / 98.66
L2 gradient-clipping with 0.3	99.74 / 100.0	95.25 / 97.17	97.17 / 98.55
L2 gradient-clipping with 0.7	99.70 / 100.0	92.11 / 97.78	95.14 / 98.86
w/o batch-normalization	99.70 / 100.0	95.96 / 97.27	97.68 / 98.61
$P = 50$	99.70 / 100.0	95.05 / 97.07	97.13 / 98.50
$P = 150$	99.76 / 100.0	93.83 / 97.88	96.29 / 98.92

Table 5: Training time of comparable methods.

Model	Time (seconds)
DT	0.69
NB	0.16
SVM	750.88
RF	1.56
Emb+Conv+Conv+Pool+FC	22.07
Conv+Conv+Pool+FC	21.12
Emb+Conv+Pool+FC	13.85
Emb+Conv+Conv+FC	31.52
Emb+Conv+Conv+Conv+Pool+FC	23.16

pooling layer takes much longer time than the other three CNN models. The reason is that the pooling layer has an effect of reducing the filter sizes, so the fourth CNN model has greater number of parameters to train. Except for the SVM, the traditional models seem computationally more efficient than the CNN models. One of the reasons for the lagging of training SVM must be the use of Poly kernel function. We may use another kernel functions (e.g., linear kernel), of course, but it will probably degrade accuracy. As the training of random forest (RF) is much faster than the CNN models, it might be preferred to choose the RF model if we want efficiency with a small loss of effectiveness (i.e., accuracy).

5. Conclusion

The threat of malicious documents keeps growing, because it is difficult to detect the malicious actions within the documents. In this work, we proposed a new convolutional neural network designed to take a byte sequence of nonexecutables as an input and predicts whether the given sequence has malicious actions or not. We illustrated how we design the network according to the characteristics of the input data and provided discussion about the experiments using the manually labelled dataset. The experimental results showed that the proposed network outperforms several representative machine-learning models as well as other convolutional neural networks with different settings. Though we conducted experiments only with the PDF files, we expect that this approach can be applicable to other types of data if they contain byte streams. Therefore, as a future work, we will collect data of other file types (e.g., .rtf files) and perform further investigation.

Data Availability

We disclose our dataset as well as a code to public (https://sites.google.com/view/datasets-for-public).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Soonchunhyang University Research Fund (no. 20170265). This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2017R1D1A3B030360 50).

References

[1] D. Liu, H. Wang, and A. Stavrou, "Detecting malicious JavaScript in pdf through document instrumentation," in *Proceedings of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 100–111, IEEE, Atlanta, GA, USA, June 2014.

[2] T. T. Um, F. M. J. Pfister, D. Pichler et al., "Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 216–220, ACM, Glasgow, Scotland, 2017.

[3] Z. M. Kim, Y. S. Jeong, H. R. Oh et al., "Investigating the impact of possession-way of a smartphone on action recognition," *Sensors*, vol. 16, no. 6, pp. 1–5, 2016.

[4] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1746–1751, Association for Computational Linguistics, Doha, Qatar, October 2014.

[5] A. Hannun, C. Case, and J. Casper, "Deep speech: scaling up end-to-end speech recognition," *Computing Research Repository*, pp. 1–12, 2014.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern*

Recognition, pp. 779–788, IEEE, Las Vegas Valley, NV, USA, July 2016.

[7] N. Šrndić and P. Laskov, "Hidost: a static machine-learning-based detector of malicious files," *EURASIP Journal on Information Security*, vol. 2016, no. 1, p. 22, 2016.

[8] B. Cuan, A. Damien, C. Delaplace et al., *Malware Detection in PDF Files Using Machine Learning [PhD. Thesis]*, REDOCS, 2018.

[9] C. Smutz and A. Stavrou, "Malicious PDF detection using metadata and structural features," in *Proceedings of the 28th Annual Computer Security Applications Conference*, pp. 239–248, Orlando, Fla, USA, December 2012.

[10] M. Li, Y. Liu, M. Yu et al., "FEPDF: a robust feature extractor for malicious PDF detection," in *Proceedings of the 2017 IEEE Trustcom/BigDataSE/ICESS*, pp. 218–224, IEEE, Sydney, Australia, August 2017.

[11] J. C. Platt, "Sequential minimal optimization: a fast algorithm for training support vector machines," in *Advances in Kernel Methods – Support Vector Learning*, MIT Press, Cambridge, Mass, USA, 1998.

[12] S. J. Khitan, A. Hadi, and J. Atoum, "PDF forensic analysis system using YARA," *International Journal of Computer Science and Network Security*, vol. 17, no. 5, pp. 77–85, 2017.

[13] J. Zhang, "MLPdf: an effective machine learning based approach for PDF malware detection," *Security and Cryptography*, 2018.

[14] B. Kolosnjaji, A. Zarras, G. Webster et al., "Deep learning for classification of malware system call sequences," *Lecture Notes in Computer Science*, vol. 9992, pp. 137–149, 2016.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] W. Huang and J. W. Stokes, "MtNet: a multi-task neural network for dynamic malware classification," *Lecture Notes in Computer Science*, vol. 9721, pp. 399–418, 2016.

[17] E. Raff, J. Sylvester, and C. Nicholas, "Learning the PE header, malware detection with minimal domain knowledge," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 121–132, ACM, Dallas, TX, USA, 2017.

[18] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," in *Proceedings of the 10th International Conference on Malicious and Unwanted Software*, pp. 11–20, IEEE, Fajardo, PR, USA, October 2015.

[19] Q. Le, O. Boydell, B. Mac Namee, and M. Scanlon, "Deep learning at the shallow end: Malware classification for non-domain experts," *Digital Investigation*, vol. 26, pp. S118–S126, 2018.

[20] E. Raff, J. B. Barker, J. Sylvester et al., "Malware detection by eating a whole EXE," in *Proceedings of the in Proceedings of the Workshops of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 268–276, New Orleans, LA, USA, 2018.

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems 25*, Lake Tahoe, NV, USA, December 2012.

[23] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, IEEE, Boston, MA, USA, June 2015.

[24] G. Huang, Z. Liu, L. Maaten et al., "Densely connected convolutional networks," in *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2261–2269, IEEE, Honolulu, HI, USA, July 2017.

[25] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "Training algorithm for optimal margin classifiers," in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pp. 144–152, ACM, Pittsburgh, PA, USA, July 1992.

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

[27] D. P. Kingma and J. L. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the in Proceedings of the 3rd International Conference for Learning Representations*, San Diego, Calif, USA, 2015.

[28] N. Srivastava, G. Hinton, A. Krizhevsky et al., "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[29] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, pp. 448–456, ACM, Lille, France, July 2015.

[30] M. Cogswell, F. Ahmed, R. Girshick et al., "Reducing overfitting in deep networks by decorrelating representations," in *Proceedings of the 4th International Conference for Learning Representations*, San Juan, PR, USA, 2016.

[31] K. He, X. Zhang, S. Ren et al., "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *Proceedings of the 15th IEEE International Conference on Computer Vision*, pp. 1026–1034, Santiago, Chile, December 2015.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 1354346, 11 pages
https://doi.org/10.1155/2019/1354346

Research Article

Integrity Audit of Shared Cloud Data with Identity Tracking

Yun Xue Yan,[1,2] Lei Wu ,[1,2,3] Wen Yu Xu,[1,2] Hao Wang,[1,2] and Zhao Man Liu[1,2]

[1]*School of Information Science and Engineering, Shandong Normal University, Jinan, China*
[2]*Shandong Provincial Key Laboratory for Novel Distributed Computer Software Technology, Jinan, China*
[3]*Shandong Provincial Key Laboratory of Software Engineering, Jinan, China*

Correspondence should be addressed to Lei Wu; wulei@sdnu.edu.cn

Received 5 January 2019; Accepted 19 February 2019; Published 6 March 2019

Guest Editor: Pelin Angin

More and more users are uploading their data to the cloud without storing any copies locally. Under the premise that cloud users cannot fully trust cloud service providers, how to ensure the integrity of users' shared data in the cloud storage environment is one of the current research hotspots. In this paper, we propose a secure and effective data sharing scheme for dynamic user groups. (1) In order to realize the user identity tracking and the addition and deletion of dynamic group users, we add a new role called Rights Distribution Center (RDC) in our scheme. (2) To protect the privacy of user identity, when performing third party audit to verify data integrity, it is not possible to determine which user is a specific user. Therefore, the fairness of the audit can be promoted. (3) Define a new integrity audit model for shared cloud data. In this scheme, the user sends the encrypted data to the cloud and the data tag to the Rights Distribution Center (RDC) by using data blindness technology. Finally, we prove the security of the scheme through provable security theory. In addition, the experimental data shows that our proposed scheme is more efficient and scalable than the state-of-the-art solution.

1. Introduction

As an emerging network storage technology, cloud storage has been extended and developed in cloud computing. Cloud computing systems are transformed into cloud storage systems when the core of computing and processing is to store and manage massive data. In simple terms, cloud storage is an emerging solution that puts storage resources on the cloud for people access.

The user can access data on the cloud easily through any connected device whenever and wherever. Through data storage and sharing services in cloud computing, group members can share data in the form of a group. As a member of a group, users can not only access the shared data, but also modify the shared data. While cloud computing makes it easier for users to share data, users are still concerned about the security of data, especially the integrity of data, due to some security factors in cloud storage. The effective way is to use third party auditor (TPA) to achieve the purpose of validating shared data integrity. However, third party auditor (TPA) can obtain the block identifier (that is, the identity of each shared block signer) during the process of verifying the data integrity. If these identity information and confidential information in the shared data group cannot get effective protection, they will be leaked to a third party auditor (TPA) such as the situations that user in the group plays a crucial role or data block in the shared data has higher value.

Although the current public auditing scheme for sharing data solves the problem of user identity protection, it also brings dynamic changes in the group. However, the identity of group members who maliciously modify the shared data cannot get traced. We can observe that the amount of computation is comparatively large during the signature of data blocks by cloud users, which takes users a long time with limited resources. This paper proposes an auditing scheme that supports user identity tracking and lightweight sharing of cloud data, which enables traceability of user identities and reduces the burden on the resource constrained users. Using the data storage and sharing services provided by cloud server, legitimate users can easily form a group by sharing data with each other. That is to say, the users can create data and share it with others in the group. Users in the group can not only access the shared data, but also modify the shared data. Although cloud service providers

provide users with a secure and reliable storage environment as much as possible, data integrity can still be compromised. For example, it is considered that operational errors, data hardware and software failures, may lead to data tampering and data loss. This series of problems happened to us [1].

2. Related Work

Users always pay more attention to data security in the cloud. In recent years, data integrity schemes have become one of the research hotspots. With the help of data integrity schemes, any data corruption or deletion can be discovered in time and then necessary measures can be taken to recover the data. To develop a better understanding of data integrity schemes, we carry out the relevant work from the audit model, soundness, and other aspects.

Performance. Many researchers have proposed a series of schemes to this problem. On the one hand, how to solve the problem of user revocation? Wang et al. [2–7] noticed the problem of shared data integrity verification and proposed a public auditing method that supports efficient user revocation for shared data. To sum up, this scheme introduces proxy resignature technology to solve the problem. However, when the user is revoked, the cloud server is allowed to replace the previously signed data block of the revoked user to a legal group instead of the group member, which can cause efficiency problem. In addition, in scheme [8], the authors propose to enable efficient user revocation in identity-based cloud storage auditing for shared big data. On the other hand, Yu et al. [8, 9] proposed the issue of key security among cloud users. In these schemes, the key exposure in one time period does not affect the security of cloud storage auditing in other time periods and verifiable out-sourcing of key updates.

Identity Privacy. With the development of related technologies in cloud computing, public audit of shared data integrity has attracted more and more attention. Yu et al. [10] proposed that the storage and sharing services of cloud servers allow users to share data in the form of a group. As a group member, they have the right to view and modify shared data. Although users can easily share data, data integrity issues remain [11, 12]. Using TPA for public auditing results in the leakage of user's identity privacy [13]. Wang et al. [14] fully considered the confidentiality of the data in the public audit process and proposed a privacy scheme that used ring signature to protect group member. Adopting the ring signature can ensure that the TPA protects the user's identity privacy while verifying the integrity of the data. However, the efficiency of the scheme is reduced by the increasing number of team members. Meanwhile, the client also takes a lot of computing. Therefore, the scheme does not apply to large user groups. Shen et al. [15] proposed a lightweight auditing scheme for shared data privacy protection, taking full account of the computational limitations of the resource constrained client. Using data blindness methods, the scheme allows (TPM) Third Party Medium instead of group users to sign the data. It not only reduces the burden on the client, but also ensures the privacy of identity during public auditing. Thus the identity of the data owner can be protected. However, this scheme does not support group dynamics and the traceability of data blocks. Wang et al. [16] proposed another public audit method for sharing data privacy protection. Using dynamic broadcast technology, group members can be signed as the owner of the data when modifying the shared data, thereby protecting the privacy of the group members. It not only realizes the dynamic operation of data by group members, but also supports group dynamics. However, this scheme does not protect the identity of data owner, making the TPA steal the identity of the data owner during public auditing, and it does not support the traceability of data blocks.

Public Auditability/Private Auditability. The first method [17] allows only the data owner to audit. The second [18] method allows a third party auditor to audit. The audit process in both approaches is performed without retrieving the remote data. If only the data owner can verify the integrity of the outsourced data, then this scheme is considered to provide private auditability. However, in some cases, it is not practically feasible for the data owner to remain online all time for data integrity verification. Hence, the data owner can delegate this responsibility for integrity verification to a third party auditor or other users. A data integrity scheme must have public auditability property to support this audit delegation.

Dynamic Data Handling. Data can be either static (backup or archival data) or dynamic nature (supporting operations like insertion, deletion, and modification). Providing integrity for dynamic data is more challenging than static data or just attaching data. Most of the schemes proposed in the literature are not able to handle dynamic data, such as the description of the schemes [19, 20] dynamic data handling characteristic demands that data integrity should remain intact, even after insertion, deletion, or modification.

Soundness. An untrusted server cannot able to deceive a challenge request. In the schemes of Wang [21] and Zhang et al. [22], the soundness property of data integrity schemes ensures data reliability. Data integrity schemes are designed to prevent tampering. Therefore, if metadata is tampered with or corrupted intentionally or unintentionally by the CSP, this should be timely identified by a data integrity scheme. If the CSP can pass a challenge request without holding the data or with corrupted data, then a client will never be able to identify data corruption promptly, and the value of the data will be lost. Therefore, a good data integrity scheme requires that the server's response must be reliable.

Privacy Preserving. Privacy protection should be emphasized in the process of data integrity verification. As involved in the scheme [23], privacy concerns are introduced due to public verifiability. On the premise that the data owner will not allow the disclosure of his private data to a third party auditor, the privacy preservation property demands that a third party auditor should not obtain any confidential information about the user's data but can still verify the integrity of outsourced data.

Fairness. In the scheme [24], fairness means that a data integrity scheme should provide protection for an honest CSP against legitimate but dishonest users, who may attempt to accuse CSP of manipulating the outsourced data. If a data integrity scheme does not support fairness, it means dishonest users can damage CSP reputation.

Organization. The organization of the paper is as follows: the first part introduces the research status and background of cloud sharing data; the second part introduces the relevant work; the third part introduces the relevant knowledge; the fourth part describes the system model and each function of its entity, and describes the integrity audit scheme in detail; the fifth part analyses the security of the scheme, including the correctness analysis, unforgeability analysis, and proof of identity privacy by using provable security theory; the sixth part analyses the performance of the proposed scheme, including the functional comparison and efficiency analysis among different schemes. Finally, according to the advantages and disadvantages of this paper, we will formulate our next research direction.

3. Preliminaries

3.1. Bilinear Pairings. Let G_1 and G_2 be two multiplicative groups of the prime order q, and g_1, g_2 be generators of group G_1. A bilinear pairing is a map $\hat{e}$: $G_1 \times G_1 \longrightarrow G_2$ with following properties.

(1) Bilinearity

$$\begin{aligned} &\text{For } \forall g_1, g_2 \in G_1 \text{ and } a, b \in_R Z_q^*, \\ &\text{there is } \hat{e}\left(g_1^a, g_2^b\right) = \hat{e}\left(g_1, g_2\right)^{ab} \end{aligned} \tag{1}$$

(2) Nondegeneracy

$$\hat{e}\left(g_1, g_2\right) \neq 1 \tag{2}$$

(3) Computability. There is an efficient algorithm to compute this pairing.

3.2. Data Blindness. In general, the blindness of the data is that user A passes the encrypted data to user B and user B cannot infer the plaintext of user A based on these data. Therefore, users are protected as privacy. Among them, a simpler and less computational scheme is proposed in the paper, which can complete the blinding of data. The method is as follows: user A blinds the data block by using the random function and sends it to user B. User B cannot obtain the original data.

3.3. Security Theory Assumption

Definition 1 (DL problem). Unknown $a \xleftarrow{R} z_q^*$, g is the generator. Given g^a calculate α.

