Computers, Materials & Continua
DOI:10.32604/cmc.2021.017574
Article

Tech Science Press

Abnormal Event Correlation and Detection Based on Network Big Data Analysis

Zhichao Hu[1], Xiangzhan Yu[1,*], Jiantao Shi[1] and Lin Ye[1,2]

[1]School of Cyberspace Science, Harbin Institute of Technology, Harbin, 150001, China
[2]Department of Computer and Information Science, Temple University, Philadelphia, 42101, USA
[*]Corresponding Author: Xiangzhan Yu. Email: yxz@hit.edu.cn
Received: 03 February 2021; Accepted: 29 March 2021

Abstract: With the continuous development of network technology, various large-scale cyber-attacks continue to emerge. These attacks pose a severe threat to the security of systems, networks, and data. Therefore, how to mine attack patterns from massive data and detect attacks are urgent problems. In this paper, an approach for attack mining and detection is proposed that performs tasks of alarm correlation, false-positive elimination, attack mining, and attack prediction. Based on the idea of CluStream, the proposed approach implements a flow clustering method and a two-step algorithm that guarantees efficient streaming and clustering. The context of an alarm in the attack chain is analyzed and the LightGBM method is used to perform false-positive recognition with high accuracy. To accelerate the search for the filtered alarm sequence data to mine attack patterns, the PrefixSpan algorithm is also updated in the store strategy. The updated PrefixSpan increases the processing efficiency and achieves a better result than the original one in experiments. With Bayesian theory, the transition probability for the sequence pattern string is calculated and the alarm transition probability table constructed to draw the attack graph. Finally, a long-short-term memory network and embedding word-vector method are used to perform online prediction. Results of numerical experiments show that the method proposed in this paper has a strong practical value for attack detection and prediction.

Keywords: Attack scene; false positive; alarm correlation; sequence mining; multi-step attack

1 Introduction

With the continuous development of the Internet, the network industry has become increasingly prosperous. At the same time, however, cybercrime and threat activities have become a critical part of our daily life, and the importance of network security has continuously emerged as a central concern [1,2]. Hacker activities infiltrate multiple environments such as the Internet of Things (IoT) and cloud computing [3]. New technologies and applications provide new attack vectors for cybercrimes, which pose a significant threat to network system security and important data [4,5].

At present, there are effective countermeasures against single-step attacks, such as SQL injection and DDOS [6]. However, many novel attack types that are challenging to detect, especially multi-step attacks, have emerged in networks [7]. These kinds of attacks can cause great difficulties to network researchers. There are two main reasons for choosing an approach to such attacks: first, advanced attackers usually choose medium-sized or large organizations with complex network topology and multi-layer security protection [8,9]. The most valuable information is often stored in the final nodes that are inaccessible from the outside. It is almost impossible to complete an invasion with a single-step attack successfully. Second, if an attack is broken down into several independent steps, it will be more difficult for the victim to identify such an attack, especially if some steps do not pose a severe risk to the system [10]. In the face of this more secretive and responsible attack mode, one must carry out an in-depth analysis, consider the attack strategy under the overall situation, and determine the cause-and-effect relationship between attacks, to defend against it.

Since intrusion detection (ID) is the core element of network security, there are many complementary security devices widely deployed regarding various current security threats, such as an intrusion-detection system (IDS) and other preventive security mechanisms (such as access control, authentication, and firewalls). In this way, the availability, integrity, and confidentiality of computers and network services can be ensured [11]. Deploying multiple security products at the same time can improve the threat-detection rate, but it can also produce many primitive low-level alarms, most of which are false positives, making it difficult to extract real attack information. The elimination of false positives is of great significance for improving research efficiency and enhancing its accuracy.

From the above discussion, it is clear that there is a need for an efficient way to mine attack patterns from massive data and predict attacks. Hence, to address these issues, in this paper a multi-step attack-correlation and -detection method based on data mining and deep learning is proposed, aiming to eliminate false positives, analyze and mine attack scenes, and make predictions based on massive alarm data.

The rest of this paper is organized as follows. In Section 2, the most relevant related work in this area is described. In Section 3, an overview of the problem statement is provided and the proposed approach described. The experiment and results are presented in Section 4. Section 5 provides conclusions and planned future work.

2 Related Work

Alarm correlation and detection have been studied extensively in recent decades. With the increasingly severe network security situation, related research content has become increasingly more extensive and in-depth. In this paper, the related research is classified into four categories.

- Similarity-based approaches relate attacks according to the similarity between various steps, based on the idea that similar alarms come from the unified attack scene [12]. Therefore, the calculation method of similarity is the core of these approaches, making them different from the causal correlation approach that mainly concerns sequences' causal structure. Qiao et al. [13] proposed a concise formula to calculate the similarity between alarms in which double clustering is applied to extract alarm sequences using the longest common substring (LCS) algorithm.

- Causal-based approaches use the cause-and-effect relationship between alarm sequences as the critical factor to identify multi-step attacks. Two main methods exist in cause-and-effect correlation. The first method uses pre- and post-conditions. Pre-conditions refer to the conditions for the attack's success and post-conditions to the possible impact after the attack. Ning [14] conducted a significant amount of research using this approach to reduce alarm attributes utilizing connection diagrams and deduce the relationship of alarms using predicate logic. The second method is statistical inference, which is used to find statistical regularities in the data set, which derives from the alarm sequence frequency. The Bayesian network and hidden Markov model (HMM) are popular implementations in probability statistics based on this method. Kavousi et al. [15] extracted the Bayesian-network graph with the help of the transfer probability of alarm sequence and reduced the graph's scale by pruning.
- The structure-based approaches focus on the organization's current network structure. In addition, the security monitoring system contains each node's information and connection, especially the vulnerability of each node and the importance of the stored information. The main difference from the previous methods is that the structure-based approach builds the attack graph from the defensive side rather than from the attacker's. Noel et al. [16] applied this approach to mine multi-step attacks for the first time in 2004. Luo et al. [17] proposed a virtual-game-corresponding algorithm based on a dynamic game tree and applied it to find multi-step attacks.
- The hybrid approach is primarily a combination of the previous approaches in different phases [7]. Ramaki et al. [18] developed an RTECA framework that combines frequency analysis to build correlation matrices by the similarity of alarm attributes and matching of IP ports. His approach incorporated previous approaches and applied them to different stages according to their merits and demerits, achieving significant effectiveness improvements. Pajouh et al. [19] applied a two-tier classification module utilizing the naive Bayes and certainty factor versions of K-nearest neighbor to identify suspicious behaviors.