Definition 2 (DL assumption). The probabilistic advantage of algorithm B to solve the DL problem in probabilistic polynomial time is

$$Adv_{DL}(B) = \text{pr}\left[a \longleftarrow B\left(g, g^a\right)\right] \tag{3}$$

If $Adv_{DL}(B)$ is negligible, it is called the DL problem which is difficult.

Definition 3 (DCDH problem). Known $a, b \xleftarrow{R} z_P^*$, given $g^{1/a}$ and g^b, calculate g^{ab}.

Definition 4 (DCDH assumption). The probability that algorithm B solves the DCDH problem in probabilistic polynomial time is

$$Adv_{DCDH}(B) = \text{pr}\left[g^b \longleftarrow B\left(g, g^{ab}, g^a\right)\right. \tag{4}$$

If $Adv_{DCDH}(B)$ is ignored, it is difficult to call the DCDH problem.

3.4. Dynamic Broadcast Technology. Broadcast encryption technology is capable of transmitting encrypted information to group members over a broadcast channel. During the dissemination of this information, only members of the group can decrypt the message. Compared with traditional BE, BE can effectively support the dynamic changes of the group.

3.5. Data Sharing Integrity Verification Threat Target

Cloud Server Storage Problem. Cloud servers face the problems in situations where data is lost or data preservation is incomplete. Considering the interest of cloud service providers, to protect their reputation, they may have the potential to defraud public auditors.

Data Leakage Problem. In the process of integrity auditing performed by the third party audit, when the cloud service provider submits the certificate to the TPA for complete public verification, the cloud service provider also sends the linear combination value of the data to the third party audit. This leads to the possibility that third parties may steal content from shared data and infer the identity of the relevant user.

Data Tamper Problem. As for shared data in the cloud, team members may make malicious changes, resulting in the fact that shared data is not available. However, due to the fact that users cannot be traced back to a particular cloud, resulting in data being tampered with, so they still cannot determine the user's identity.

4. Our Construction

4.1. System Architecture

Rights Distribution Center (RDC). Figure 1 shows the cloud shared data model. In the process of data integrity verification, users, third party audit, and cloud service provider are often involved in privacy disclosure and user identity

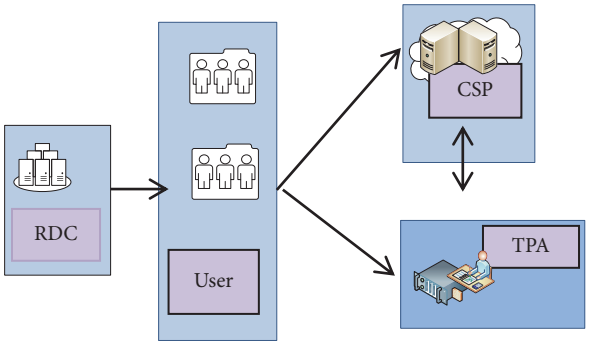


Figure 1: Cloud sharing data model.

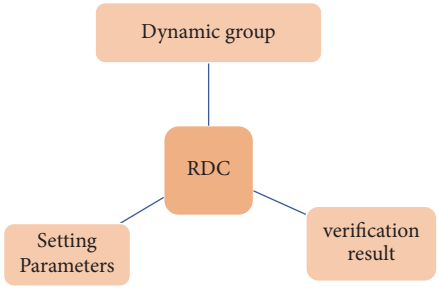


Figure 2: Right Distribution Center.

traceability issues. In this paper, by introducing the Rights Distribution Center, as shown in Figure 2 the users will be reasonably grouped and the RDC will record the operations of the data performed by the user. The RDC first performs an initialization operation to set global parameters $(G_1, G_2, \hat{e}, g, \mu, PK)$ for the system. RDC selects x as its own private key and $X_j \in Z_q^*$ as the private key of the member M_j and sets a hash function $H: Z_q^* \longrightarrow G_1$. Secondly, the RDC generates auxiliary information of the relevant data according to the (id_i, δ_i) sent by the user. The relevant information is counted in the table. Finally, when the user requests to operate on the data, the RDC will record the operation of the corresponding user to achieve identity tracing.

User. As a member of the cloud sharing data service, after registering an account, the user needs to insert, modify, and delete his or her own data. As shown in Figure 3. In the scheme, when the user sends the data to the cloud service provider, the user first performs data blinding operation on the data. On the one hand, the user blinds the data using the pseudorandom function and sends the blinded (5) to the cloud service provider.

$$m_i' = m_i + a_i \tag{5}$$

On the other hand, the user sends the tag δ_i generated by his data block to the Rights Distribution Center. Finally, the cloud user generates its own integrity verification request.

$$\sigma_i = \sigma_i' \bullet \left(\frac{PK}{g^{x_j}}\right)^{-r_j} \left(H(id_i) \bullet \mu^{\delta_i}\right)^{x_j} \tag{6}$$

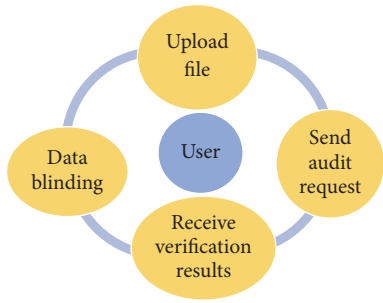


Figure 3: User.

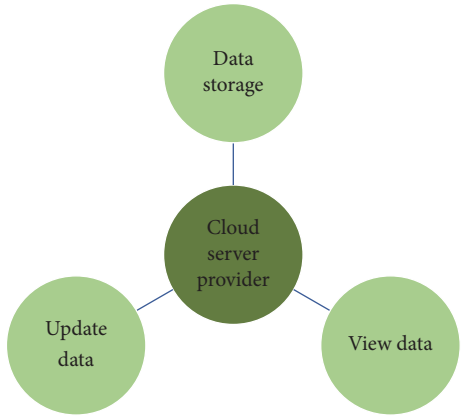


Figure 4: Cloud server provider.

According to the auxiliary information sent by RDC and its own private key, the audit request is sent to the TPA. The third party audit center verifies the integrity of the data and returns the results to the user.

Cloud Service Provider (CSP). The cloud storage service provides data owners with data storage capabilities, so that the client does not need to back up locally when using it, reducing the pressure on local storage. When the cloud service provider receives the challenge of the TPA, the cloud service provider generates evidence to indicate the integrity of the data and sends it to the TPA based on the stored file. According to the proposed scheme, on the one hand, the cloud service provider processes the data sent by the user and obtains the original data through processing. It will use the pseud random key π_k to get the original data m_i and store the data in the next step. On the other hand, according to the challenge sent by the TPA, the cloud service provider calculates the δ_i corresponding to m_i. It calculates the linear combination value u of the sample block, and sends proof = (σ, u) to TPA, from which the TPA detects whether the data is complete. Figure 4 provides a brief description of the cloud service provider.

Third Party Audit (TPA). In Figure 5, when receiving a user's audit request, the TPA first sends a challenge to the cloud service provider and then verifies the data based on the evidence returned by the cloud service provider to determine

Table 1: The meaning of the notation.

Notation	Meaning	Notation	Meaning
$\hat{e}$	A bilinear pairing map	name	Representative data identifier
H	A hash function	id_i	The identity of the user
G_1 G_2	Multiplicative groups with order q	m_i'	The blinded i_{th} block
Z_q^*	A prime field with nonzero elements	v_i	Random select from Z_q^*
F	The data file shared in the cloud	x_j	The group member's secret key
m_i	The i_{th} block of the shared cloud data file,that is $F=\{m_1,m_2,m_3\cdots\cdots m_n\}$	x_j'	The group member's partial secret key
i	Represents data block number	X	The RDC's secret key
chal	The challenge message sent to CSP by the TPA	proof	The proof message sent to TPA by the CSP
audit request	The audit message sent to TPA by the RDC	TPA	Third party audit
RDC	Rights distribution center	User	Receive cloud service
CSP	Cloud service provider	UIT	User identity table

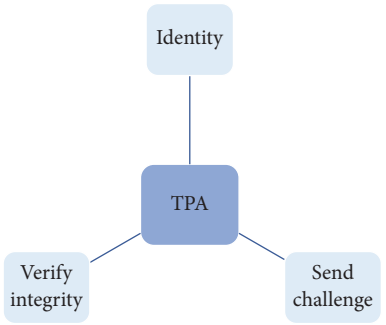


Figure 5: Third party audit.

whether the data is complete. Finally, the TPA returns the result of the integrity verification to the user. If it is complete, it returns 1; else it returns 0. In this scheme, we first initialize the user identity hash value as a reservation to the TPA. It will be used to verify the identity of the user. After the identity of the user is verified by the TPA, the TPA sends a challenge to the cloud service provider. Receiving the evidence returned by the cloud service provider, the TPA verifies whether (7) is true to judge the integrity of the data.

$$\hat{e}\left(\prod_{i\in I} H\left(id_i\right)^{v_i} \bullet u^{\mu}, PK\right) = \hat{e}\left(g,\sigma\right) \tag{7}$$

4.2. The Proposed Scheme. To verify the integrity for shared data efficiently [15, 25–30], our scheme is designed to achieve the following goals.

Cloud Data Privacy. In our scheme, we need to make sure that TPA does not know the real data from the user. At the same time, it cannot get the content of the real data from the cloud response in the audit phase.

Audit Soundness. When the cloud stores the data intact, the cloud server can be validated by TPA.

Identity Privacy. TPA cannot determine which user sent the audit request during the validation of data integrity.

The cloud sharing model mentioned in this paper includes RDC, CSP, TPA, and Client. In the following introduction, the relevant notations are shown in the Table 1. The details of the algorithm are shown in Figure 6.

(1) Setup. User can be expressed as M_j(j=1, 2...s) in the scheme. The initialization work is completed by RDC. RDC generates two multiplicative groups G_1,G_2,

$$\hat{e}\colon G_1 \times G_1 \longrightarrow G_2 \tag{8}$$

RDC selects two independent generators $g, \mu \in G_1$, chooses a hash function $H:Z_q^* \longrightarrow G_1$, and calculates

$$\mathrm{PK} = g^x \tag{9}$$

RDC selects $X_j \in Z_q^*$ as the private key of member M_j and selects x as its own private key. Select r_j and calculate g^{r_j}. So the public parameters are $(G_1, G_2, \hat{e}, g, \mu, \mathrm{PK})$. RDC distributes the private key to user.

(2) Encryption. The user selects the file and divides the file into blocks $M = \{m_1, m_2, m_3 \cdots\cdots m_n\}$. The user's identity can be identified as id_i. For each file block we can make the following operations. First, the file is blinded; we blind the data by using pseudo-random functions. We use $a_i = f_{\pi_K}(i, name)$. Each blinded file block is $m_i' = m_i + a_i$. Second the user generates a file label for each file block by using a short signature Tag_{m_i}. For convenience, we use δ_i to represent Tag_{m_i}. On the one hand, the user sends (m_i', π_k) to the CSP. On the other hand, the user sends (id_i, δ_i) to RDC. Once RDC receives the user's (id_i, δ_i), it will generate user's identity table referred to as UIT, which is shown in Table 2. RDC chooses x as its own private key and calculates

$$x_j' = x - x_j \quad \left(j = 1, 2, \cdots s\right) \tag{10}$$

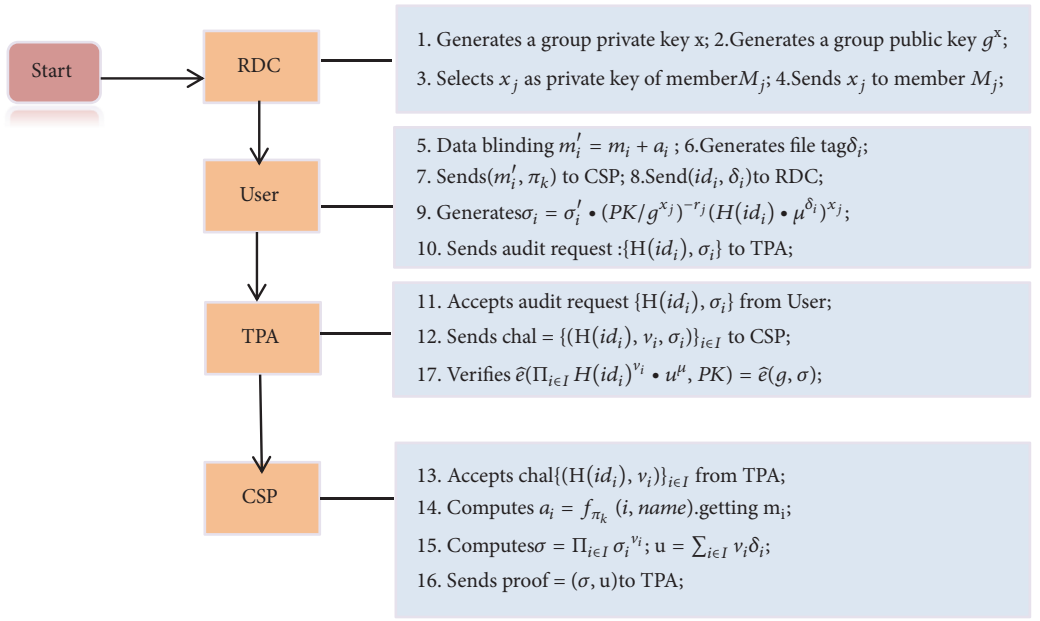


Figure 6: The relationships and interaction order of four entities.

Table 2: User identification table.

No	id	tag	Identity hiding	Member private key
1	id_1	δ_1	$H(id_1)$	x_1
2	id_2	δ_2	$H(id_2)$	x_2
3	id_3	δ_3	$H(id_3)$	x_3
…	…	…	…	…
n	id_n	δ_i	$H(id_n)$	x_j

Table 3: Stores related user hash values.

No	The hash of the user's identity
1	$H(id_1)$
2	$H(id_2)$
3	$H(id_3)$
…	…
n	$H(id_n)$

Using δ_i and x_j', RDC calculates σ_i'.

$$\sigma_i' = \left(H(id_i) \bullet \mu^{\delta_i} g^{r_j}\right)^{x_j'} \tag{11}$$

Send σ_i' to the user. The RDC sends the hash value to the TPA. As shown in Table 3, TPA keeps a copy of the legal user's identity table.

(3) Audit Request. User calculates σ_i by σ_i'.

$$\sigma_i = \sigma_i' \bullet \left(\frac{PK}{g^{x_j}}\right)^{-r_j} \left(H(id_i) \bullet \mu^{\delta_i}\right)^{x_j} \tag{12}$$

Send an audit request $\{H(id_i), \sigma_i\}$ to the TPA.

(4) Send Challenge. TPA receives and uses the look up table to determine whether it is a valid identity. If it is an invalid user, the result is returned to user. If it is a legitimate user, the TPA sends the corresponding challenge to the cloud service provider.

The TPA randomly selects $v_i \in z_q^*$ and sends (13) to CSP.

$$\text{chal} = \left\{\left(H(id_i), v_i, \sigma_i\right)\right\}_{i\in I} \tag{13}$$

The cloud server uses the pseudorandom key π_k to compute

$$a_i = f_{\pi_k}(i, name) \tag{14}$$

thus restoring the original data m_i. According to a random value, calculating the m_i corresponding δ_i by the CSP.CSP aggregates

$$\sigma = \prod_{i\in I} \sigma_i^{v_i} \tag{15}$$

and calculates a linear combination of sampling blocks

$$u = \sum_{i\in I} v_i\delta_i \tag{16}$$

Then the CSP sends proof= (σ, u) to the TPA as an evidence of whether the data is complete.

(5) Verify. TPA receives proof= (σ, u) and verifies whether the equation is true. If the equation is satisfied, it means the data is complete, and then the TPA returns 1; else it returns 0.

(6) Members Join or Remove. When a member joins, the new user needs to register the corresponding account firstly and sends his identity to the RDC. RDC will redistribute the key

to the user. The user who gets the key will have the same rights as other users and he can perform data processing on the shared data. At the same time, RDC will also add this new user in the user identification table. When a user wants to leave the group, or if some malicious users are removed forcibly, RDC will mark the user's key as a special treatment. When a user with the same key logs in again, the user can no longer continue to view and modify the data.

(7) Members Modify Data and Achieve Identity Tracking. When the user wants to modify his own data, the user needs to send a request to the CSP. After the CSP authenticates, the CSP immediately informs the RDC and the RDC will use the dynamic broadcast list to broadcast in the group where the user is located. They can receive information about the data change. If there is no objection, the RDC will record the identity of the member. And the CSP will rereceive the user's modified data. When there is an argument about the operation of the data block m_i, the RDC can find the dishonest member by looking up the operation of the relevant user. The RDC finds the corresponding element by looking up the list (id_i, δ_i). Finally, it finds the cloud user M_j.

5. Security Analysis

In this section, we will prove the correctness, unforgeability, identity privacy protection, data confidentiality, and identity traceability of the scheme in detail. By certification we can make a conclusion that the proposed scheme has high security.

5.1. Correctness Analysis. In this paper, the correctness firstly means that a cloud user uploads data to a cloud server, after receiving permission from the RDC. We do this by applying for authentication. Only legitimate users can apply for this right. Malicious users are flagged and locked in time.

Secondly, correctness means that after a cloud user obtains reasonable authority and sends an audit request to the TPA, the TPA receives the evidence sent by the cloud service provider to perform data integrity audit. Therefore, the correctness of the scheme is that TPA can complete the integrity verification through the evidence provided by the cloud service provider, thus giving the cloud user an accurate answer to the data integrity audit. If the data is complete, the result is 1 and if the data is incomplete, 0 is returned. Now it is proved in detail as follows.

We can prove that the validation results are correct; that is, the left side of the equation equals the right.

$$\widehat{e}\left(\prod_{i\in I} H\left(id_i\right)^{v_i} \bullet u^{\mu}, PK\right) = \widehat{e}\left(g, \sigma\right) \tag{17}$$

Firstly, we simplify the equation

$$\begin{aligned}\sigma_i &= \sigma_i' \bullet \left(\frac{PK}{g^{x_j}}\right)^{-r_j} \left(H\left(id_i\right) \bullet \mu^{\delta_i}\right)^{x_j} \\ &= \left(H\left(id_i\right) \bullet \mu^{\delta_i} \bullet g^{r_j}\right)^{x_j'} \\ &= \left(H\left(id_i\right) \bullet \mu^{\delta_i} \bullet g^{r_j}\right)^{x_j'} \bullet \left(\frac{g^x}{g^{x_j}}\right)^{-r_j} \\ &\quad \bullet \left(H\left(id_i\right) \bullet \mu^{\delta_i}\right)^{x_j} = \left(\mathrm{H}\left(id_i\right) \bullet \mu^{\delta_i}\right)^{x}\end{aligned} \tag{18}$$

Secondly, we calculate

$$\begin{aligned}\widehat{e}\left(g, \sigma\right) &= \widehat{e}\left(g, \prod_{i\in I} \sigma_i^{v_i}\right) \\ &= \widehat{e}\left(g, \prod_{i\in I} \left(\left(\mathrm{H}\left(id_i\right) \bullet \mu^{\delta_i}\right)^{x}\right)^{v_i}\right) \\ &= \widehat{e}\left(g, \prod_{i\in I} \sigma_i^{v_i}\right) \\ &= \widehat{e}\left(g, \prod_{i\in I} \left(\left(\mathrm{H}\left(id_i\right) \bullet \mu^{\delta_i}\right)^{x}\right)^{v_i}\right) \\ &= \widehat{e}\left(g^x, \prod_{i\in I} \left(H\left(id_i\right)^{v_i} \bullet \mu^{\sum_{i\epsilon I} \delta_i v_i}\right)\right) \\ &= \widehat{e}\left(\prod_{i\in I} \left(H\ \left(id_i\right)^{v_i} \bullet \mu^{u}, PK\right)\right)\end{aligned} \tag{19}$$

The proof is over, so we can know that when the cloud server can save the data correctly, we can verify the integrity of the data through the evidence sent by the cloud service provider.

5.2. Unforgeability Analysis. Based on the security definition based on the discrete logarithm problem, we assume that there are malicious attackers who can falsify evidence and successfully authenticate with a third party. There must be an algorithm that solves the difficult problem of discrete logarithms based on the probability of nonnegligible. In order to complete the statement that the evidence in the scheme is not falsified now, we make the following game.

Game. We assume that there is shared data M. When a third party audit sends a challenge to the cloud service provider, challenge is $\{id_i, v_i\}$. The evidence generated by the original data is (σ, u) when the cloud based on the data M' $(\mathrm{M} \neq M')$; the service provider assumes that the evidence it generates is (σ, u'), and we specify $\mathrm{u} \neq u'$. If the TPA passes the integrity verification, then we say that cloud service providers have won this game.

When the cloud service provider wins the game, we can get the two TPA equations for verifying the data's integrity:

$$\widehat{e}\left(g, \sigma\right) = \widehat{e}\left(\prod_{i\in I} \left(H\ \left(id_i\right)^{v_i} \bullet \mu^{u}, PK\right)\right) \tag{20}$$

$$\widehat{e}\left(g, \sigma\right) = \widehat{e}\left(\prod_{i\in I} \left(H\ \left(id_i\right)^{v_i} \bullet \mu^{u'}, PK\right)\right) \tag{21}$$

Through the above two formulas, we know that g, μ are generators of G_1. And we know that PK=g^x, so PK is also a generator of group G_1. By applying the relevant properties of the bilinear map, we can infer the following equations:

$$\mu^u = \mu^{u'} \tag{22}$$

$$\mu = g^{r_1} PK^{r_2} \quad \left(r_1, r_2 \text{ from } Z_q.\right) \tag{23}$$

$$\mu^{\Delta u} = \left(g^{r_1} PK^{r_2}\right)^{\Delta u} = 1 \tag{24}$$

From the above three equations, we can infer that

$$PK = g^x = g^{-r_1 \Delta u / r_2 \Delta u} \tag{25}$$

From this, we can conclude that

$$x = \frac{-r_1 \Delta u}{r_2 \Delta u} \tag{26}$$

By observing the above formula, we find that the value of x can be solved when this equation is established, which is known from our previous game definition that $\Delta u \neq 0$. Therefore, the equation is meaningless only when r_2 is zero. We can calculate it. The probability of finding x in the group Z_q is 1-1/q. Since q is a large prime number, the probability of 1-1/q cannot be ignored. That is, when the cloud service provider wins this *g*ame, we can solve the problem of discrete logarithm with a nonnegligible advantage. This is contrary to the difficulty of discrete logarithm. Therefore, cloud service providers mentioned in the scheme can only pass the verification of the TPA if they provide the correct evidence, which illustrates that the proposed scheme has unforgeability.

5.3. Identity Privacy. As described in this scheme, the user's identity privacy means that when the TPA receives the audit request sent by the user, it cannot obtain the identity of the user from the audit request.

When we perform data integrity verification, we should pay attention to the protection of user's identity privacy. In the process of integrity auditing by a third party audit, the identity verification process hides the identity of the user by exploiting the good nature of the hash function so as to better protect the user's identity privacy. Specifically, on the one hand, during the integrity audit process, when the TPA authenticates the user, it is not necessary to directly compare the user's specific id value but rather compares the hash value stored by the third party audit center with itself. If the hash value shows that the user identity exists, then the identity of the sender of the audit request can be verified, and the third party audit center can send evidence to the cloud service provider. On the other hand, TPA cannot infer relevant information about the user's identity based on the audit request sent by the user.

5.4. Data Privacy. In the scheme proposed of this paper, the privacy of data refers to when a user sends a data authentication request: on the one hand, the information about user's data cannot be acquired by other parties except for the server; on the other hand, the user combines data. When the audit request is sent, the user's data information is not leaked out to the third party audit center during the processing of the audit request.

Table 4: Comparison of scheme features.

Features	Scheme [14]	Scheme [15]	Scheme [16]	Our scheme
data block identity privacy	√	√	√	√
Dynamic group	×	×	√	√
Identity tracking	×	×	×	√
Cloud user identity privacy	×	√	×	√

6. Results and Discussion

6.1. Algorithm Function Analysis. In cloud computing, data is usually shared by several users. Through comparative analysis of different schemes, as shown in Table 4, we can compare and analyze the different functions involved in the scheme, including identity tracking, data block privacy, dynamic groups, and identity privacy. Therefore, on the one hand, we can have a basic understanding of our scheme's function. On the other hand, we can better conduct the next step of research by comparing different schemes.

6.2. Algorithm Performance Analysis. In this section, we performed the following experiment. Based on these functions, we designed several experiments to assess the workload of involved entities. These experiments are carried out on a server running Linux OS with an Intel Pentium processor of 2.70GHZ and 4GB memory.

In terms of audit generation time efficiency, we evaluated the authentication algorithm. In terms of running time, we compared the efficiency of the three schemes (Yang [31], Ateniese G [32], and Wang [14]). The experimental results are shown in Figures 7 and 8. Our signature scheme is based on the BLS signature scheme and it is similar to the Yang [31] scheme. The scheme of Ateniese G [32] is based on proxy resignature. The computational cost is mainly the resignature of the data block and the modular exponent calculation on the G_1 group. The scheme of Wang [14] is based on RSA signatures. Its computational complexity is similar to that of ring signatures, and the amount of computation is also huge. It can be seen from the figure that Ateniese G [32] and Wang [14] are very time consuming, so our scheme has advantages.

We compare the time-consuming calculation with the number of other challenge blocks. The running time is shown in Figure 9. We can see the calculations of the three schemes, our scheme, Dongare D [33], and Yuan J and Yu [34]. The amount of computation for the three schemes is linear

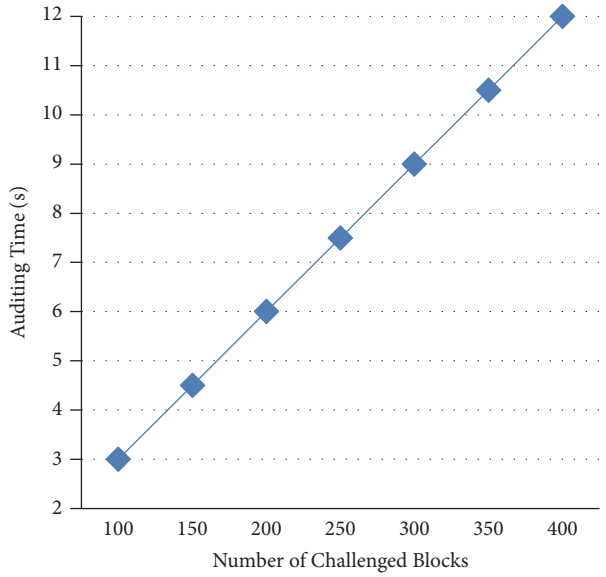


Figure 7: Audit request time.

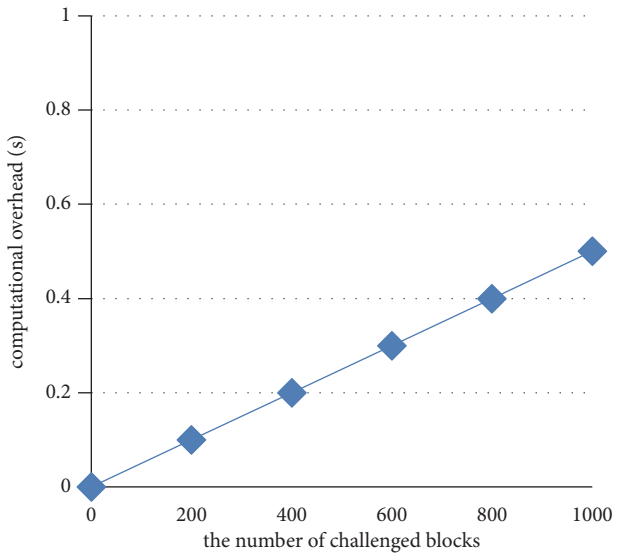


Figure 8: Challenge audit time.

with the number of data blocks being challenged increasing or decreasing. The more data blocks are challenged, the more time it takes to calculate. In the same experimental environment, our scheme spends less time than Yuan. J's scheme in calculating time. It takes more time than the Dongare D's scheme. However, this scheme can only achieve identity privacy; it cannot implement identity traceability. In terms of feasibility, our scheme has more obvious advantages. Specifically, generating a challenge message that specifies 400 random blocks takes only about 20 milliseconds, while the time specified as 1000 blocks increases to 50 milliseconds. The scheme meets the current mainstream cloud server configuration and it has strong feasibility.

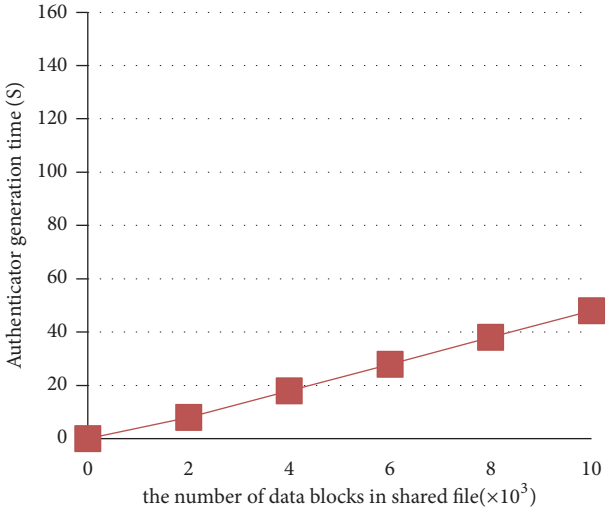


Figure 9: Authenticator generation time.

7. Conclusions

According to the above analysis, we can see that our proposed scheme is able to realize the desired security goals. In this paper, we establish a data sharing framework in cloud environment and propose a public auditing scheme with identity privacy and identity traceability for group members. The proposed auditing scheme achieves the security requirements that a well-constructed auditing scheme for shared cloud data should satisfy. As far as future work is concerned, we will continue to study how to improve the allocation of rights in the data integrity audit process and how to improve the security level of user data and protect identity privacy. The above will be the focus of our next research.

Data Availability

The data source of this paper is true and reliable. The relevant code link in this paper is https://github.com/xiaofeixue123/Integrity-audit.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61602287 and 61672330), Primary Research & Development Plan of Shandong Province (no. 2018GGX101037), and Major Scientific and Technological Innovation Project of Shandong Province (no. 2018CXGC0702).

References

[1] J. Lee, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2013.

[2] C. Erway, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09)*, pp. 213–222, ACM, Chicago, Ill, USA, November 2009.

[3] B. Wang, B. Li, and H. Li, "Knox: privacy-preserving auditing for shared data with large groups in the cloud," in *Proceedings of the International Conference on Applied Cryptography and Network Security*, pp. 507–525, Springer-Verlag, 2012.

[4] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.

[5] Q.-A. Wang, C. Wang, K. Ren, W.-J. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 847–859, 2011.

[6] C. Wang, Q. Wang, K. Ren et al., "Privacy-preserving public auditing for data storage security in cloud computing," in *Proceedings of the IEEE INFO-COM*, vol. 62, pp. 525–533, IEEE, San Diego, Calif, USA, March 2010.

[7] L. Chen, "Using algebraic signatures to check data possession in cloud storage," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1709–1715, 2013.

[8] Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren, "Enabling efficient user revocation in identity-based cloud storage auditing for shared big data," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[9] J. Yu and H. Wang, "Strong key-exposure resilient auditing for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1931–1940, 2017.

[10] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Proceedings of the IEEE INFOCOM*, pp. 534–542, IEEE, Piscataway, NJ, USA, March 2010.

[11] B. Liang, J. Y. Cao, Y. Z. Qin et al., "Survey of proofs on data storage security in cloud computing," *Application Research of Computers*, vol. 29, no. 7, pp. 2416–2421, 2012.

[12] G. Z. Qin, K. S. Wu, and H. Xiong, "A review on data integrity auditing protocols foe data storage in cloud computing," *Net Info Security*, pp. 1–6, 2014.

[13] S. H. Wang, D. W. Chen, Z. W. Wang et al., "A new solution of privacy preserving public auditing scheme foe cloud storage security," *Telecommunications Science*, vol. 28, no. 9, pp. 15–21, 2012.

[14] B. Wang, B. Li, and H. Li, "Oruta: privacy-preserving public auditing for shared data in the cloud," in *Proceedings of the IEEE 5th International Conference on Cloud Computing (CLOUD '12)*, pp. 295–302, IEEE Computer Society, June 2012.

[15] W. Shen, J. Yu, H. Xia, H. Zhang, X. Lu, and R. Hao, "Lightweight and privacy-preserving secure cloud auditing scheme for group users via the third party medium," *Journal of Network and Computer Applications*, vol. 82, pp. 56–64, 2017.

[16] B. Wang, H. Li, and M. Li, "Privacy-preserving public auditing for shared cloud data supporting group dynamics," in *Proceedings of the ICC 2013 - 2013 IEEE International Conference on Communications*, pp. 1946–1950, Budapest, Hungary, June 2013.

[17] G. Ateniese, S. Kamara, and J. Katz, "Proofs of storage from homomorphic identification protocols," in *Proceedings of International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, pp. 319–333, Springer-Verlag, London, UK, 2009.

[18] W. Luo and G. Bai, "Ensuring the data integrity in cloud data storage," in *Proceedings of the 2011 IEEE International Conference on Cloud Computing and Intelligence Systems, CCIS2011*, pp. 240–243, IEEE, China, September 2011.

[19] R. Curtmola, O. Khan, and R. Burns, "Robust remote data checking," in *Proceedings of the 4th ACM International Workshop*, pp. 63–68, ACM, Alexandria, Va, USA, October 2008.

[20] Y. Zhang and M. Blanton, "Efficient dynamic provable possession of remote data via balanced update trees," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security, ASIA CCS 2013*, pp. 183–194, China, May 2013.

[21] H. Wang and Y. Zhang, "On the knowledge soundness of a cooperative provable data possession scheme in multicloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 264–267, 2014.

[22] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 12, pp. 2231–2244, 2012.