Case-based, pattern-matching, network-structure, prerequisite, and post-condition methods all manually use rules constructed by security researchers to some extent. Therefore, they all share certain similarities and are prone to confusion. In the case-based approach, the attack model must be built separately for each scene. In a pattern-matching approach, a higher-level model must abstract from each scene. In the structure-based approach, model construction mainly depends on the importance of network structure and information. The possible invasion path is assumed from vulnerable nodes, not on the attacker's actual behavior. The obvious advantage of these approaches based on manual rules is that they can significantly reduce false positives. However, when security experts pre-determine no rule, it cannot be effectively detected, and the scalability will be affected.

Methods based on similarity, probability statistics, and machine learning differ from knowledge-based ones to a certain extent. These methods tend to mine the attack patterns in existing data. The similarity-based method uses unsupervised clustering to generate behavior sequences by defining different similarity functions. The method based on probability statistics uses the idea of Bayesian probability to construct a probability attack graph. The graph can represent the specific scene when the attack occurs with intuitive graphical results and can predict the attack scene. Machine-learning-based methods belong to supervised classification, and are mainly used to identify false positives and attack-chain threat degrees. They automatically extract features from the data and reduce the analysis work of security personnel. These methods rely on collected

data, which means that the system faces a cold-start problem; that is, how it works when there is no data. Besides, there are some differences between automatic and manual detection in terms of accuracy and false positives. However, from the perspective of scalability in the face of new attacks, the former is better.

Researchers are currently working towards the integration of artificial intelligence and rules [11,20]. The combination saves a large amount of resources and improves the performance of the security system's accuracy and usability.

3 Proposed Solution

Different kinds of security facilities are equipped on servers, network switches, and other devices. They filter traffic and trigger alarms based on rules. Then, data-processing platforms collect and standardize alarms. The mining of attack patterns from alarm sequences and prediction of attacks are the main goals of the proposed solution.

As shown in the flow diagram (Fig. 1), there are four pivotal components of the proposed framework alarm correlation, false-positive detection, attack-scene mining, and attack-scene prediction. After the standard alarm data's initial processing, the alarm-correlation and false-alarm detection modules carry out the next level of clustering. These modules first remove redundancy through IP, port, alarm type, and so on to generate a meta-alarm, and then complete the alarm correlation using a streaming clustering algorithm. For alarm clustering, the feature-extraction module extracts the context features to identify false positives. The attack-scene mining module then extracts the frequent patterns from these data. It then calculates the transfer probability based on Bayesian theory and draws an attack graph to show the attack path. Finally, the prediction module segments the alarm sequence by way of the window and predicts the attack scene based on the training of word-vector embedding and LSTM.

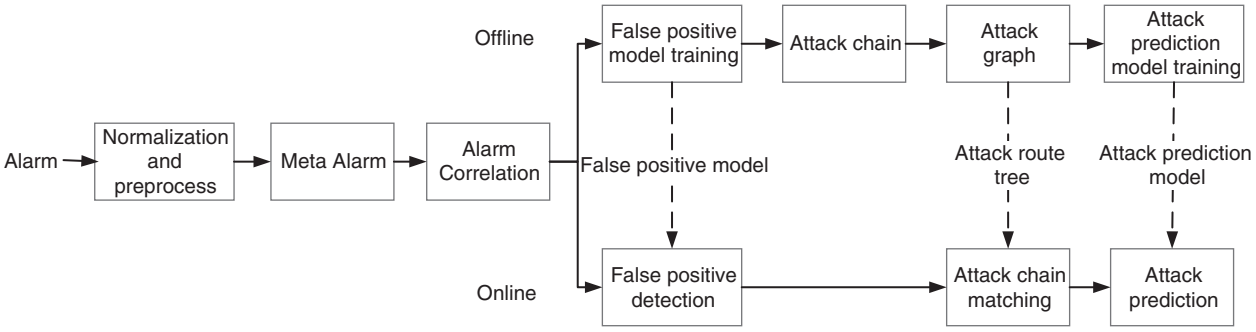

Figure 1: Flow diagram of alarm correlation and detection

3.1 Alarm Correlation

An alarm is the result of threat level obtained by rule matching of the IDS system for each packet, mainly including occurrence time, alarm type (action), alarm category (class), packet type (pack_type), source IP(src_ip), destination IP(dst_ip), source port (src_port), destination port (dst_port), and priority under IDS. The alarm-correlation algorithm deals with the alarm's characteristics and internal relationship between them, which aggregates them into a relatively complete attack chain.

IDS or NIDS creates raw alarms, and then standardized modules generate standard alarms with basic attributes, called raw alarms $A = \{a_1, a_2, a_3, \ldots, a_n\}$. When an alarm arrives, multiple IDSs may receive the same traffic and generate alarms with same contents. A host may issue many similar traffic packets with the same content and IP in a brief period. In this case, the generated alarms are redundant for subsequent analysis and must be merged to produce a meta-alarm, which is denoted $MA = \{ma_1, ma_2, ma_3, \ldots, ma_n\}$. The attributes contained in the ith meta-alarm ma_i are the same those contained in the standard alarm. While ma_i contains all the information for the standard alarms, it is essentially a queue that holds the original alarms. According to the meta-alarm concept, IP port and time are the critical characteristics. The algorithmic flow of meta-alarm aggregation calculates whether the IP ports are the same and at intervals. If the alarms a_i and a_j belong to the original alarm set A, then their source IP, source port, target IP, and target port are identical, and they satisfy the relationship $a_i.time - a_j.time \leq th$.

The meta-alarm aggregation algorithm uses a sliding time window. When the time difference between the timestamp of the newly arrived alarm and the earliest meta-alarm in the current time window is larger than the threshold, it is defined as a timeout. The meta-alarm cannot participate in the merge, so the time window slides backward until all meta-alarms in the event window are within the threshold range. Each alarm pairs with a meta-alarm, and if the port IP and alarm type are the same, it is merged with the meta-alarm, resulting in a final output of the meta-alarm set MA.

The meta-alarm still has all the characteristics of all standard alarm formats. This process eliminates redundant data, avoids interference to the subsequent operation, and improves data quality.