[23] Y. Zhu, H. Wang, Z. Hu, G. Ahn, H. Hu, and S. S. Yau, "Efficient provable data possession for hybrid clouds," in *Proceedings of the 17th ACM Conference*, pp. 756–758, ACM, Chicago, Ill, USA, October 2010.

[24] Y. Zhu, H. Hu, G. Ahn, Y. Han, and S. Chen, "Collaborative integrity verification in hybrid clouds," in *Proceedings of the 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pp. 191–200, Miami, Fla, USA, October 2011.

[25] F. Zafar, A. Khan, U. R. S. Malik et al., "A survey of cloud computing data integrity schemes: design challenges, taxonomy and future trends," *Computers & Security*, 2016.

[26] K. Han, Q. Li, and Z. Deng, "Security and efficiency data sharing scheme for cloud storage," *Chaos Solitons & Fractals the Interdisciplinary Journal of Nonlinear Science & Nonequilibrium & Complex Phenomena*, vol. 86, pp. 107–116, 2016.

[27] T. Jiang, X. Chen, and J. Ma, "Public integrity auditing for shared dynamic cloud data with group user revocation," *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2363–2373, 2016.

[28] L. Xue, J. Ni, Y. Li, and J. Shen, "Provable data transfer from provable data possession and deletion in cloud storage," *Computer Standards & Interfaces*, vol. 54, pp. 46–54, 2017.

[29] D. Sánchez and M. Batet, "Privacy-preserving data outsourcing in the cloud via semantic data splitting," *Computer Communications*, vol. 110, pp. 187–201, 2017.

[30] Y. Yu, J. Ni, W. Wu et al., "Provable data possession supporting secure data transfer for cloud storage," in *Proceedings of the 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, pp. 38–42, IEEE, Krakow, Poland, 2016.

[31] G. Yang, J. Yu, W. Shen, Q. Su, Z. Fu, and R. Hao, "Enabling public auditing for shared data in cloud storage supporting identity privacy and traceability," *The Journal of Systems and Software*, vol. 113, pp. 130–139, 2016.

[32] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp. 598–609, ACM, Virginia, Va, USA, November 2007.

[33] D. Dongare and V. Kadroli, "Panda: Public auditing for shared data with efficient user revocation in the cloud," in *Proceedings of the 2016 Online International Conference on Green*

Engineering and Technologies (IC-GET), pp. 2904–2912, IEEE, Coimbatore, India, November 2016.

[34] J. Yuan and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification," in *Proceedings of the 33rd IEEE Conference on Computer Communications, IEEE INFOCOM 2014*, pp. 2121–2129, Canada, May 2014.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 6417407, 9 pages
https://doi.org/10.1155/2019/6417407

Research Article

Multifeature Named Entity Recognition in Information Security Based on Adversarial Learning

Han Zhang,[1,2] **Yuanbo Guo**,[1] **and Tao Li**[1]

[1]*Information Engineering University, Zhengzhou 450001, China*
[2]*Zhengzhou University, Zhengzhou 450000, China*

Correspondence should be addressed to Han Zhang; zhang_han@zzu.edu.cn

Received 5 November 2018; Accepted 6 February 2019; Published 24 February 2019

Guest Editor: Pelin Angin

In order to obtain high quality and large-scale labelled data for information security research, we propose a new approach that combines a generative adversarial network with the BiLSTM-Attention-CRF model to obtain labelled data from crowd annotations. We use the generative adversarial network to find common features in crowd annotations and then consider them in conjunction with the domain dictionary feature and sentence dependency feature as additional features to be introduced into the BiLSTM-Attention-CRF model, which is then used to carry out named entity recognition in crowdsourcing. Finally, we create a dataset to evaluate our models using information security data. The experimental results show that our model has better performance than the other baseline models.

1. Introduction

Named entity recognition (NER) aims to extract various types of entities from text. This is a fundamental step in text mining and has received much attention recently, especially in medicine [1–6] and biochemistry [7–10]. In contrast, the development of NER tasks in information security has been relatively slow. In previous works, several methods have been proposed for extracting vulnerabilities and extracting information from unstructured texts [11–14]. In the past two years, research into NER has basically entered a stagnant state in the domain of information security. The lack of large-scale labelled data in this field is one of the main reasons for this situation.

Snow et al. proposed a way to quickly and cost-effectively obtain large-scale labelled data using Amazon Mechanical Turk [15] and demonstrated that nonexpert annotations were relatively useful for training models [16]. We can use crowdsourcing as an effective way of obtaining large-scale labelled data at low cost within a short time. However, in a professional field, crowd annotations may be of lower quality than those of experts, and we therefore need to integrate high-quality consensus labelling from crowdsourcing annotations. Although we can obtain high-quality annotations using the majority vote method [17], this requires a great deal of manpower, and for some sentences or entities, whose meanings are rather ambiguous, it may be difficult to reach an agreement among the annotators. A generative adversarial network (GAN) has the ability to generate data. Goodfellow et al. proved theoretically that when the GAN model converges, the generated data have the same distribution as the real data [18]. Yang et al. demonstrated the usability of the GAN model for NER using Chinese crowd-sourced annotations [19]. However, the focus of this work is on using the GAN model to find the features of the trust annotators, and its application is mainly in the general domain.

In practice, there is a significant difference between entity types used in general applications and those used in information security. Some entity categories in the information security domain are not simply nouns or nominal phrases, as traditionally defined in NER. For example, consider the text: [a Trojan] "known as 'Bicololo' was first discovered in October 2012 and specially designed to steal login credentials". In this sentence, the phrasal verb "steal login credentials" should be extracted as an entity of the consequence class of the unified cybersecurity ontology (UCO) [20]. Therefore, when

we identify named entities in the information security field, we need to use certain supplementary features in addition to the traditional (word and character) features.

In this paper, we propose a new model entitled BiLSTM-Attention-CRF-Crowd to improve the quality of the crowdsourcing annotations in information security. Our goal is to combine a GAN model with the BiLSTM-Attention-CRF model. The GAN model is used to find the common features of annotations in order to integrate the best unique consensus annotations and then pass them to the BiLSTM-Attention-CRF submodel as one type of additional feature. Here, we add an attention layer between the BiLSTM and CRF layers, primarily to process long sentences appearing in the text. A neural network model using a conventional encoder-decoder structure is needed to represent the information in the input sequence as a fixed-length vector; it is difficult to retain all the necessary information when the input sequence is long, especially when the length of the input sequence is longer than the length of the sequence in the training data set, and we therefore added the attention layer to address this limitation. The submodel performs NER again on the crowdsourced annotations for the information security data to improve the quality of these annotations. The main contributions of our work can be summarised as follows:

(I) In order to solve the problem of a lack of high-quality annotated data in the field of information security, a new model called BiLSTM-Attention-CRF-crowd is proposed to improve the quality of the crowdsourcing dataset by combining a GAN model with the BiLSTM-Attention-CRF model.

(II) Due to the diversity and specificity of the entity categories in information security, only basic features such as word and character features are used as input for the BiLSTM-Attention-CRF model, which cannot meet the requirements for NER tasks in this field. Based on this, domain dictionary features and sentence dependency features are introduced. These are used as additional features along with the common features learned by the GAN model. The experimental results show that these additional features have practical value for improving the performance of the model.

2. Related Works

2.1. GAN. Compared with its applications in computer vision, the GAN model is less widely used in the field of language processing because the values used in images and video are continuous, and the generator and discriminator can be directly trained using the gradient descent method; in contrast, letters and words in text are all discrete, and the gradient descent method cannot be directly applied. Zhang et al. proposed the TextGAN model, which uses several techniques to deal with discrete variables [21]. For example, it uses a smooth approximation to approximate the discrete output of the LSTM and feature matching techniques in the generator training process. Since the number of parameters in the LSTM is significantly greater than that for the CNN, the LSTM is more difficult to train, and the TextGAN discriminator (CNN) only updates once after the generator (LSTM) has been updated multiple times. Yu et al. proposed SeqGAN, which draws on the concept of reinforcement learning to deal with the discrete output problem. It regards the error in the discriminator output as the reward value in reinforcement learning and regards the training process of the generator as a decision-making process in reinforcement learning. This model is applied to speech text and music generation [22]. Li et al. and Kusner et al. applied GAN to open dialogue text generation and context-free grammar (CFG), respectively [23, 24]. We mainly draw on the method of processing of discrete variables and its objective function for feature matching in [21].

2.2. Crowdsourcing. David et al. presented a confusion matrix for each annotator, using an expectation maximisation (EM) estimation of these matrices as parameters and the true token labels as hidden variables [25]. Dredze et al. proposed a conditional random field (CRF) model for learning from multiple annotations, but the features that the CRF can learn are limited [26]. In previous work [19, 27, 28], the aim was to model the differences between the annotators and to extract the more trustworthy annotators. Although this improves the performance of the model, this choice of annotation is too dependent on the credibility of an annotator.

2.3. NER in Information Security. In order to solve the problem of a lack of large-scale labelled data, Rodrigo Agerri et al. designed a projection algorithm to transport NER annotations across languages [29]. However, in information security, there are no scaled annotations sets in any language. Giorgi et al. combined gold-standard corpora with silver-standard corpora by using transfer learning to expand the scale of high-quality labels [30]. However, this is applied in the field of biology.

Few documents from the last three years can be found on the subject of NER in information security. A small number of works have focused on extracting vulnerabilities and attack information from unstructured texts in the past few years [11–14]. Bridges et al. proposed a maximum entropy model trained with an averaged perceptron to extract entities from text, but these authors extracted only the entities and did not classify the types of these entities [11]. Weerawardhana et al. extracted vulnerability information from an online vulnerability database [12]. Lal proposed a CRF to extract vulnerabilities from the text [13]. Mulwad et al. used wikitology to extract vulnerabilities and attacks from web text, although wikitology is an ontology in the general domain [14].

3. Feature Selection

A high-quality feature set is key to the success of NER tasks in information security. In this paper, we use word and character features as our basic feature set and others (such as domain dictionary and sentence dependency features) as two kinds of additional features.

3.1. Word Features. Word features (word embedding) involve a distributed representation of a word in vector space. This can capture semantic and grammatical information about a word from an unlabeled corpus [31]. Words with similar context or semantics are closer in the word vector space, and word vectors can therefore be used for natural language processing tasks such as entity classification, entity alignment, and relation extraction. Word2vec [32] is the most commonly used tool for training word vectors. In this paper, in order to obtain high-quality word vectors, we select 94,534 vulnerability record descriptions from entries in the Common Vulnerabilities and Exposures (CVE) corpus [33] listed since 1997 for word vector training.

3.2. Character Features. Character features contain structural information about the name of the entity and can represent the specific composition of the entity's name, especially in the information security domain. For example, viruses such as Backdoor.Win32.Gpigeon.pd and Backdoor.Win32.Gpigeon2010.pc, which are PE viruses affecting Windows, have the same prefix; hence, when we encounter one these words, we know that it is the name of a PE virus for windows, based on its prefix. Unlike traditional manually designed character features, we can obtain the character feature vectors for words through training. First, the character vector of each character in the word is obtained by querying the character table, and then the character vector corresponding to the word is used as input for the BiLSTM.

3.3. Additional Features

3.3.1. Domain Dictionary Feature. In information security, in order to better identify entities, it is not sufficient to use only word and character features; we also need to add domain knowledge, including features such as a domain dictionary. At present, there is no established dictionary for reference in information security, so we can use the Internet to construct a preliminary domain dictionary. In this paper, we use Wikipedia as a corpus and the UCO in the information security domain to construct a domain dictionary. Wikipedia contains three types of page: an entry page, no_created entry page, and a list page. Entry pages are mainly used to describe concepts. In Wikipedia, the URLs of these three types of pages follow certain rules. For example, the URL for the entry page is usually in the form http://en.wikipedia.org/wiki/∗, and the URL for the list page is usually in the form n.wikipedia.org/wiki/Category. In computer-related fields, we therefore only need to use en.wikipedia.org/wiki/Category: computing as a crawler input for concept capture. After the concept is captured, k-means clustering is carried out. There are eight important concept classes in UCO, and k can therefore be set to eight for k-means clustering.

The resulting clustering categories are labelled by hand and then reclassified using the Mahalanobis distance. The specific algorithms are as follows:

(I) We have a set of classes $\{c_{1,}c_{2,\ldots}c_k\}$ where k is the number of classes, and $\{x_a^1, x_a^2, \ldots x_a^n\}$ expresses the original concept vectors in class C_a in UCO. Then, the mean vector μ_a and covariance matrix $\sum_a \mathrm{f}\, C_a$ are as follows:

$$\mu_a = \frac{1}{n}\sum_n x_a^i$$
$$\sum_a = \frac{\sum_{i=1}^n \left(x_a^i - u_a\right)}{n-1} \tag{1}$$

(II) If w_i represents the vector of the subclass of C_a after clustering, then the semantic similarity is calculated by the Mahalanobis distance:

$$D_m\left(w_i, c_a\right) = \sqrt{\left(w_i - u_a\right)^T \sum_a^{-1} \left(w_i - u_a\right)} \tag{2}$$

The clustering result is determined again. We set the threshold y; if $D_m(w_i, c_a) \geq y$, then the concept is considered not to belong to the current concept category. The initial value of the threshold y can be obtained from the original concept under the training category.

(III) If a concept has not been classified into a category after filtering, it can be judged based on its upper and lower concept in Wikipedia. If its upper and lower concept are not in the same category at this time, then the distance between the upper (lower) concept and the class centre of the category is calculated separately. We select a category with a smaller distance as its category. In contrast, we put the concept into the class to whose upper and lower concept belong.

(IV) Before classifying a new concept into a category, the algorithm returns to Step I to recalculate the mean vector and covariance matrix for the category.

3.3.2. Sentence Dependency Feature. As discussed above, the entity types in information security are no longer simply the types defined in traditional NER tasks.

We use the example given above to analyse the annotation of the semantic role and syntactic dependencies. We use Stanford CoreNLP [34] as a syntactic analysis tool. The result of this syntactic analysis is shown in Figure 1.

The core verbs in this sentence are "discovered" and "designed", and "Bicololo" is the passive subject of these two verbs. The subject of this sentence is a worm virus from the attacker class. If we want to extract the entity of the consequence class, we should focus on the modifier followed by "designed", for which the phrase "steal login credentials" is the open clausal complement. The open clausal complement is a verb or a verb phrase which is used to add a description of the core verb. At this point, we can create the following rules for extracting consequence class entities:

(I) The subject of the sentence should be the type of attacker entity.

(II) We consider the verbs associated with each attacker entity, such as "be designed to/for", "be used to",

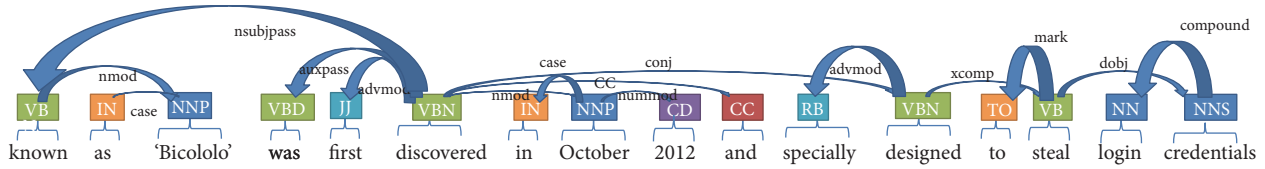


Figure 1: Sentence dependency analysis.

"result", "cause", and other predicate verbs and extract the minimum verb phrases or clauses associated with these. Here, the minimum verb phrase or clause is a phrase or clause that does not contain any nested or identical type.

(III) If the relationship between the minimum phrase or clause and the foregoing predicate verbs is that of a complement or modifier, the minimum phrase or clause can be considered as an entity of the consequence class.

4. Model Design for BiLSTM-Attention-CRF-Crowd

Our model focuses on two tasks. Task 1 is the generation of crowd annotations through adversarial learning in order to integrate the optimal single-consensus annotations. In Task 2, the common features generated in Task 1 are used in conjunction with the domain dictionary features and the sentence dependency features and are input to the BiLSTM-Attention-CRF submodel for renamed entity recognition of the crowdsourcing annotations in order to improve their quality. The architecture of our model is illustrated in Figure 2.

The model consists of two submodels. Figure 2(a) shows the GAN model, which consists of the generator BiLSTM-Attention and the discriminator CNN. The crowd annotations are used as input for the generator to form new feature representations after being processed by the BiLSTM-Attention layer. The new feature representations are passed to the CNN, which is trained using the expert annotations, and the CNN determines whether the distributions of the new features from the crowd annotations and the expert annotations are consistent. If so, the new features can be passed as one of the additional features to the model shown in Figure 2(b); otherwise, the new features are passed back to the BiLSTM layer. Figure 2(b) shows the BiLSTM-Attention-CRF submodel for NER in the crowdsourced domain dataset. Several features, such as the dictionary, sentence dependency, and common features of the crowdsourced annotations are input to the model to improve the quality of the crowdsourced annotations.

4.1. Adversarial Learning for Common Features. The structure of the GAN contains two models: a generative network and discriminative model. The goal is to learn a generative distribution that matches the real data distribution. More

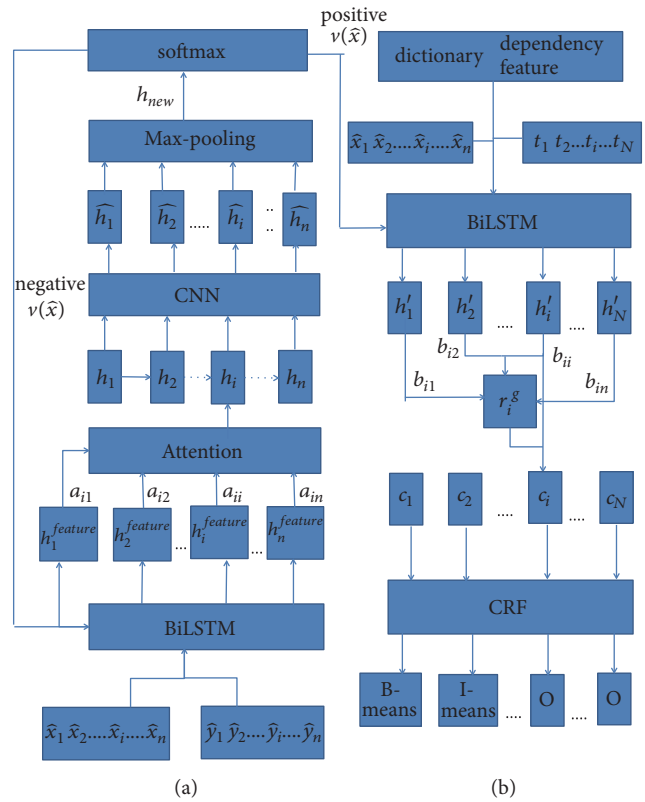


Figure 2: Diagram of the proposed model.

specifically, the generative network generates samples from the generator distribution, and the discriminative model learns to determine whether a sample is from a generative distribution or a real data distribution.

Adversarial learning is used to find the common features of the crowd annotations. The discriminative model is trained using expert annotations and crowd annotations, which are used as the input for the generated model, and the generated feature distribution is passed to the discriminative model and used to determine the similarities and differences in the feature distributions of the crowd and expert annotations. The model is repeatedly trained until the discriminator can no longer distinguish a difference between them. At this point, the result of the model is the set of common features of the crowd annotations, which is also the set of optimal single-consensus annotations that we want to integrate. As shown in Figure 2(a), the model consists of

two parts: the generation model composed of BiLSTM-Attention and the discriminant model composed of the CNN. The max-pooling layer and the softmax layer in the CNN optimise the feature map generated by the CNN layer and normalise the selected new features to determine whether they are consistent with the expert annotations feature distribution.

4.1.1. BiLSTM-Attention Model. We consider a sentence $s = \{w_1, w_2, \ldots, w_n\}$, where n is the number of the words in the sentence. Given the crowd-annotated NE label sequence $y = \{y_1, y_2, \ldots, y_n\}$, where y_i represents the corresponding annotation of the named entity w_i, we use $\widehat{x_i}$ and $\widehat{y_i}$ as the word vectors and the annotation vectors of word w_i, which are trained using word2vec, and pass them as input to the generative model BiLSTM to get the features $h_i^{feature}$, where

$$h_i^{feature} = BiLSTM\left(\widehat{x}_i, \widehat{y}_i\right) \tag{3}$$

In order to obtain more important features, a new attention layer is used above the BiLSTM layer to capture a new representation of the feature h_i:

$$\begin{aligned} f\left(h_i, h_j\right) &= \left(h_i^{feature}\right)^T w_a h_j^{feature} \\ a_{ij} &= soft\max\left(f\left(h_{i-1}, h_j^{feature}\right)\right) \\ h_i &= \sum_j^n a_{ij} h_j^{feature} \end{aligned} \tag{4}$$

where w_a is the model parameter.

4.1.2. CNN. Following this, we add a CNN module based on the outputs of the BiLSTM-Attention model, to determine the similarities and differences between the feature distributions for the crowd and expert annotations. A convolutional operator with a window size of five is used, and then a max-pooling strategy is applied to the convolution sequence to obtain the final fixed-dimensional feature vector. The overall process can be described by the following equations:

$$\widehat{h_i} = \tanh\left(W^c\left[h_{i-2}, h_{i-1}, \ldots h_{i+2}\right]\right) \tag{5}$$

$$h_{new} = \max\left\{\widehat{h_1}, \widehat{h_2} \ldots \widehat{h_n}\right\} \tag{6}$$

where W^c is the CNN model parameters, and the activation function tanh is used primarily to normalise and prevent the loss of features. In the pooling method, the maximum feature is the most important, as it effectively filters out word combinations with less information and can ensure that the extracted features are independent of the length of the input sentence.

The feature h_{new} is then mapped to the output $\mathrm{D}(h_{new}) \in [0, 1]$ using a softmax function to determine whether the input feature is consistent with the feature distribution of the expert annotations.

4.1.3. Objective Function. We use the feature matching method in [21], set S as the expert annotations and use iterative optimisation schemes consisting of two steps:

$$\begin{aligned} &minimizing: L_D \\ &= -E_{s:S}\log\left(D\left(s\right)\right) - E_{\widehat{x}:p(\widehat{x})}\left[\log 1 - D\left(G\left(\widehat{x}\right)\right)\right] \\ &minimizing: L_G \\ &= \mathrm{tr}\left(\sum_s^{-1}\sum_{\widehat{x}} + \sum_{\widehat{x}}^{-1}\sum_s\right) \\ &\quad + \left(\mu_s - \mu_{\widehat{x}}\right)^T\left(\sum_s^{-1} + \sum_{\widehat{x}}^{-1}\right)\left(\mu_s - \mu_{\widehat{x}}\right) \end{aligned} \tag{7}$$

where $\sum_{\widehat{x}}$ and $\sum_s$ represent the covariance matrices for the expert and the crowd annotation features, respectively, μ_s, $\mu_{\widehat{x}}$ denote the mean features of the expert and the crowd annotation features, respectively, and their values are empirically estimated using mini-batch. L_G represents the Jensen-Shannon divergence between two multivariate Gaussian distributions $dtri(\mu_s, \sum_s)$ and $dtri\ (\mu_{\widehat{x}}, \sum_{\widehat{x}})$. The main purpose of this is to provide a stronger signal for modifying the generation model in order to make the feature distribution generated by the generated model more similar to that of the discriminant model [21].

In training the generation model, which contains discrete variables, the direct application of gradient estimation would fail. Thus, we draw on the method used in [21], and use a soft-argmax function when performing the inference as an approximation to the inputs of the generated model BiLSTM:

$$v\left(\widehat{x}\right) = W_e\, soft\max\left(Vh_{new} \circ L\right) \tag{8}$$

where $\circ$ represents the element-wise product, V is a weight matrix used to calculate the word distribution, and W_e is model parameter. When $L \longrightarrow \infty$, this expression approximates the default input vector calculation formula for BiLSTM.

4.2. BiLSTM-Attention-CRF SubModel. The BiLSTM-Attention-CRF submodel adds the attention mechanism to the classical BiLSTM-CRF model to allow it to pay attention to the correlation between the current entity and the other words in the sentence and to obtain the feature representation of words at the sentence level to improve the accuracy of the model labelling. The model structure is shown in Figure 2(b).

Using the word feature $\widehat{x_i}$, the character feature t_i, and additional features corresponding to the words w_i as the input to BiLSTM, we get the new representation h_i' of the word w_i (here, the method of calculating h_i' is the same as for the $h_i^{feature}$ above), which is used as input to the attention layer. The attention weight value b_{ij} in the attention matrix is derived by comparing the current word w_i with the other words $w_j\ (j = 1, 2, \ldots i-1, i+1, \ldots n)$ in the sentence.

$$b_{ij} = \frac{\exp\left(f\left(w_i, w_j\right)\right)}{\sum_{k=1}^{n} \exp\left(f\left(w_i, w_k\right)\right)} \tag{9}$$

The method of calculation of $f(w_i, w_j)$ is shown in (2).

A sentence-level vector r_i^g is then computed as a weighted sum of each BiLSTM output h_i':

$$r_i^g = \sum_{j=1}^{n} b_{ij} h_i' \tag{10}$$

Next, we combine the sentence-level vector r_i^g with the BiLSTM output of the target word as a vector $[r_i^g, h_i']$ to be passed to the tanh function to produce the output of the attention layer.

$$c_i = \tanh\left(W_g\left[r_i^g, h_i'\right]\right) \tag{11}$$

Finally, we use c_i as the input of the upper CRF layer. Here, the CRF has two roles: the first is to calculate the score o_i for each c_i in the corresponding annotation, and the second is to use the tagging transition matrix T (to define the score of two consecutive annotations) and the Viterbi algorithm to calculate the best annotation sequence. This process is expressed as follows:

$$o_i = Wc_i \tag{12}$$

$$score\left(D, y\right) = \sum_{i=1}^{N} \left(o_{i,y_i} + T_{y_{i-1}, y_i}\right) \tag{13}$$

$$y^{result} = \text{argmax}\ \left(score\left(D, y\right)\right) \tag{14}$$

where the function $score(\,)$ is used to calculate the score of the annotation sequence $y = y_1 y_2 \ldots y_n$ of the input sentence, y^{result} is the final output annotation sequence result (i.e., BIO annotation), and W represents the model parameter.

5. Experimental Results and Analysis

To evaluate our model, we divided the baseline models into two groups based on their different uses. First, our model and the first group of baseline models were applied to the crowdsourcing annotations to verify the ability of our model to integrate consensus annotations in comparison with other baseline models. Secondly, our model and the second group of baseline models were applied to identify specific entities in information security to verify the ability of our model to identify specific types of entities. Finally we verified the effect of additional features on the performance of the model.

5.1. Data Sources. The dataset used in this experiment was mainly drawn from the field of information security, and included related blog posts (such as we live security, threatpost), CVE descriptions, Microsoft security bulletins, and information security abstracts. From this corpus, 10,187 sentences were selected (consecutive paragraphs including 20 abstracts, 45 blog posts, 59 CVE descriptions and 50 Microsoft security bulletins) and each sentence was assigned to three annotators to generate crowd annotations. These three annotators were students at the authors' institution with no educational background in information security. Each annotator only needed to annotate four types of named entities in the sentence: the product, the consequence, the attacker, and the version. Two senior students taking information security courses were asked to annotate 1,000 sentences that were randomly selected to train the discriminant model in the GAN. From the crowd annotations, we randomly chose 7,000 sentences as a training set and used the remainder as a test set.

5.2. Baseline Models. The comparison models were divided into two groups for experiments.

Group 1: to learn the common features of crowd annotations, we used the following as comparison models:

(I) Majority vote (MV)[17]

(II) Dawid and Skene Model [25].

Group 2: to predict the named entity sequence from unlabeled text, we used the following as comparison models:

(I) BiLSTM-Attention-CRF: The model in [35] was trained directly using the crowd annotations. When we used this model, we removed the part of the model that used image features.

(II) BiLSTM-Attention-CRF-VT: This was trained on the data selected from the crowd annotations by majority vote.

(III) HMM-crowd [28].

(IV) CRF-MA: From the model in [26], we used the source code provided by the author.

5.3. Settings. There are several hyperparameters in our model. We set the dimensions of the futures vector to 300, the number of units in BiLSTM to 1000, and the minibatch size to 64. The max-epoch iteration was set to 100. The method described in [36] with a learning rate of 10^{-3} was used to update the model parameters, and the l_2 regularisation was set to 10^{-5}. We adopted the dropout technique to avoid overfitting. The dropout was 0.3 for BiLSTM and 0.5 for the attention layer.

Our experiment was implemented on two NVIDIA GTX 1080Ti GPU with 64 GB memory, and the model was trained for approximately one hour.

5.4. Evaluation of Experimental Results. The indicators used in the evaluation of the experiment were the accuracy rate (P), the recall rate (R), and the F1 value.

5.4.1. Performance Comparison of Integrated Crowd Annotation Model

(I) Performance Comparison between Our Model and the First Group of Baseline Models. We use the accuracy rate of the correct annotation obtained from the training corpus used by each model as our evaluation criterion.

Table 1: Performance comparison for the models used.

Method	Accuracy rate
MV	65.3%
Dawid and Skene	72.5%
BiLSTM-Attention-CRF-crowd	78.9%

Table 2: Comprehensive performance evaluation for each model.

Method	Precision	Recall	F1
BiLSTM-Attention-CRF	88.5%	79.9%	84.4%
CRF-MA	88.4%	75.6%	81.5%
Dawid and Skene-LSTM	79.3%	75.1%	77.1%
BiLSTM-Attention-CRF-VT	75.5%	75.6%	75.6%
BiLSTM-Attention-CRF-crowd	89.1%	90.0%	89.0%

Performance comparisons for various models on the test corpus are shown in Table 1.

The performance of MV was relatively poor. This is because it is difficult to achieve uniformity for an ambiguous entity due to the professionalism of the field and the uneven distribution of the annotation level. Our model achieves the best performance. In addition to obtaining the correct annotations in the training corpus, our model also can generate a positive sample that is consistent with the feature distribution of the expert annotations through repeat training. Through the secondary extraction of the BiLSTM-Attention-CRF submodel, the accuracy rate of correct annotations in the training corpus is improved.

(II) Comparison of the Overall Performance of NER in the Information Security Field for Each Model. Table 2 shows that in terms of precision, the BiLSTM-Attention-CRF model that was directly trained using unprocessed crowd annotations had the highest precision. This means that the crowd annotations are useful for training the NER model. The BiLSTM-Attention-CRF-VT model trained on data selected using the MV method from the crowd annotations showed the poorest performance. This model may therefore not be suitable for the information security field. The reason for this may be the complexity and professionalism of the information security text statements. Many entities cannot be selected by voting. In addition, contextual information is lost by voting selection, which means important feature information is lost. The BiLSTM-Attention-CRF model, which directly uses unprocessed crowdsourcing label data as training data, does not lose important context information; however, the level of noise is increased, so its overall performance is slightly lower than that of the BiLSTM- Attention-CRF-crowd model.

5.4.2. Performance of Specific Type Entity Recognition

(I) Integrated Performance Evaluation of Specific Types of Entity Annotation. In order to verify the integration performance of our model in terms of recognising different types of entities, the integration results of the four types of entities proposed above in the crowd annotations were compared. In Figure 3, the ordinates represent the P, R, and F1 values, respectively. The labels 1, 2, 3, and 4 on the abscissa correspond to the entity type (product, attacker, consequence, and version, respectively), and 5 represents the mean of the three indicators of the model. As can be seen from Figure 3, the model performs better on Types 1 and 4, mainly because the class indicated by Type 1 is the product. In general, although this type of entity belongs to the information security domain, public awareness of this field is relatively high, meaning that the precision of the annotation is relatively high. The category corresponding to Type 4 has a relatively fixed pattern; it always appears after the product name and is usually expressed by numbers. The performance of the model is best for entities with a fixed patterns, because its features are easier to learn. For Types 2 and 3, which are relatively specific types of entity in the information security field, the precision of the crowd annotations was low. In particular, the consequence class of Type 3 has higher requirements for the professional ability of the annotator and the entity part of the category is not fixed, so the performance is slightly weaker.

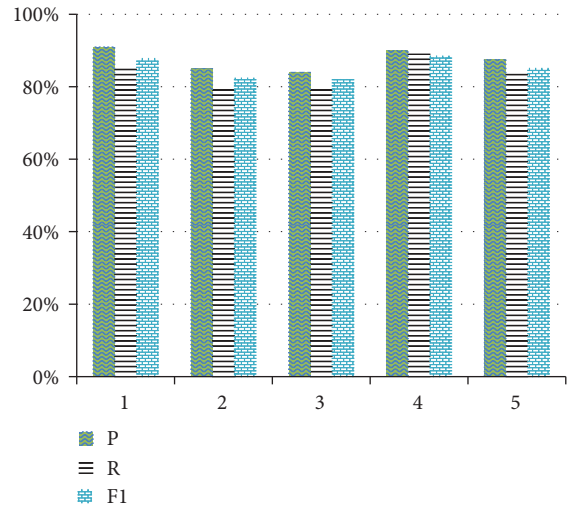


Figure 3: Performance comparison for different types of entity.

(II) Performance Evaluation of Specific Types of Entity with Other Models. In Figure 4, the ordinate represents the accuracy of each model, and the abscissa is the same as in Figure 3. These models take the basic, domain dictionary and sentence-dependency features as additional features for input. It can be seen from the figure that the performance of the BiLSTM-Attention-CRF-crowd model is better than that of the other models. However, the model also has a lower accuracy than the recognition of each type in Figure 3, which also proves that the common feature of adversarial learning has a significant effect on improving the accuracy of the model.