Alarm correlation is a clustering process that deals with alarms and their relationship to obtain attack chains. The CluStream algorithm, proposed by Aggarwal et al. [21], aims to accomplish clustering quickly while ensuring the accuracy of clustering, and is utilized to correlate alarms. The algorithm contains two processes: online and offline clustering. In the online phase, the goal is to quickly scan data streams and generate a micro-set, which refers to the set of one-step attacks. In the offline phase, the main idea is to detect the IP set of two single-step attacks and determine whether the alarm type conforms to the multi-step attacks' characteristics.

The multi-step attack usually carries out a series of malicious behaviors with the jumpers' help, and the key to identifying the jumpers is whether there is an "end-to-end" IP set with each single-step attack. There are several points optimized for alarm correlation in this paper, as follows.

- Alarms' amount varies with time, so the number of clusters cannot be set directly. As the alarm streams increase, the number of micro-sets must be limited to control the calculation times of distance. The control method takes a boundary time to store the micro-collection and will not add new alarm data when timeout.
- CluStream uses a Pyramid timeframe structure to accelerate the query and a snapshot to store part of data instead of all data to reduce storage. In this case, each micro-set may not be retrieved and critical alarms may be omitted and cause correlation failures. To solve this problem and reduce the number of searches, in this paper a search strategy based on dichotomy is designed that takes the average triggering time of all alarms in a micro-set as its time and sorts each micro-set according to chronological order, which can be used with a binary strategy.

- The offline phase cluster micro-sets using the improved K-means algorithm depend on the pyramid timeframe structure's snapshot. In this paper, the micro-set does not have snapshots, but directly stores according to the order of timestamps. Therefore, only data before the time-critical point are needed to search, and all the data after the critical point will be included in the clustering algorithm for correlation.

After analyzing the CluStream algorithm flow and reforming, another core of the alarm-correlation algorithm is to design an alarm-attribute similarity function. For any two alarms, a measure of their similarity must be calculated based on their attributes. For attribute similarity of ports, alarm types, and packet types, the similarity value is 1 if they are equal or 0 otherwise. The similarity value must be converted into a 34-bit binary number for attribute IP address and then the maximum matching length from high to low is calculated. The ratio of this length to 32 is the similarity of IP.

For the attribute time, the closer the time, the greater the similarity. The characteristics of the time interval are different for different types of attacks. Therefore, a time-similarity formula based on the Sigmoid function is designed. α indicates the threshold at which the function falls and β the fall rate. The specific formula is expressed as follows:

$$d_{time}(x_i, x_j) = \frac{1}{1 + e^{\alpha+\beta|x_i.time - x_j.time|}}. \tag{1}$$

Accordingly, the formula for calculating the similarity of two alarms is

$$sim(x_1, x_2) = \frac{\sum_{i=1}^{k} W_i * d(x_1^i, x_2^i)}{\sum_{i=1}^{k} W_i}. \tag{2}$$

3.2 False-positive Detection

The attack chain is generated from all alarms process by the alarm correlation, but there are some meaningless alarm sequences called false-alarm sequences. When an alarm exists alone, it is impossible to determine whether it is a false alarm based on its category, IP port, or other properties. For example, one intranet node B sends an ICMP message to another intranet node C, which is very common in intranet environments, and one usually has low awareness of this message. However, before this event, an external node A carried out an overflow attack on B through software vulnerability and successfully obtained the system permission of B. Then, node A controlled node B to detect all the machines in the internal segment using ICMP. At this point, the ICMP sent by B to C may become a link in the attacker's behavior chain. Therefore, in this paper the characteristics that lead to false positives from the alarm set's perspective are analyzed based on the attack-chain context.

- **Attack duration.** From the time an attacker launched the detection to reach the goal, the time interval is an attack duration. After the clustering completes, the alarms will sort based on the timestamp. As can be known from analysis of a multi-step attack, the attacker will look for the next opportunity to invade again after each step is complete.
- **Attack interval.** When an attacker penetrates step by step to find nodes with possible vulnerabilities, the attack interval is often long. In a one-step attack, the detection and scanning behaviors generally produce many alarms in a brief period. Therefore, the time interval between multiple steps and the time interval within a single step is considered a feature.

- **Source of the attack.** The source of network intrusion is bound to come from an external system. When some abnormal behavior of the internal nodes is found, it often implants malicious software. Therefore, finding a high-frequency external IP at the start of the attack chain is the key to mining the attack sequence.
- **Sensitivity.** The attacker's target is the high-value data or important services within the network, so whether the nodes in the attack chain contain these targets is an important indicator with which to measure the sensitivity.
- **Frequency of attack types.** Most of the false positives in a network are generated by the regular service. With increasing time, the frequency will tend to be stable. However, the alarms generated by intrusion are uncommon, and the frequency of these false alarms is different from that of ordinary alarms.
- **Number of actions in attack chain.** In a network with a complex structure, it is difficult for an attacker to achieve the attack target directly. It may take several attempts to get close to the critical nodes, which means that the number of behaviors will also increase.
- **Threat of external IP.** All operations generated by a high-risk IP are suspect. Through monitoring some malicious IPs by security researchers, the determination of an IP's threat can introduce external knowledge. A threat can also be defined as the number of alarms calculated from an IP history.
- **Type of one-step attack.** In the real attack scenes, many attackers will use a scanning method for detection. Then, they may use a DDOS to attack after finally finding the target node. Thus, the type of single-step attack should be taken into consideration.

Based on the discussion above, in this paper LightGBM is used to determine whether an alarm sequence is a false positive. Let attack chain $s_i = m_1, m_2, m_3, \ldots, m_n$ represent the multi-step attack sequence, in which each step is a single-step attack. $m_i = a_1, a_2, a_3, \ldots, a_n$ contains the underlying meta-alarm. For each alarm set s_i, the formulations for calculating properties are shown in Tab. 1.

The feature operations of an alarm set depend on its internal statistical characteristics and global statistical information. With increasing alarms, the global statistics tend to be stable, and the changes in the distribution of data have a low impact on it, which is not conducive to reflecting the current network situation. Therefore, in this paper the time is restricted, the alarm data within one month are selected, and alarm-data updates are daily.

3.3 Attack-Scene Mining

Attack-scene mining extracts attack patterns from real alarm sequences for online detection and prediction. PrefixSpan [22] is commonly used in the production environment since it only must scan the database twice, and the projection database shrinks rapidly. It has more advantages in dense datasets, but there are still some shortcomings, as follows.