(III) Effect of Additional Features on the Performance of Each Model. The above four entity types are used as extraction targets, and we compare our model with itself which uses

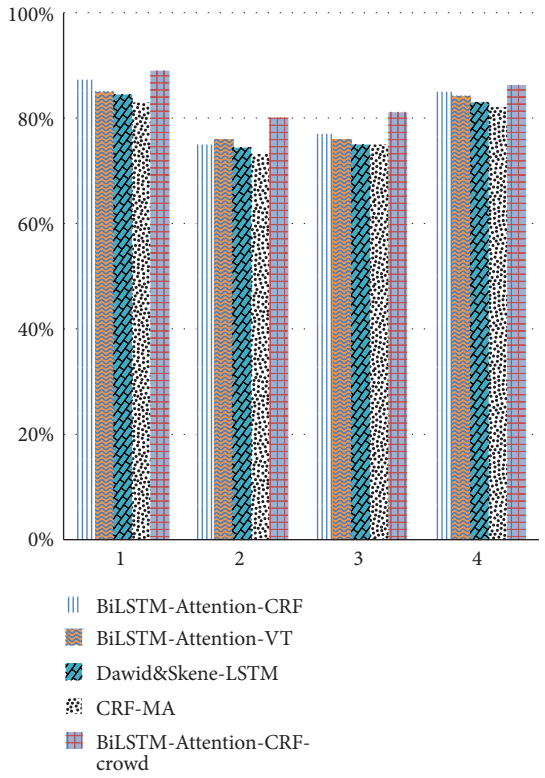


Figure 4: Performance comparison of models.

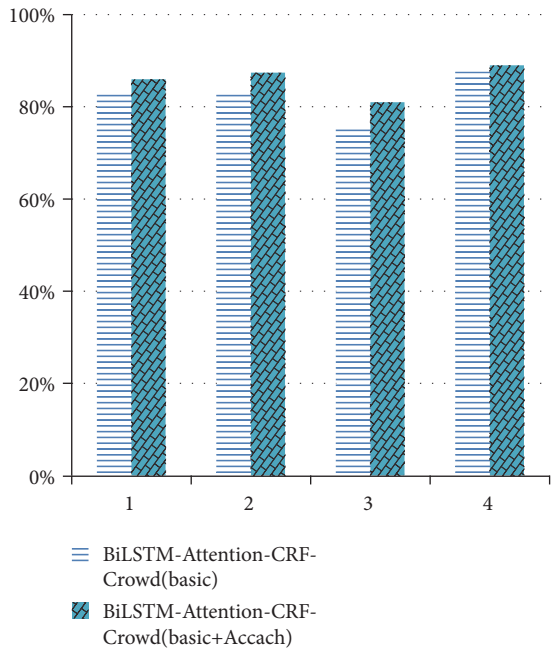


Figure 5: Effect of additional features on model performance.

word features and character features as inputs (i.e., BiLSTM-Attention-CRF-crowd (basic)) and another using basic features and additional features as inputs (BiLSTM-Attention-CRF-crowd (basic+attach)). The results for the extraction accuracy are shown in Figure 5.

It can be seen that the extraction accuracy of the BiLSTM-Attention-CRF-crowd (basic+attach) model is significantly higher than that of the BiLSTM-Attention-CRF-crowd (basic) model, which proves that the additional features have a measurable effect on the accuracy of entity extraction, and, especially on the performance of extracting the consequence class, the practical value of the sentence dependency feature for extracting the entity type of nonnoun or nonnoun phrases verified.

Combined with the above experiments, the performance of the BilSTM-Attention-CRF-Crowd model is excellent and is superior to the other models studied in this paper.

6. Conclusion

In this paper, we have proposed a new model BiLSTM-Attention-CRF-crowd to improve the quality of crowdsourcing annotations in information security field. The main work includes the following: (1) the common features of crowd annotations are found by the GAN model to generate the best unique consensus annotation; (2) these common features, domain dictionaries, and sentence dependencies are used as additional features to identify the entities of crowdsourcing annotations again, so as to improve the quality of crowdsourcing annotations. We evaluate our model on data sets in the field of information security, and the results show that its performance is better than the other baseline models mentioned in this paper. It is also verified that the proposed domain dictionary features and sentence dependency features have practical value for improving the performance of the model. However, the increase of input features will inevitably lead to an increase in the time complexity of the model. In future, we will consider further improvements to the model.

Data Availability

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

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61501515) and the Project of Henan Provincial Key Scientific and Technology (Grant no. 172102210002).

References

[1] C. Jochim and L. A. Deleris, "Named entity recognition in the medical domain with constrained CRF models," in *Proceedings of the 15th Conference of the European Chapter of the Association*

for Computational Linguistics, pp. 839–849, Computational Linguistics, Spain, April 2017.

[2] S. Liu, B. Tang, Q. Chen, and X. Wang, "Drug name recognition: approaches and resources," *Information*, vol. 6, no. 4, pp. 790–810, 2015.

[3] J. Liang, X. Xian, X. He, M. Xu, S. Dai, J. Xin et al., "A novel approach towards medical entity recognition in chinese clinical text," *Journal of Healthcare Engineering*, vol. 2017, Article ID 4898963, 16 pages, 2017.

[4] J. Lei, B. Tang, X. Lu, K. Gao, M. Jiang, and H. Xu, "A comprehensive study of named entity recognition in chinese clinical text," *Journal of the American Medical Informatics Association*, vol. 21, no. 5, pp. 808–814, 2014.

[5] Z. Liu, M. Yang, and X. Wang, "Entity recognition from clinical texts via recurrent neural network," *BMC Medical Informatics & Decision Making*, vol. 17, no. 2, 2017.

[6] S. Ramamoorthy and S. Murugan, "An attentive sequence model for adverse drug event extraction from biomedical text," https://arxiv.org/abs/1801.00625.

[7] R. Leaman and C. H. Wei, "A high performance approach for chemical named entity recognition and normalization," *Journal of Cheminformatics*, vol. 7, no. 1, pp. S1–S3, 2015.

[8] L. Luo and Z. Yang, "An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition," *Bioinformatics*, vol. 34, no. 8, pp. 1381–1388, 2018.

[9] T. H. Dang, H.-Q. Le, T. M. Nguyen, and S. T. Vu, "D3NER: biomedical named entity recognition using CRF-biLSTM improved with fine-tuned embeddings of various linguistic information," *Bioinformatics*, vol. 34, no. 20, pp. 3539–3546, 2018.

[10] I. Korvigo, M. Holmatov, A. Zaikovskii, and M. Skoblov, "Putting hands to rest: efficient deep CNN-RNN architecture for chemical named entity recognition with no hand-crafted rules," *Journal of Cheminformatics*, vol. 10, no. 1, 2018.

[11] R. A. Bridges, C. L. Jones, M. D. Iannacone, K. M. Huffer, and J. R. Goodall, "Automatic labeling for entity extraction in cyber security," https://arxiv.org/abs/1308.4941.

[12] S. Weerawardhana, S. Mukherjee, I. Ray, and A. Howe, "Automated extraction of vulnerabilit-y information for home computer security foundations and practice of security," in *Proceedings of the springer International Publishing*, vol. 8930, pp. 356–366.

[13] R. Lal, "Information extraction of cybersecurity related terms and concepts from unstructured text," in *Proceedings of the 2013 IEEE Seventh International Conference on Semantic Computing (ICSC)*, University of Maryland, California, Calif, USA, September 2013.

[14] V. Mulwad, W. Li, and A. Joshi, "Extracting information about security vulnerabilities from Web text," in *Proceedings of the 2011 IEEE/WIC/ACM International Conference On Web Intelligence and Intelligent Agent Technology*, pp. 257–260, IEEE Computer Society, August 2011.

[15] Amazon Mechanical Turk., https://www.mturk.com/.

[16] R. Snow and B. O'Connor, "Cheap and fast – but is it good? Evaluating non-expert annotations for natural language tasks," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 254–263, ACL, USA, October 2008.

[17] R. S. Boyer and J. S. Moore, "MJRTY—a fast majority vote algorithm," in *Automated Reasoning Series*, vol. 1 of *Automated Reasoning Series*, pp. 105–117, Springer Netherlands, Dordrecht, 1991.

[18] I. J. Goodfellow, J. Pouget-Abadie, and M. Mirza, "Generative adversarial nets," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pp. 2672–2680, Quebec, Canada, 2014.

[19] Y. S. Yang, M. Zhang, and W. Chen, "Adversarial learning for chinese NER from crowd annotations," https://arxiv.org/abs/1801.05147.

[20] Z. Syed, A. Padia, and T. Finin, "UCO: a unified cybersecurity ontology," in *Proceedings of the twenty-sixth AAAI*, 2016.

[21] Y. Z. Zhang, Z. Gan, and L. Carin, "Generating text via adversarial training," in *Proceedings of the 2016 Conference of Neural Information Processing Systems Workshop on Adversarial Training*, 2016.

[22] L. Yu, W. Zhang, and J. Wang, "SeqGAN: Sequence generative adversarial nets with policy gradient," https://arxiv.org/abs/1609.05473.

[23] J. Li and W. Monroe, "Adversarial learning for neural dialogue generation," https://arxiv.org/abs/1701.06547.

[24] M. J. Kusner and J. M. Hernandeslobato, "GANS for sequences of discrete elements with the gumbel-softmax distribution," https://arxiv.org/abs/1611.04051.

[25] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer Error-Rates using the EM algorithm," *Journal of Applied Statistics*, vol. 28, no. 1, pp. 20–28, 1979.

[26] M. Dredze, P. Talukdar, and K. Crammer, "Sequence learning from data with multiple labels," *Rule Based*, pp. 39–48, 2010.

[27] H. Kajino, Y. Tsuboi, H. Kashima et al., "A convex formulation for learning from crowds," in *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pp. 73–79, 2012.

[28] A. T. Nguyen, B. C. Wallace, and J. J. Li, "Aggregating and predicting sequence labels from crowd annotations," *Meeting of the Association for Computational Linguistics*, pp. 299–309, 2017.

[29] R. Agerri, Y. Chung, I. Aldabe, N. Aranberri, G. Labaka, and G. Rigau, "Building named entity recognition taggers via parallel corpora," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018*, 2018.

[30] J. M. Giorgi and G. D. Bader, "Transfer learning for biomedical named entity recognition with neural networks," Bioinformatics, 2018.

[31] S. Lai, K. Liu, L. Xu, and J. Zhao, "How to generate a good word emdedding," *IEEE Intelligent Systems*, vol. 31, pp. 5–14, 2016.

[32] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representation of words and phrases compositionality," *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.

[33] "CVE homepag," https://cve.mitre.org/.

[34] "Corenlp homepage," http://corenlp.stanford.edu.

[35] Q. Zhang, J. Fu, X. Liu, and X. Huang, "Adaptive co-attention network for named entity recognition in tweets," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[36] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," https://arxiv.org/abs/1412.6980.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 9325082, 8 pages
https://doi.org/10.1155/2018/9325082

Research Article

Generalized Bootstrapping Technique Based on Block Equality Test Algorithm

Xiufeng Zhao and Ailan Wang

Department of Information Research and Security, Zhengzhou Information Science Technology Institute, Zhengzhou, 450001, China

Correspondence should be addressed to Xiufeng Zhao; zhao_xiu_feng@163.com

Received 29 September 2018; Revised 19 November 2018; Accepted 9 December 2018; Published 24 December 2018

Guest Editor: Pelin Angin

With the rapid development of cloud computation and big data, the data storage and outsource computation are delegated to the untrusted cloud, which has led to a series of challenging security and privacy threats. Fully homomorphic encryption can be used to protect the privacy of cloud data and solve the trust problem of third party. The key problem of achieving fully homomorphic encryption is how to reduce the increasing noise during the ciphertext evaluation. Bootstrapping procedure can refresh ciphertext with large error, such that the resulting ciphertext has potentially smaller error and allows being continuous homomorphic evaluation. In this paper, we investigated the bootstrapping procedure used to construct fully homomorphic encryption scheme. We proposed a new concept of block homomorphic equality test algorithm and gave an instance based on the FH-SIMD scheme. Furthermore, based on the block homomorphic equality test algorithm, we proposed a faster bootstrapping procedure with smaller bootstrapping keys. Both theory analysis and experiment simulation validate high performance of our bootstrapping algorithm.

1. Introduction

Rapidly developing cloud storage and computation platform allow user delegate data outsource to the cloud server. Cloud computing has the characteristics of data concentration, resource sharing, highly interconnecting, fully opening, etc. It breaks the information island of traditional IT field; meanwhile, it brings even more serious security problems. To protect the privacy of data and the confidential of business secret, it is necessary to encrypting the upload data. However, it is difficult to process ciphertext for traditional encryption algorithm, and this promoted the improvement and development of fully homomorphic encryption (FHE). The prominent advantage of the fully homomorphic encryption is that it can solve ciphertext evaluation problem.

In 2009, Gentry [1, 2] constructed the first fully homomorphic encryption scheme using ideal lattice, which supports arbitrary depth circuit evaluation. Since then many fully homomorphic encryption schemes have appeared involving new mathematical concepts and NP hard problems and improving efficiency, such as FHE from LWE [3], Ring LWE [4], Integer [5], and LWR [6].

In PKC 2010, Smart and Vercauteren [7] proposed a variant of Gentry's scheme with relatively small key and ciphertext sizes. Packing messages allows us to apply single-instruction-multiple data (SIMD) homomorphic operations to many encrypted messages. Smart and Vercautren [8] showed that applying the Chinese reminder theorem (CRT) to number fields partitions the message space of Gentry's FHE scheme into a vector of plaintext slots, resulting in a substantial speed-up, the scheme denoted as FH-SIMD. In the work, they explained that the SIMD operations could be utilized to perform many higher level operations, such as performing AES encryption homomorphically and searching an encrypted database on a remote untrusted server.

Gentry, Sahai and Waters [9] constructed a simple homomorphic encryption scheme from learning with errors in Crypto 2013, called GSW scheme. In this work, they proposed a new technique for building FHE scheme via the approximate eigenvector method. The homomorphic addition and multiplication In GSW scheme are just matrix addition and multiplication, which makes GSW scheme both asymptotically faster and easier to understand. Otherwise, GSW scheme operates single bit once encryption and it is

required to take heavy cost for evaluating a large number of ciphertexts.

Bootstrapping technique is a central technique on fully homomorphic encryption (FHE), which converts "somewhat homomorphic" encryption (SHE) scheme into a fully homomorphic one. That is, bootstrapping procedure homomorphically evaluating the SHE scheme's decryption function on a ciphertext that cannot support any further homomorphic operations, and produces a new one that encrypts the same message and can handle more homomorphic operations.

Bootstrapping procedure is computationally very expensive, and it is becomes the main bottleneck of fully homomorphic encryption practicability. Therefore, there are lots of works try to improve its efficiency. Gentry, Halevi, and Smart [9] proposed a simpler approach that bypasses the homomorphic modular-reduction bottleneck by working with a modulus very close to a power of two. In Crypto 2013, Alperin-Sheriff and Peikert [10] gave entirely algebraic algorithm for bootstrapping in quasilinear time. They gave a method for homomorphically evaluating a class of structured linear transformation using "ring-switching" procedure, resulting in evaluating the decryption function efficiently.

Recently, Alperin-Sheriff and Peikert [11] proposed generalized bootstrapping technique using GSW scheme. The homomorphic decryption of FHE scheme from LWE concludes inner production and rounding operation, and homomorphic equation text algorithm is the key subprocedure of the rounding operation. Embedding the additive group $\mathbb{Z}_q$ into the symmetric group of $q \times q$ permutation matrices is another technique used in the work [11].

In Eurocrypt 2015, Ducas and Micciancio [12] gave an efficient bootstrapping technique by encoding the cyclic group $\mathbb{Z}_q$ into the group of roots unity: $i \longmapsto X^i$, where i is primitive $q-$ th root of unity. This allows implementing a bootstrapping procedure similar to the work of Alperin-Sheriff and Peikert [11], but where each cyclic group element is encoded by a single ciphertext, rather than a vector of ciphertext, this efficiently reduces the size of bootstrapping key.

In AsiaCrypt2016, Chillotti et al. constructed an efficient bootstrapping fully homomorphic encryption scheme, called TFHE [13]. Its time of running bootstrapping is less than 0.1 second. In AsiaCrypt2017, Chillotti et al. [14] optimized the multiple addends of work [13], and made the bootstrapping time reduced 13 milliseconds. 2018, Zhou et al. [15] optimized the serial addends to parallel addends, and the speed of single bootstrapping gate is faster that of work [14]. TFHE scheme and the optimized version both are single bit bootstrapping procedure [13–15]. Although a lot of effort is being spent on improving bootstrapping, the efficient and effective method has yet to be developed. And how to construct efficient multibit bootstrapping procedure is worth further study.

Our Results. In this paper we investigate the homomorphic equality test algorithm in bootstrapping procedure and proposed the concept of block homomorphic equality test algorithm B_Eq? and give an instance based on the FH-SIMD scheme. Furthermore, we proposed a faster bootstrapping procedure based on the block homomorphic equality test algorithm. Both theory analysis and experiment simulation validate the higher performance of our bootstrapping algorithm than that of Alperin-Sheriff and Peikert's work [11].