- The possibility of repeated projection increases with increasing data size, and the algorithm may establish multiple identical projection databases during scanning. In this paper, a hash table is used to store the scanned location to avoid this problem. As the scanning continues, the location information's length will rapidly decrease, and the storage cost is much less than the scanning cost.
- One generated projection sequence is discarded directly and no longer mined when its length adds the current prefix length is less than the minimum control length. This is because the final output series' length will not exceed its projection sequence, in which case one must perform pruning.

Table 1: Table of feature-calculation methods

Property name	Description of attributes	Formula for calculation
Feat_time	Attack duration	$m_n.time - m_1.time$
Feat_interval	Attack interval	$\frac{\sum_{n=1}^{n-1} m_{i+1}.time - m_i.time}{n-1}$
Feat_outside	Source of attack	$\frac{output\_ip\_num}{n}$
Feat_sensitive	Sensitivity	$\frac{value\_num}{n}$
Feat_frq	Frequency of attack types	$\frac{\sum_{i=1}^{n} frq(a_i)}{n}$
Feat_act_num	Number of actions in attack chain	$s_i.count$
Feat_threat	Threat of external IP	$\frac{threat_{ip_{set}} \cap s_i.ipset}{n}$
Feat_type	Type of one-step attack	$frq(m_i.type)$

- Every time a projected database is created, the sequence suffix must be fully copied, which incurs a significant amount of database overhead. Therefore, in this paper a position table is established instead of creating the projection database to avoid adding new storage space when scanning locations.
- The number of sequences in the projected database determines the calculation of the subsequent support degree. When the amount is less than the current support degree, subsequent mining will not generate a frequent sequence. Therefore, it will not create a projected database in this case.

The improved algorithm flow is shown as Algorithm 1.

Algorithm 1: PosPrefixSpan(a, S, a_pos, pre_tree, minsup)

Input: prefix a, project database (with prefix a) $S \mid a$, prefix tree *pre_tree*, minimum support *minsup*
Output: frequent sequence A

```
1       count_dict ← get_count(S | a)
2       for each a1 in count_dict do:
3           if count(a1) > minsup:
4              A.append(a1+a)
5           End if
6           a1_pos ← get_prefixPos(S | a, a1)
7           if a1_pos.len < minsup:
```

(Continued)

```
8              Continue
9            End if
10           if a1_pos is in pos_list:
11             pre_tree[a1] ← get_tree(pos_list, a1_pos)
12           End if
13           PosPrefixSapn(a1, S, a1_pos, pre_tree[a1], minsup)
14       End for
```

A hash structure and prefix tree structure based on location information are proposed herein to optimize the PrefixSpan algorithm. To reduce the construction of duplicate projection databases, the searched projection database's location must be saved. When searching the same location again, recursion mining is not needed again.

Fig. 2 shows how to use the hash table and prefix tree. After mining sequence <ab> in the sequence database, the location information for the projected database generated by <ab> is recorded in the format "Sequence ID_prefix location & Sequence ID_ prefix location…" The primary key of the hash database is position information. To reduce its length, only sequences with a suffix length greater than 0 will be stored. After the storage is complete, the prefix tree must be built according to the mined prefix sequence. The former item is the parent node of the following item, and the child nodes are generated recursively based on subsequent mining results. At this time in the hash database, location information is the key, and the value is the leaf node corresponding to the projected database. Then, when finding the prefix <b>, the location information corresponding to the projected database has been searched, the database is then directly interrogated, and the suffix will be connected.

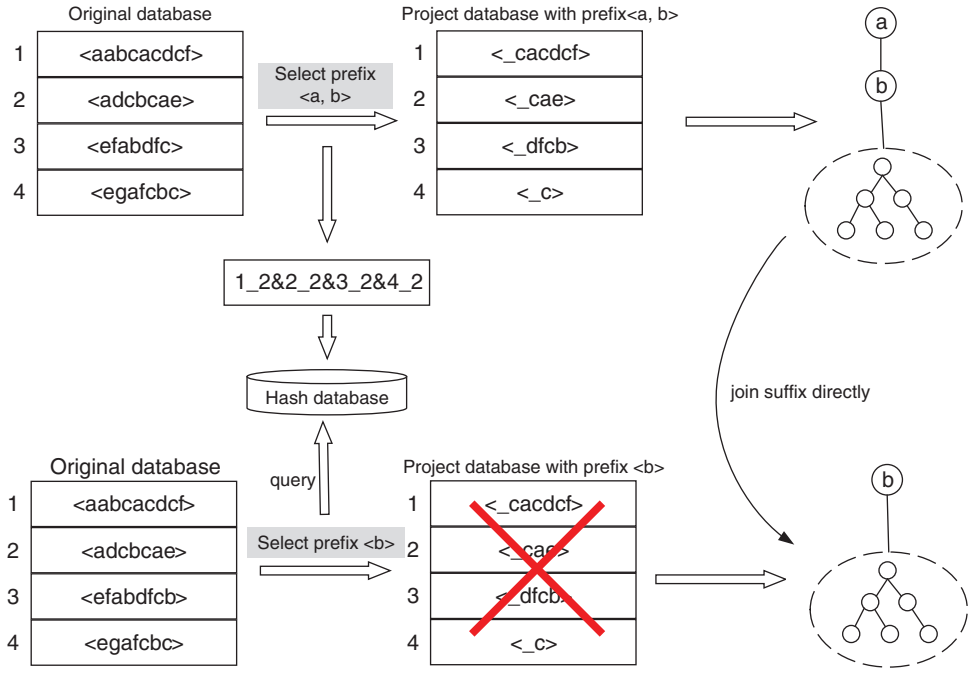

Figure 2: Hash table and prefix tree of improved PrefixSpan

In the improved PrefixSpan algorithm, the output is an attack tree based on sequence construction. This is done for two reasons: First, in mining of the same projection database, the results are often the same, so the result of the previous mining can be used directly; that is, directly connect a node and its leaf nodes. Second, the attack tree is a more visual description of the attack path. After obtaining an attack tree with frequent sequences, each path from the root node to the child node is an attack chain.

To better show the correlation between each attack, it is necessary to calculate the probability of state transition between attacks. The Markov theory, which can define the cause-and-effect correlation of alarm, is used to obtain the probability. A Markov chain represents the transition graph from one state to another. In this model, the sum of the transition probability of each state should be 1. The Markov model is forward-oriented, which means the next state only depends on the value of the previous one, as in the following equation:

$$p(x_{i+1} \mid x_i, x_{i-1}, \ldots, x_1) = p(x_{i+1} \mid x_i). \tag{3}$$

The transition matrix of the attack graph is calculated by counting the occurrence times of each alarm. For the two nodes connected in the alarm chain a_i and a_j, the transition probability is

$$p(a_j \mid a_i) = \frac{count(a_i) \rightarrow count(a_j)}{count(a_i)}. \tag{4}$$

Online detection uses the LCS method to compare the real-time online aggregated alarm sequence with the attack-pattern string. If the similarity is greater than the threshold value, it is considered an attack chain with high risk.

3.4 Attack-scene Prediction

After the above-described processing, the alarm sequence still has many properties, such as alarm type, IP and port, and time. However, in the alarm-sequence prediction, the target predicted is the attacker's next action. Therefore, only the alarm-type sequence is taken as the training data. The names of alarms are all strings that must be digitized. The digitization method encodes the alarm name and labels into integer numbers. After the encoding is complete, the next task is sequence-length processing. The data that the prediction model can receive are all of the same length, but the length is uneven for the real alarm sequence, so the sequence length L is defined as the length of the training data. Then, a sliding time window is used to obtain the training sequence of equal length, as shown in Fig. 3.

The length of the sliding window is L, each slide takes L as the training set, and the $L+1$ data are used as the label to generate data pairs. For sequences with a length less than L, 0 will be added in the header.

In this paper, LSTM, the most typical model in RNNs, is selected as the sequence prediction model. As shown in Fig. 4, the prediction network structure contains three layers: an embedding, LSTM, and dense layer.

Embedding layer. The input data are labeled sequence number of the alarm name, and the role of the embedding layer is to re-encode the alarm name and transform it into a vector. The similarity between vectors determines the similarity between words, indicating the connection between alarms.

LSTM layer. The LSTM layer is the core module for learning the features of an alarm sequence. Through the transfer of its cell state, the LSTM layer preserves its transfer

probability and finds the correlation before and after an alarm more accurately to complete the alarm-sequence prediction.

Dense layer. Also known as the full connection layer, the dense layer combines the features extracted by the LSTM layer. It outputs the probability of each tag through the activation function.

The final outputs are the next alarm probabilities, among which the alarm with the maximum probability is the prediction result.

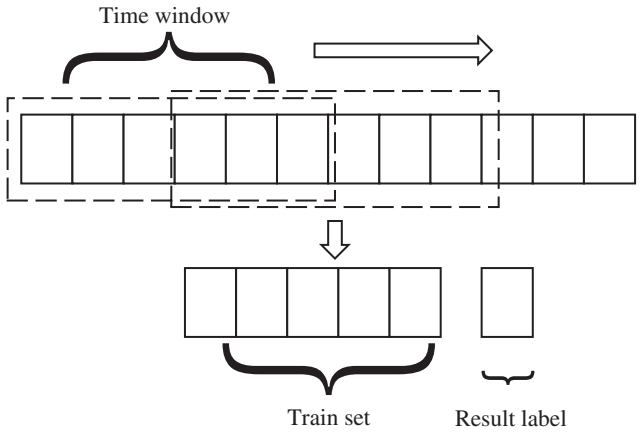

Figure 3: Sequence-cutting process

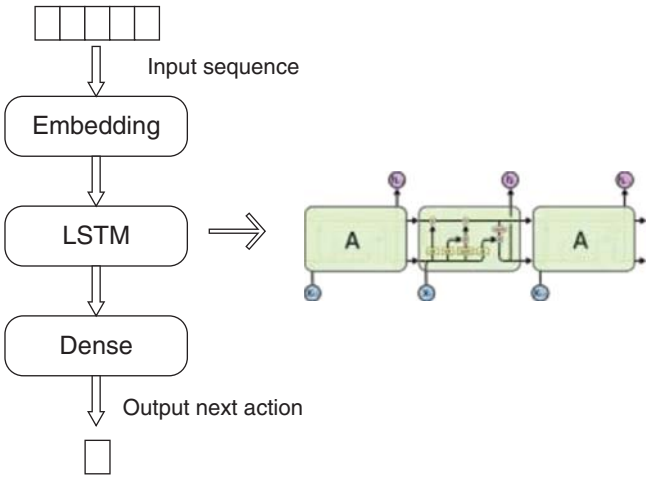

Figure 4: Prediction-network structure

4 Experimental Results

DARPA2000 and ISCX2012 were used in the present work as multi-step attack datasets and Snort 2.9 was employed to replay traffic with rule snortrules-snapshot-29130 since these datasets are raw traffic packets. In this section, the experimental results are described and analyzed

according to the following flow: alarm correlation, false-positive detection, attack-scene mining, and attack-scene prediction.

4.1 Alarm Correlation

Two results are combined in the alarm-correlation step. One is an alarm set used to calculate the features for false-positive detection that also contains detailed information such as time and IP. The other are the alarm sequences for frequent sequence mining. In this process, one can preliminarily find some attack steps, such as the sample attack procedures shown in Fig. 5.

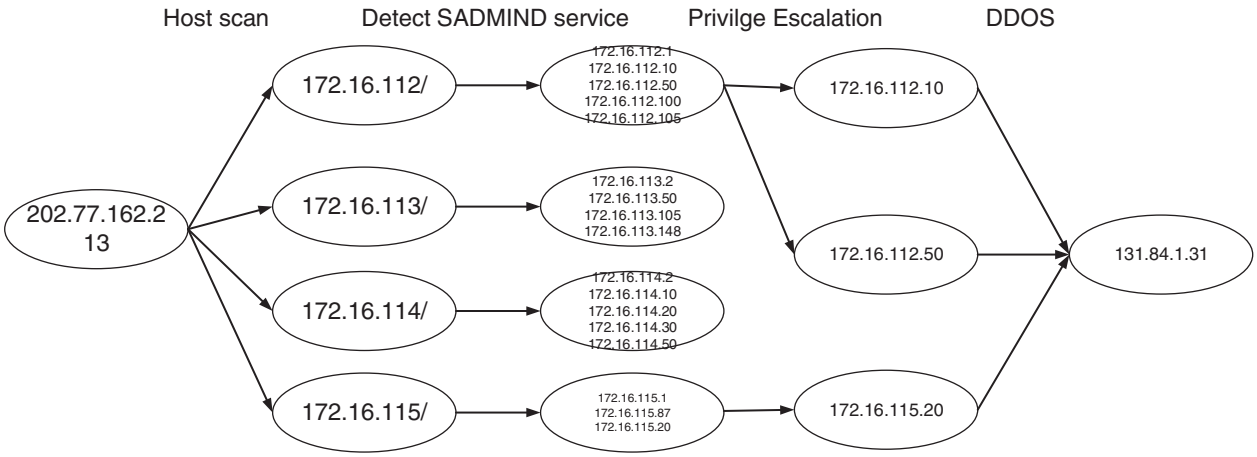

Figure 5: Sample attack process on DARPA dataset

4.2 False-positive Detection

In the false-positive-detection step, LightGBM is used to determine false positives of alarm sets; four other algorithms were chosen for comparison. The results on two datasets (DARPA and ISCX) are shown as Tabs. 2 and 3, respectively.