Organization. In Section 2, we describe some preliminaries on the field and homomorphism, and the concept of generalized bootstrapping technique. In Section 3, we proposed block homomorphic equality test algorithm B_Eq? and give a faster bootstrapping procedure based on B_Eq? algorithm. In Section 4, we give theory analysis and experiment simulation. We give conclusions in Section 5.

2. Preliminaries

2.1. Field and Homomorphism. Let $F(\mathrm{x}) \in \mathbb{F}_2[x]$ be a monic polynomial of degree N, which decomposed to exactly l distinct irreducible factors as follows:

$$F(\mathrm{x}) = \sum_{i=1}^{l} F_i(x), \tag{1}$$

where every polynomial $F_i(x)$ has degree $D = N/l$.

Letting A denote the algebra $\mathrm{A} \coloneqq \mathbb{F}_2[x]/(F)$, we can get the natural homomorphism via Chinese Remainder Theorem (CRT):

$$\mathrm{A} \cong \frac{\mathbb{F}_2[x]}{(F_1)} \otimes \cdots \otimes \frac{\mathbb{F}_2[x]}{(F_l)} \cong \mathbb{F}_{2^D} \otimes \cdots \otimes \mathbb{F}_{2^D}. \tag{2}$$

For $n \mid D$, the finite field $\mathbb{K}_n \coloneqq \mathbb{F}_{2^n}$ is a subfield of $\mathbb{F}_{2^D}$. Let $\mathbb{F}_2[x]/(K_n(x))$ denote a fixed canonical representation of $\mathbb{K}_n$, where $K_n(x) \in \mathbb{F}_2[x]$ is some irreducible polynomial of degree n. Let ψ be a fixed root of $K_n(x)$ in the algebraic closure of $\mathbb{F}_2$. Since $\mathbb{K}_n$ is contained in each of $\mathbb{L}_i \coloneqq \mathbb{F}_2[x]/(F_i)$, there is a homomorphic embedding as follows:

$$\Psi_{n,i}: \begin{cases} \mathbb{K}_n \longrightarrow \mathbb{L}_i \\ \alpha(\psi) \longmapsto \alpha(\sigma_{n,i}(\theta_i)), \end{cases} \tag{3}$$

where $\sigma_{n,i}(\theta_i)$ is a root of $K_n(x)$ in algebra $\mathbb{L}_i$, that is,

$$K_n(\sigma_{n,i}(x)) \equiv 0 \pmod{F_i(x)}. \tag{4}$$

According to CRT and the above homomorphic embedding, we can obtain a homomorphic embedding of $\mathbb{K}_n^l$ into the algebra A which defined as follows:

$$\Gamma_{n,l}: \begin{cases} \mathbb{K}_n^l \longrightarrow A \\ (\mathscr{k}_1(\psi), \ldots, \mathscr{k}_l(\psi)) \longmapsto \sum_{i=1}^{l} \mathscr{k}_i(\sigma_{n,i}(x)) \cdot H_i(x) \cdot G_i(x). \end{cases} \tag{5}$$

where the polynomials $H_i(x)$ and $G_i(x)$ is obtained by CRT and computed as follows:

$$\begin{aligned} H_i(x) &\longleftarrow \frac{F(x)}{F_i(x)}, \\ G_i(x) &\longleftarrow (H_i(x))^{-1} \pmod{F_i(x)}. \end{aligned} \tag{6}$$

From the above definition of $\Gamma_{n,l}$, we can see that $\Gamma_{n,l}$ maps a vector of l binary polynomials $(\mathscr{k}_1(\psi), \ldots, \mathscr{k}_l(\psi))$ each of degree less than n, into a single polynomial $a(x)$ of degree less than N. The map $\Gamma_{n,l}$ defines an isomorphism between $\mathbb{K}_n^l$ and $\Gamma_{n,l}(\mathbb{K}_n^l)$, so the inverse map $\Gamma_{n,l}^{-1}$ is well defined from $\Gamma_{n,l}(\mathbb{K}_n^l)$ to $\mathbb{K}_n^l$. We can represent $\Gamma_{n,l}^{-1}$ as follows:

$\Gamma_{n,l}^{-1}$:

$$\begin{cases} \Gamma_{n,l}\left(\mathbb{K}_n^l\right) \subseteq A \longrightarrow \mathbb{K}_n^l \\ a(x) \longmapsto \left(a(x) \bmod F_1(x), \ldots, a(x) \bmod F_l(x)\right). \end{cases} \tag{7}$$

There are two methods to compute elements in $\mathbb{K}_n^l$: one method is computes component wise on vectors of l elements in $\mathbb{K}_n$; the other concludes three process, firstly, mapping all the inputs to the algebra A by $\Gamma_{n,l}$; secondly, performing computations in algebra A; finally, mapping the results back to $\mathbb{K}_n^l$ by $\Gamma_{n,l}^{-1}$. Furthermore, the fully homomorphic encryption scheme FH-SIMD performs one evaluation for l elements in $\mathbb{K}_n$ using the algebra A.

2.2. Generalized Bootstrapping Technique. Gentry firstly proposed bootstrapping technique, which may transform a somewhat homomorphic encryption scheme to a fully homomorphic encryption scheme. Subsequently, Jacob Alperin-Sheriff and Chris Peikert [11] proposed generalized bootstrapping technique. The generalized bootstrapping technique involves two encryption schemes, outer encryption scheme and inner encryption scheme. It performs decryption procedure of inner encryption scheme using outer encryption scheme, resulting in reducing error in ciphertext. The generalized bootstrapping technique allows that the outer encryption is different from the inner one, realizing that we can design corresponding outer encryption scheme for the concretely inner encryption scheme, such that it effectively performs the decryption circuit of inner encryption scheme. Therefore, the generalized bootstrapping is more efficient than the ordinary one.

2.3. The Decryption of FHE from LWE. The decryption of all fully homomorphic schemes based on LWE involved computing inner production and rounding, that is, input secret key $s \in \mathbb{Z}_q^d$ and binary ciphertext $c \in \{0,1\}^d$; the decryption algorithm is written as

$$\mathrm{Dec}(s, c) = \lfloor \langle s, c \rangle \rceil_2 \in \{0,1\}, \tag{8}$$

where the modular rounding function $\lfloor \cdot \rceil_2 : \mathbb{Z}_q \longrightarrow \{0,1\}$ indicates whether its arguments is "far from" or "close to" 0 (modulo q), and the modulus q and the dimension d can both be made as small as quasi-linear $\widetilde{O}(\lambda)$ in the security parameter via dimension-modulus reduction [3], while still providing provable 2^λ security under conventional lattice assumption. The inner product $\langle s, c \rangle$ is just summing the elements of vector s selectively, that is,

$$\langle s, c \rangle = \sum_{c_j=1} s_j. \tag{9}$$

Supposing that $\langle s, c \rangle = v$, the algorithm rounding can be interpreted by iteration as

$$\lfloor v \rceil_2 = \sum_{x \in \mathbb{Z}_q \ s.t. \lfloor x \rceil_2 = 1} [x = v], \tag{10}$$

where $[x = v]$ denotes the equality test algorithm, when x is equality to v, $[x = v]$ outputs 1; otherwise, $[x = v]$ outputs 0.

Now, we give the decryption algorithm of FHE based LWE in the ciphertext state. During the bootstrapping procedure, the ciphertext of secret $s \in \mathbb{Z}_q^d$ is written by $\overline{s} = (\overline{s_1}, \ldots, \overline{s_d})$ as bootstrapping public key. The inner product in the ciphertext state is denoted as $\overline{v} = \langle \overline{s}, c \rangle = \sum_{c_j=1} \overline{s}_j$. And the rounding algorithm in the ciphertext state is denoted as

$$\lfloor \overline{v} \rceil_2 = \boxplus_{x \in \mathbb{Z}_q \ s.t. \lfloor x \rceil_2 = 1} \left(\overline{[x = v]}\right), \tag{11}$$

where "$\boxplus$" denotes the homomorphic addition on the ciphertext space and $\overline{[x = v]}$ indicates the homomorphic equality test algorithm; it outputs the ciphertext of 1 if and only if $x = v$; otherwise it outputs the ciphertext of 0. We let $\overline{1}$ denote the ciphertext of 1 and $\overline{0}$ denote the ciphertext of 0.

2.4. Generalized Bootstrapping Procedure of FHE from LWE. Assume that the binary ciphertext to be bootstrapped is $c \in \{0,1\}^d$, the secret key is $s \in \mathbb{Z}_q^d$, and the dimension d and the module q are enough small $(q, d = \widetilde{Q}(\lambda))$. The decryption function of FHE scheme from LWE is $Dec_s(c) = \lfloor \langle s, c \rangle \rceil_2 \in \{0,1\}$. We also supposed that the outer encryption scheme is FH-SIMD, that is, FHE scheme which supports SIMD operation. The generalized bootstrapping technique concludes two algorithms: **BootGen** algorithm and **Bootstrap** algorithm [11].

(i) **BootGen**($s \in \mathbb{Z}_q^d$, pk): input secret key vector $s \in \mathbb{Z}_q^d$, and the public key of FH-SIMD encryption; output the bootstrapping public key bk, that is, encrypt the secret key vector s via FH-SIMD scheme and resulting the ciphertext as the bootstrapping public key bk.

(ii) **Bootstrap**($bk, c = \{0,1\}^d$): input the bootstrapping pubic key bk and the ciphertext vector $c = \{0,1\}^d$, output a new ciphertext c' of original encryption scheme based LWE, and the result of decrypting c' using secret key $s \in \mathbb{Z}_q^d$ is same as the one decrypting c using secret key $s \in \mathbb{Z}_q^d$, but c' with less error.

3. Faster Bootstrapping Based on FH-SIMD

3.1. Main Ideas. Jacob Alperin-Sheriff and Chris Peikert proposed the generalized bootstrapping method based on the GSW scheme. Homomorphic equality test is a key component of the generalized bootstrapping algorithm, that is, for the fixed $v \in \mathbb{Z}_q$, under the ciphertext state, travels every $x \in \mathbb{Z}_q$ which satisfies $\lfloor x \rceil_2 = 1$, and decide that whether $v = x$ or not, see Figure 1.

We intend to proposed block homomorphic equality test algorithm, that is, it travels a block $(x_1, x_2, \ldots, x_l)$, $x_i \in \mathbb{Z}_q$

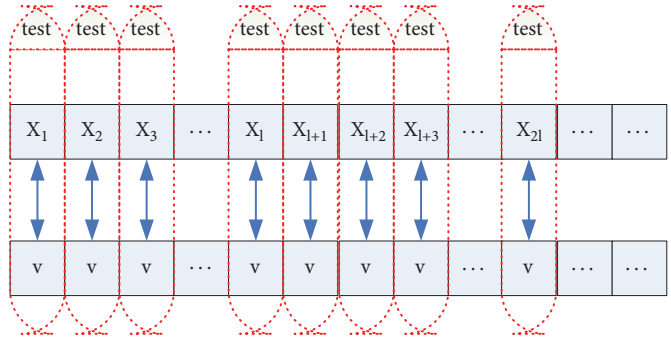


Figure 1: Homomorphic equality test.

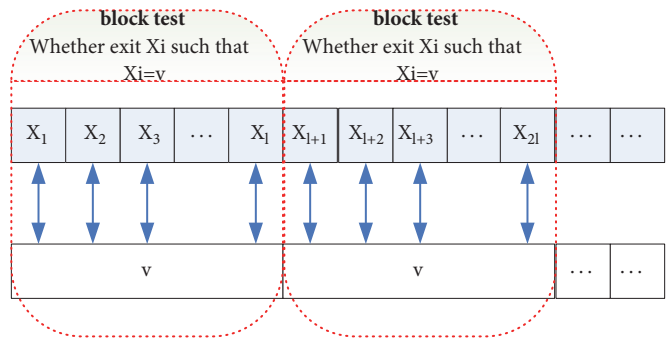


Figure 2: Block homomorphic equality test.

which satisfies $\lfloor x \rceil_2 = 1$, and decide that whether this batch of $(x_1, x_2, \ldots, x_l)$ exits a x_i such that $v = x_i$ holds, see Figure 2.

Resort to the FH-SIMD homomorphic encryption scheme [7], we will give a block homomorphic equality test algorithm B_Eq?, and then propose an efficient generalized bootstrapping algorithm. The bootstrapping key is an encryption of each coordinate of the secret key $s \in \mathbb{Z}_q^d$, and consists of d FH-SIMD ciphertexts. To bootstrapping $c \in \{0,1\}^d$, the inner product $\langle s, c \rangle \in \mathbb{Z}_q$ is computed homomorphically as a subset-sum using addition method, and the rounding function is computed using block homomorphic equality test algorithm and addition method.

3.2. Block Homomorphic Equality Test Algorithm. In this section, we describe our block homomorphic equality test algorithm, called B_Eq?.

Input: ciphertex $\overline{v} \in \mathbb{Z}_q$ and plaintext block $(x_1, \ldots, x_l) \in \mathbb{Z}_q^l$.

Output: if there exits a $x_i \in \{x_1, x_2, \ldots, x_l\}$ such that $v = x_i$ holds, then the block homomophic equality test algorithm outputs $\overline{1}$; otherwise it outputs $\overline{0}$.

We assume that $\overline{v}$ and all x_i are n bits in length, and thus can be encoded as an element of the finite filed $\mathbb{K}_n = \mathbb{F}_{2^n}$, where $n = \lceil \log q \rceil$. The concrete procedure is described as follows:

(1) For $(x_1, \ldots, x_l) \in \mathbb{Z}_q^l$, where every x_i satisfies $\lfloor x_i \rceil_2 = 1$, embed each coordinate $x_i \in \mathbb{Z}_q$ as an element of $\mathbb{F}_{2^n}$. Our aim is to pack $(x_1, \ldots, x_l)$ into single element $X \in \mathbb{K}_n$, so we compute $X = \Gamma_{n,l}(x_1, \ldots, x_l)$. Then compute the trivial encryption of X in the algebra $\mathbf{A}$ using FH-SIMD scheme. It is worth noting that we encrypt X without random, such that saving computational cost. That is,

$$\overline{X} = \text{FH} - \text{SIMD.Encrypt}\left(\Gamma_{n,l}\left(x_1, \ldots, x_l\right), pk\right). \tag{12}$$

Then compute

$$\overline{V} = \Gamma_{n,l}\left(\overline{v}, \ldots, \overline{v}\right). \tag{13}$$

(2) Sum the ciphertext $\overline{V}$ and $\overline{X}$, and denote the sum as $c^{(1)}$, that is,

$$c^{(1)} = \overline{V} \boxplus \overline{X}. \tag{14}$$

(3) Homomorphically raised $c^{(1)}$ to the power $(2^n - 1)$, that is,

$$c^{(2)} = \left(c^{(1)}\right)^{2^n - 1}. \tag{15}$$

And compute

$$\Gamma_{n,l}^{-1}\left(c^{(2)}\right) = \left(\left(\overline{v} \boxplus \overline{x_1}\right)^{2^n-1}, \ldots, \left(\overline{v} \boxplus \overline{x_l}\right)^{2^n-1}\right). \tag{16}$$

According to the homomorphism of encryption scheme FH-SIMD, the $(2^n - 1)$ power of ciphertext is corresponding to the $(2^n - 1)$ power of plaintext. Of course, the ciphertext is homomorphically raised to the power $(2^n - 1)$, via performing $2n$ applications of multiplication. Since the plaintext corresponding to $(\overline{v} \boxplus \overline{x_i})$ is an element of finite field $\mathbb{F}_{2^n}$ and

its max multiplicative order is $(2^n - 1)$, then the plaintext corresponding to $(\overline{v} \boxplus \overline{x}_i)^{2^n-1}$ is either 0 or 1. Therefore, $(\overline{v} \boxplus \overline{x_i})^{2^n-1}$ is a ciphertext of encrypting 1 (nonzero element) or a ciphertext of encrypting 0 (zero element). When $(\overline{v} \boxplus \overline{x}_i)$ is an encryption of 0, this means that $v = x_i$ holds; when $(\overline{v} \boxplus \overline{x}_i)$ is an encryption of 1, this means that $v = x_i$ does nit hold.

(4) Compute

$$c^{(3)} = c^{(2)} \boxplus \mathrm{FH} - \mathrm{SIMD.Encrypt}\left(\Gamma_{n,l}(1, \ldots, 1), pk\right), \quad (17)$$
$$\Gamma_{n,l}^{-1}\left(c^{(3)}\right) = \left(\left(\overline{v} \boxplus \overline{x_1}\right)^{2^n-1} \boxplus \overline{1}, \ldots, \left(\overline{v} \boxplus \overline{x_l}\right)^{2^n-1} \boxplus \overline{1}\right).$$

We can see that as long as the equation $x_i = v$ holds, the i-th component of $c^{(3)}$ is $\overline{0} \boxplus \overline{1} = \overline{1}$; otherwise, the i-th component of $c^{(3)}$ is $\overline{1} \boxplus \overline{1} = \overline{0}$. Therefore, if there exits a $x_i \in \{x_1, x_2, \ldots, x_l\}$ such that $v = x_i$ holds, then $(\overline{v} \boxplus \overline{x}_i)^{2^n-1} \boxplus \overline{1} = \overline{1}$, and all other $(\overline{v} \boxplus \overline{x}_j)^{2^n-1} \boxplus \overline{1} = \overline{0}$, for all $j \neq i$. It follows that $c^{(3)} = \overline{1}$. That is, if there exits a $x_i \in \{x_1, x_2, \ldots, x_l\}$ such that $v = x_i$ holds, the block homomophic equality test algorithm outputs $\overline{1}$; otherwise it outputs $\overline{0}$.