Table 2: Comparison of algorithms on DARPA dataset

Model	Accuracy	Precision	Recall	F1	AUC
LightGBM	**0.988**	**1.00**	**0.909**	**0.952**	**0.954**
XGBoost	0.967	0.984	0.876	0.924	0.943
SVM	0.957	0.958	0.696	0.807	0.846
DecisionTree	0.961	0.953	0.768	0.848	0.864
LogisticRegression	0.906	1.00	0.272	0.428	0.636

It can be seen that the performance of LightGBM is the best among the five models compared, and has a high false-alarm recognition rate. Overall accuracy is approximately 98% in the DARPA dataset and approximately 95.5% in the ISCX dataset.

Table 3: Comparison of algorithms on ISCX dataset

Model	Accuracy	Precision	Recall	F1	AUC
LightGBM	**0.955**	**1.000**	**0.571**	**0.727**	**0.785**
XGBoost	0.925	0.666	0.571	0.615	0.769
SVM	0.863	0.750	0.571	0.640	0.653
DecisionTree	0.895	0.500	0.571	0.533	0.752
LogisticRegression	0.915	0.750	0.428	0.545	0.705

Since the false-alarm detection described in this paper is trained based on the alarm context, the study of Yao et al. [23] was selected for comparison based on a single-alarm feature-clustering model. The comparison results are shown in the Tab. 4.

Table 4: False-positive detection comparison

Method	Dataset	Accuracy (%)
Present paper	DARPA	**98.8**
Yao et al.	DARPA	92.1
Present paper	ISCX	**95.5**
Yao et al.	ISCX	90.2

4.3 Attack-scene Mining

In the attack-scene-mining step, the goal is to find frequent sequences on the correlated alarms and extract the attack pattern. In this paper, the attack-scene-mining algorithm is implemented by optimizing PrefixSpan. Experiments were carried out based on four different attributes. The selected dataset is from the IBM Quest Synthetic Data Generator (https://sourceforge.net/projects/ibmquestdatagen/) program.

Fig. 6 shows a comparison between the improved and original algorithms. The factors that affect algorithm performance include the degree of support, number of data rows, average sequence length, and number of item sets. The runtimes of the two algorithms are very close when the conditions are loose. However, as the conditions become stricter or the amount of data becomes larger, the improved algorithm becomes increasingly faster than the original.

It is necessary to count the occurrence times and transition probability based on Bayesian theory for each alarm. The attack-scene-mining results are in the form of alarm chains. Owing to the large number of alarm types, only some primary state-transition diagrams were selected in this experiment. Fig. 7 is a probability attack diagram for the DARPA dataset.

Each node in the figure represents a type of attack, the arrow direction indicates the attack order, and the value on the arrow indicates the transition probability from one to another. A DDOS attack on a target is depicted, as can be seen from the transition process with the help of a jump board through one-step detection.

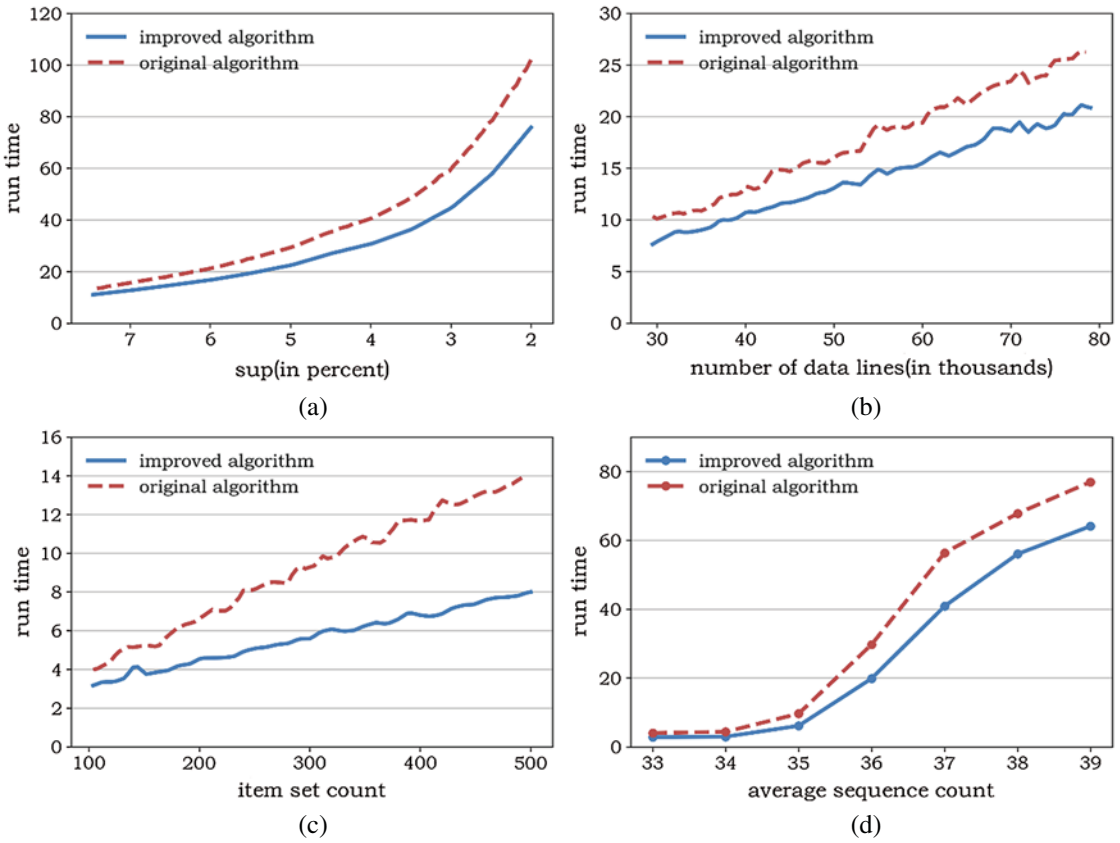

Figure 6: Algorithm performance comparison under different conditions (a) Under different support levels (b) Under different row numbers (c) Under different sequence lengths (d) Under different item categories