From the above steps, we finish the block homomorphic equality test; then we can homomorphic compute $\lfloor v \rceil_2$, that is, $\lfloor \overline{v} \rceil_2 \coloneqq c^{(3)}$.

3.3. Faster Bootstrapping Technique. In this section, we construct faster bootstrapping procedure from the block homomorphic equality test algorithm B_Eq?. The bootstrapping procedure consists of two algorithms: BootKeyGen and Bootstrap. The procedure is used to refresh ciphertexts of all known standard LWE-based FHE. We get the input ciphertext $c \in \{0,1\}^d$ for Bootstrap, and it is from the dimension-modulus reduction and bit-decomposition of the ciphertext to be bootstrapped. Let $\mathrm{s} \in \mathbb{Z}_q^d$ be the secret key that corresponding to the ciphertext c.

BootGen($\mathrm{s} \in \mathbb{Z}_{\mathrm{q}}^{\mathrm{d}}$, pk): input the secret key $\mathrm{s} \in \mathbb{Z}_{\mathrm{q}}^{\mathrm{d}}$ for the ciphertext to be refreshed and the public key pk of FH-SIMD scheme. Without loss of generality, for every $j \in [d]$, encode each coordinate $\mathrm{s}_{\mathrm{j}} \in \mathbb{Z}_{\mathrm{q}}$ to the element of finite field $\mathbb{F}_{2^{\mathrm{n}}}$. Then encrypt its ciphertext under FH-SIMD scheme, and generate the bootstrapping key:

$$\overline{\mathrm{s}}_j = \mathrm{FH} - \mathrm{SIMD.Encrypt}\left(\mathrm{s}_{\mathrm{j}}, pk\right). \quad (18)$$

Output the bootstrapping public key $bk = \{\overline{s_1}, \ldots, \overline{s_d}\}$. The bootstrapping key consists of d FH-SIMD ciphertexts.

Bootstrap($bk, c \in \{0,1\}^d$): input the binary ciphertext $c \in \{0,1\}^d$, and perform the following two phases:

(i) **Inner Product** Homomorphically compute inner product $\overline{v}$ using the bootstrapping public key bk. It is known that

$$v \triangleq \langle s, c \rangle = \sum_{j:c_j=1} s_j \in \mathbb{Z}_q. \quad (19)$$

It follows that

$$\overline{v} \triangleq \langle \overline{s}, c \rangle = \boxplus_{j:c_j=1} \overline{\mathrm{s}}_j. \quad (20)$$

(ii) **Round** For every $x_i \in \mathbb{Z}_q$ which satisfies $\lfloor x_i \rceil_2 = 1$, arrange in order of size, and divide them into blocks of l items, and let us suppose they are distinct from one another, and there are altogether k blocks:

$$\left(x_1, \ldots, x_l\right), \left(x_{l+1}, \ldots, x_{2l}\right), \ldots, \left(x_{(k-1)l+1}, \ldots, x_{kl}\right). \quad (21)$$

For every block $(x_{jl+1}, \ldots, x_{(j+1)l})$, $j = 0, 1, \ldots, k-1$, run block homomorphic equality test algorithm in parallel,

$$c_j = \mathrm{B\_Eq?}\left(\overline{v}, \left(x_{jl+1}, \ldots, x_{(j+1)l}\right)\right). \quad (22)$$

Then compute

$$C = \boxplus_{j:1 \leq j \leq k} c_j. \quad (23)$$

We can see that c_j is either $\overline{1}$ or $\overline{0}$, and then C is the encryption of $\lfloor v \rceil_2$, C is also either $\overline{1}$ or $\overline{0}$, and it refreshed ciphertext with smaller error. Note that the output C is a FH-SIMD ciphertext encrypted under pk. If desired, we can convert this ciphertext back to one for the original LWE FHE cryptosystem. We can also perform key-switch from sk back to the original secret keys.

4. Analysis

4.1. Correctness Analysis

Lemma 1 (correctness). *For $bk \longleftarrow$ BootGen(s, pk), the FH-SIMD ciphertext $C \longleftarrow Bootstrap(bk, c)$ is designed to encrypt $Dec_s(c) = \lfloor \langle s, c \rangle \rceil_2 \in \{0,1\}$.*

Proof. Firstly, the FH-SIMD ciphertext $\overline{s}_j$ is designed to encrypt s_j from (18). Therefore, since $\varphi : \mathbb{Z}_q \longrightarrow \mathbb{F}_{2^n}$ is homomorphic embedding, the ciphertext $\overline{v}$ as defined as in (20) designed to encrypt

$$\sum_{j:c_j=1} s_j = \langle s, c \rangle = v. \quad (24)$$

By correctness of block homomorphic equality test algorithm B_Eq?, the homomorphic sum $C = \boxplus_{j:1 \leq j \leq k} c_j$ is designed to encrypt 1 if and only if $v = x$. Finally, since the homomorphic sum is taken over every $x \in \mathbb{Z}_q$ such that $\lfloor x \rceil_2 = 1$, it is designed to encrypt 1 if and only if $\lfloor v \rceil_2 = 1$. □

4.2. Security Analysis

Lemma 2 (semantic security). *Suppose that the FH-SIMD scheme secret key sk is generated independently of the secret key $\mathrm{s} = (s_1, \ldots, s_d) \in \mathbb{Z}_q^d$ of FHE scheme from LWE; then Ind-CPA security of the bootstrapping key follows immediately from the Ind-CPA security of FH-SIMD, hence from SIVP of ideal lattice.*

Table 1: Compare of the performance of homomorphic equality test algorithm.

algorithm	Enc	C_A	C_M	$\Gamma_{n,l}$	$\Gamma_{n,l}^{-1}$
Eq? [11]	0	0	$O(q)$	0	0
B_Eq?	2	2	$O(\log q)$	2	1

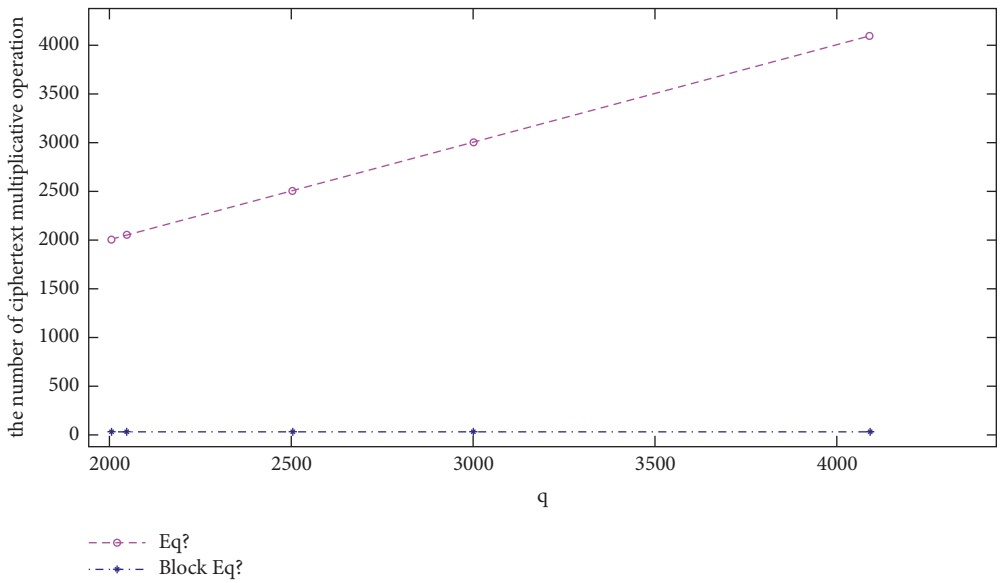


Figure 3: The relation between the ciphertext multiplicative quantity and the modulus q.

Proof. For $bk \longleftarrow$ BootGen(s, pk), if there is not an adversary can distinguish the bootstrapping key bk from a random element in the same space, then the bootstrapping procedure called satisfing semantic security.

In our bootstrapping procedure, for $s = (s_1, \ldots, s_d) \in \mathbb{Z}_q^d$, the bootstrapping key consists of d FH-SIMD ciphertexts; that is, $bk = \{\overline{s_1}, \ldots, \overline{s_d}\}$ is generated via FH-SIMD scheme, where

$$\overline{s}_j = \text{FH} - \text{SIMD.Encrypt}\left(s_j, pk\right). \tag{25}$$

Suppose that there is an adversary $\mathscr{A}$ can distinguish the bootstrapping bk from random element. Then we can construct an algorithm $\mathscr{B}$ by calling the adversary $\mathscr{A}$ and break the Ind-CPA security of FH-SIMD scheme and, furthermore, solve the SIVP of ideal lattice. □

4.3. Performance Analysis. Our block homomorphic equality test algorithm B_Eq? has a cost of $(2\cdot\text{Enc}+2\cdot C_A+2\log q\cdot C_M+2\cdot\Gamma_{n,l}+\Gamma_{n,l}^{-1})$ per data block, where C_A denotes add operation of the ciphertext and C_M denotes multiplicative operation of the ciphertext, whereas the homomorphic equality test algorithm Eq? involves $q \cdot C_A$, which is exponential times of B_Eq? algorithm, meaning that the computation of Eq? algorithm is more costly, reference to Table 1.

In the work of Alperin-Sheriff and Peikert [11], one inner product evaluation of bootstrapping needs to compute d ciphertexts compose evaluation, and one rounding evaluation of bootstrapping needs to call $O(q)$ Eq? algorithm, and $O(q)$ ciphertext multiplicative operation. Whereas one inner production of our faster bootstrapping needs to compute d ciphertext additions, and one rounding evaluation needs to call k B_Eq? algorithm, where l is the size of block, and k is the number of block, $k = \lceil q/l \rceil$.

Suppose that LWE problem has 80 bits security when q is set to be 2003. Parameters setting as above, and when q is set to be 2003, 2047, 2501, 3001, 4093, and 12899, we give the relation between multiplicative operation quantity and modulus q, as shown in Figure 3. As the modulus q increases, the number of ciphertext multiplicative operation grows swiftly in the AP's bootstrapping procedure, whereas the number of ciphertext multiplicative operation grows slowly.

On the other hand, for the fixed modulus q and m, we give the relation between multiplicative operation quantity once running our faster bootstrapping procedure based on the block size of block homomorphic equality test algorithm B_Eq?. When $m = 11441$, $N = \varphi(m) = 10752$, and $2^{48} \bmod 11441 = 1$, so $d = 48$, we set $n = 12$, which satisfies $n \mid d$. Then we set the size of block as $16, 32, 64, 128, 224$, that is, the size of block $l = 16, 32, 64, 128, 224$. Then the number of blocks $k = 3024, 1512, 768, 384, 216$. We can see from Figure 4, as the block size increases, the ciphertext multiplicative operation drops dramatically.

5. Conclusions

Fully homomorphic encryption scheme allows evaluating encrypted data, without decrypting the corresponding ciphertext. In fully homomorphic encryption scheme, the

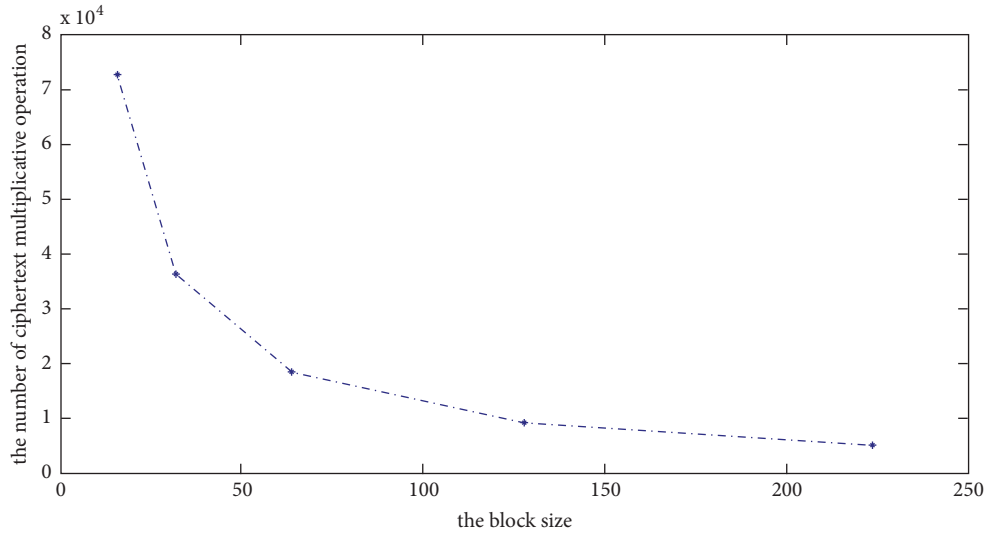


Figure 4: The relation between the ciphertext multiplicative quantity and the block size.

ciphertext has a noise that grows at each homomorphic evaluation. When the noise reaches a threshold, then the ciphertext cannot be decrypted correctly. The number of homomorphic operations can be made asymptotically large using bootstrapping technique.

In this paper, we further investigated the bootstrapping procedure. We proposed the concept of block homomorphic equality test algorithm and give an instance based on the FH-SIMD scheme. Furthermore, we give a faster bootstrapping procedure based on the block homomorphic equality test algorithm. Both theory analysis and experiment simulation validate the higher performance of our bootstrapping than that of the work [11].

Data Availability

Our underlying data related to the article is the paper as cited as in [11].

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Nature Science Foundation of China under Grant no. 61601515 and Nature Science Foundation of Henan Province under Grant no. 162300410332.

References

[1] C. Gentry, "Fully homomorphic encryption using ideal lattices," In Proc of the 41th Annual ACM Symp on Theory of Computing (STOC)," pp. 169–178, ACM, New York, NY, USA, 2009.

[2] C. Gentry, *A fully homomophic encryption scheme [Ph.D. thesis]*, Stanford University, 2009, http://crypto.stanford.edu/craig.

[3] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *Proceedings of the IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS '11)*, pp. 97–106, Palm Springs, Calif, USA, October 2011.

[4] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Advances in Cryptology—CRYPTO 2011*, R. Phillip, Ed., vol. 6841, pp. 505–524, Springer, Berlin, Germany, 2011.

[5] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Advances in cryptology (EUROCRYPT)*, vol. 6110, pp. 24–43, Springer, Berlin, Germany, 2010.

[6] Fucai Luo, Fuqun Wang, Kunpeng Wang, Jie Li, and Kefei Chen, "LWR-Based Fully Homomorphic Encryption, Revisited," *Security and Communication Networks*, vol. 2018, Article ID 5967635, 12 pages, 2018.

[7] N. P. Smart and F. Vercauteren, "Fully homomorphic SIMD operations," *Designs, Codes and Cryptography*, vol. 71, no. 1, pp. 57–81, 2014.

[8] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based," in *CRYPTO, LNCS*, R. Canetti and J. A. Garay, Eds., vol. 8042, pp. 75–92, Springer, Heidelberg, Germany, 2013.

[9] C. Gentry, S. Halevi, and N. P. Smart, "Better bootstrapping in fully homomorphic encryption," in *Public key cryptography (PKC)*, vol. 7293 of *Lecture Notes in Comput. Sci.*, pp. 1–16, Springer, Heidelberg, 2012.

[10] J. Alperin-Sheriff and C. Peikert, "Practical bootstrapping in quasilinear time," in *Advances in Cryptology – CRYPTO*, vol. 8042, pp. 1–20, 2013.

[11] J. Alperin-Sheriff and C. Peikert, "Faster bootstrapping with polynomial error," in *Proceedings of the International Cryptology Conference*, J. A. Garay and R. Gennaro, Eds., pp. 297–314, Springer, Berlin, Germany, 2014.

[12] L. Ducas and D. Micciancio, "FHEW: Bootstrapping homomorphic encryption in less than a second," in *EUROCRYPT, Part I, LNCS*, E. Oswald and M. Fischlin, Eds., vol. 9056, pp. 617–640, Springer, Heidelberg, Germany, 2015.

[13] I. Chillotti, N. Gama, M. Georgieva et al., "Faster fully homomorphic encryption: bootstrapping in less than 0.1 seconds," in *Advances in Cryptology - ASIACRYPT*, pp. 3–33, Springer, Heidelberg, Germany, 2016.

[14] I. Chillotti, N. Gama, M. Georgieva et al., "Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE," in *ASIACRYPT 2017. LNCS*, T. Takagi and T. Peyrin, Eds., vol. 10624, pp. 377–408, Springer, Cham, UK, 2017.

[15] T. Zhou, X. Yang, L. Liu, W. Zhang, and N. Li, "Faster Bootstrapping With Multiple Addends," *IEEE Access*, vol. 6, pp. 49868–49876, 2018.