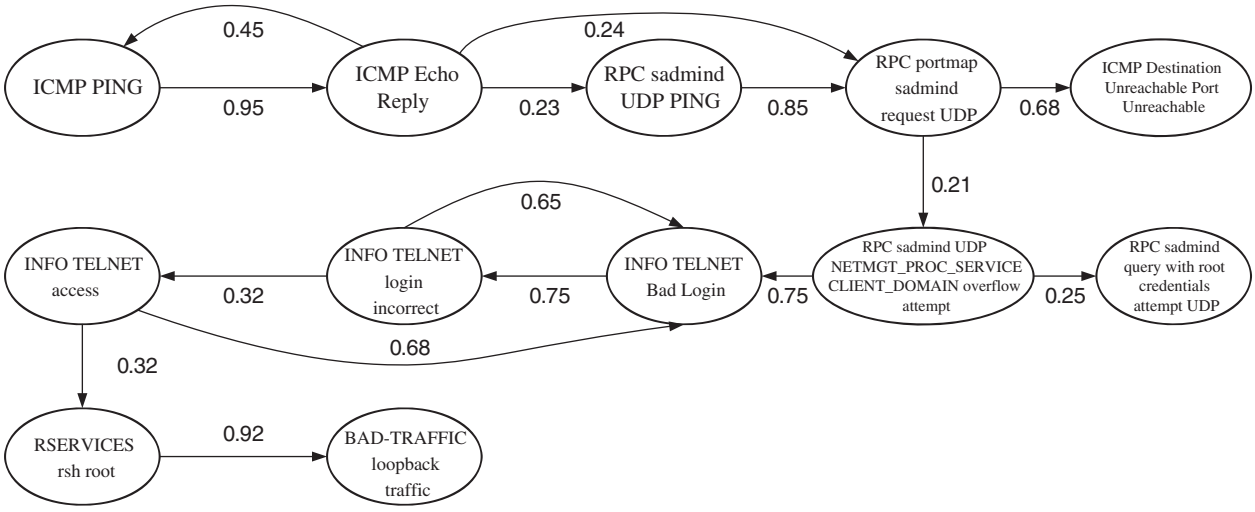

Figure 7: Schematic of attack process on DARPA dataset

4.4 Attack-scene Prediction

In the final step, attack-scene prediction, attack patterns from scene mining were used as training data for LSTM. Since the sequence set length is mostly between 0 and 10, the intercepted window length was set to 5. Besides, if the sequence length is less than 5, 0 is added to the header. In this experiment, 70% of the data were randomly selected as the training set and 30% as the verification set. The accuracy rate is then calculated for different sequence lengths. Owing to the different number of steps, the prior knowledge given will vary, and the accuracy of the prediction is slightly different. Fig. 8 shows the main results.

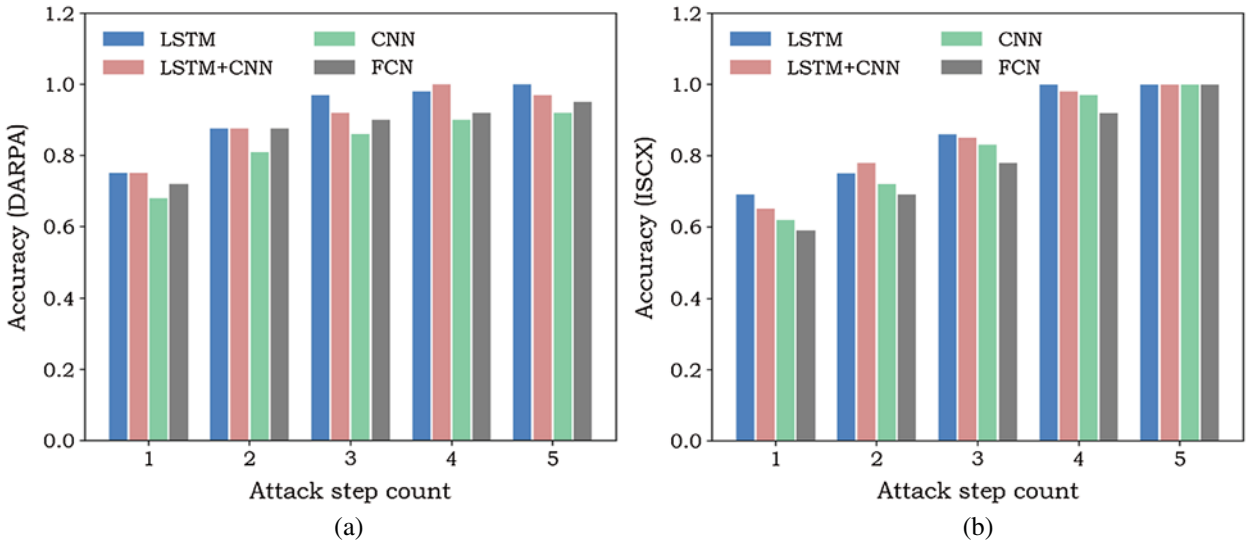

Figure 8: Attack-scene-prediction results on two different datasets (a) Prediction result with DARPA dataset (b) Prediction result with ISCX dataset

The figure shows the accuracy of four different algorithms under a different number of attack steps. It is clear that when there is only one alarm, it is difficult to determine what the next attack step is. With the gradual increase and completeness of the attack, the prediction accuracy also increases gradually, finally reaching an accuracy rate of 98%. LSTM + CNN also presented a good effect, but compared with LSTM alone, the model is much more complex and its accuracy is not qualitatively improved. Therefore, the LSTM algorithm was selected.

Perry et al. [24] also used LSTM to predict attacks from alarm data with single-layer LSTM, without an embedding method. For comparison, in the present work 80% was selected as the test set and 20% as the validation set for training. The average accuracy rate in prediction instead of the accuracy rate for each step was then calculated.

As the results show in Tabs. 5, the method presented in this paper exhibits good performance on the two datasets, reaching an average accuracy of 85.3% on DARPA and 82.5% on ISCX.

Table 5: Attack-scene prediction comparison

Method	Dataset	Average accuracy (%)
Present paper	DARPA	**85.3**
Perry *et al*.	DARPA	81.2
Present paper	ISCX	**82.5**
Perry *et al*.	ISCX	79.6

5 Conclusion

In this paper, the problem of multi-step attacks hidden in massive alarm data is studied, and a method proposed for alarm correlation, revealing attack patterns, and the online prediction of attacks, thus providing an effective way to protect network systems from multi-step attacks. By improving and optimizing the key algorithms for the steps of alarm correlation, false-alarm detection, attack-scene mining, and attack-scene prediction, the proposed method exhibits good performance and accuracy. This study has some limitations, and planned future work includes exploring ways to reduce the impact of alarm misses and to optimize those multi-step attacks with particularly long time spans.

Funding Statement: This work is supported by the National Key R&D Program of China (2016QY05X1000) and the National Natural Science Foundation of China (Grant No. 201561402137).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] W. Wu, R. Li, G. Xie, J. An, Y. Bai *et al.* "A survey of intrusion detection for in-vehicle networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 919–933, 2020.

[2] A. Badshah, A. Ghani, M. A. Qureshi and S. "Shamshirband "Smart security framework for educational institutions using internet of things (iot)," *Computers, Materials & Continua*, vol. 61, no. 1, pp. 81–101, 2019.

[3] C. Lv, J. Zhang, Z. Sun and G. Qian, "Information flow security models for cloud computing," *Computers*, *Materials & Continua*, vol. 65, no. 3, pp. 2687–2705, 2020.

[4] S. Su, Z. Tian, S. Liang, S. Li, S. Du *et al.* "A reputation management scheme for efficient malicious vehicle identification over 5G networks," *IEEE Wireless Communications*, vol. 27, no. 3, pp. 46–52, 2020.

[5] C. Qian, X. Li, N. Sun and Y. Tian, "Data security defense and algorithm for edge computing based on mean field game," *Journal of Cyber Security*, vol. 2, no. 2, pp. 97–106, 2020.

[6] Z. Tian, C. Luo, J. Qiu, X. Du and M.Guizani, "A distributed deep learning system for web attack detection on edge devices," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1963–1971, 2020.

[7] S. Garg, N. Kumar, J. J. Rodrigues and J. J. Rodrigues, "Hybrid deep-learning-based anomaly detection scheme for suspicious flow detection in SDN: A social multimedia perspective," *IEEE Transactions on Multimedia*, vol. 21, no. 3, pp. 566–578, 2019.

[8] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su *et al.* "A survey on access control in the age of internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020.

[9] M. Li, Y. Sun, H. Lu, S. Maharjan and Z. Tian, "Deep reinforcement learning for partially observable data poisoning attack in crowdsensing systems," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6266–6278, 2020.

[10] Z. Tian, S. Su, W. Shi, X. Du, M. Guizani *et al.* "A data-driven method for future internet route decision modeling." *Future Generation Computer Systems*, vol. 95, no. 1, pp. 212–220, 2019.

[11] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim *et al.* "A survey of deep learning-based network anomaly detection," *Cluster Computing*, vol. 22, no. 1, pp. 949–961, 2019.

[12] J. Navarro, A. Deruyver and P. Parrend, "A systematic survey on multi-step attack detection," *Computers Security*, vol. 76, no. 1, pp. 214–249, 2018.

[13] L. B. Qiao, B. F. Zhang, Z. Q. Lai and J. S. Su, "Mining of attack models in ids alerts from network backbone by a two-stage clustering method," in *IEEE 26th Int. Parallel & Distributed Processing Symp. Workshops & Phd Forum IEEE Computer Society*, shanghai, SHH, China, pp. 1263–1269, 2012.

[14] P. Ning, "TIAA: A visual toolkit for intrusion alert analysis," *in Technical Report*, vol. 1, pp. 1–20, 2013.

[15] F. Kavousi and B. Akbari, "A Bayesian network-based approach for learning attack strategies from intrusion alerts," *Security and Communication Networks*, vol. 7, no. 5, pp. 833–853, 2014.

[16] S. Noel, E. Robertson and S. Jajodia, "Correlating intrusion events and building attack scenes through attack graph distances," *in Computer Security Applications Conf.*, Tucson, AZ, USA, pp. 350–359, 2004.

[17] Y. Luo, F. Szidarovszky, Y. Al-Nashif and S. Hariri, "A fictitious play-based response strategy for multistage intrusion defense systems," *Security and Communication Networks*, vol. 7, no. 3, pp. 473–491, 2014.

[18] A. A. Ramaki, M. Amini and R. E. Atani, "Rteca: Real time episode correlation algorithm for multi-step attack scenes detection," *Computers Security*, vol. 49, no. 1, pp. 206–219, 2015.

[19] H. H. Pajouh, R. Javidan, R. Khayami, A. Dehghantanha and K. K. R. Choo, "A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks," *IEEE Transactions on Emerging Topics in Computing*, vol. 7, no. 2, pp. 314–323, 2019.

[20] W. Liang, K. -C. Li, J. Long, X. Kui and A. Y. Zomaya, "An industrial network intrusion detection algorithm based on multifeature data clustering optimization model," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2063–2071, 2020.

[21] C. C. Aggarwal, J. Han, J. Wang, P. S. Yu and R. Ctr, "A framework for clustering evolving data streams," in *Proc. 2003 VLDB Conf.*, Berlin, BER, Germany, pp. 81–92, 2003.

[22] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto and M. Hsu,"Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth,– *in Int. Conf. on Data Engineering, Heidelberg*, Baden-Wuerttemberg, Germany, pp. 215–224, 2001.

[23] Y. Yao, Z. Wang, C. Gan, Q. Kang, X. Liu *et al.* "Multi-source alert data understanding for security semantic discovery based on rough set theory," *Neurocomputing*, vol. 208, no. 1, pp. 39–45, 2016.

[24] I. Perry, L. Li, C. Sweet, S. -H. Su, F. -Y. Cheng *et al.* "Differentiating and predicting cyberattack behaviors using lstm," in *IEEE Conf. on Dependable and Secure Computing*, Taiwan, TW, China, pp. 1–8, 2018.