Mining Attack Correlation Scenarios Based on Multi-agent System

Sisi Huang, Zhitang Li, and Li Wang

Computer Science Department
Huazhong University of Science and Technology, Hubei Wuhan 430074, China
Huangsisi1984@gmail.com,leeying@hust.edu.cn,wtwl@hust.edu.cn

Abstract. Nowadays, one very complicated problem bothering network analysts too much is the redundant data generated by IDS. The objective of our system SATA (Security Alert & Threat Analysis) is trying to solve this problem. Several novel methods using data mining technologies to reconstruct attack scenarios were proposed to predict the next stage of attacks according to the recognition the attackers' high level strategies. The main idea of this paper is to propose a novel idea of mining "complicated" attack scenarios based on multi-agent systems without the limitation of necessity of clear attack specifications and precise rule definitions. We propose SAMP and CAST to mine frequent attack behavior sequences and construct attack scenarios. We perform a series of experiments to validate our method on practical attack network environments of CERNET. The results of experiments show that our approach is valid in multi-agent attack scenario construction and correlation analysis.

Keywords: correlation analysis; attack scenario; frequent attack sequence.

1 Introduction

Nowadays, more and more intrusion detection systems (IDSs) such as intrusion detection system, firewall, anti-virus software, vulnerability scanner and VPN etc are deployed to defend attacks against enterprise networks. Unfortunately, these different security sensors provide not only wealthy information but also a large volume of security data to network administrator. Therefore, it is important to develop a network security correlation system whose functions are reducing the redundancy of alarms, correlating different alerts, constructing attack scenarios, discovering attack strategies and predicting the next step of attacks.

In this paper, we focus on the attack scenario construction module. Attack scenario construction module is a very important component of such systems because it can help forensic analysis, response and recovery, and even prediction of forthcoming attacks. However, there have been several proposals on alert correlation, but most of these proposed approaches are lack of flexibility and depend on complex correlation rules definition and hard-coded domain knowledge that lead to their difficult implementation.

© Springer-Verlag Berlin Heidelberg 2007

Based on the substantial researches about attack behaviors for several years, we discovered the features of attack behaviors in the same attack strategy are: sequence and frequency. We proposed novel methods to discover attack strategies via mining frequent patterns.

1.1 Related Work

The research of the intrusion detection has started for many years, and some researchers have already developed several practical approaches to facilitate analysis of intrusion. In [1], Valdes used a probabilistic method to correlate alerts according to constitute similarity metric between their features. In [2], Wenke Lee and Xinzhou Qin proposed a GCT-based and Bayesian-based correlation approach without the dependence of the prior knowledge of attack transition patterns. Debar,H and Wespi use a consequence mechanism[3] to specify that types of alerts may follow a given alert type. In [4], Templeton and Levitt proposed an attack modeling language based on prerequisite consequence relation called JIGSAW. Ning in [6] also developed a method considered as a variation of JIGSAW. The focuses of these methods are both to uncover attack scenarios based on specification of individual attacks, and the main idea of the approach is also used by Cuppens and Meige in their work of MIRADOR[7].

Although these approaches mentioned above can potentially discover the relationship between alerts, most of these approaches have limited capabilities because they rely on predefined knowledge of attack conditions and consequences. They lack of the capabilities of recognizing a correlation when an attack is new or the relationship is new. The method this paper mentioned of construct attack scenario through attack sequence and frequency pattern mining method is enlightened by Jian Pei, who introduced a data mining method in [8] to mine the sequence and frequency patterns in a database.

1.2 Organization of the Paper

In this work we present the approach we suggest implementing the scenario construction function. The remainder of this paper is organized as follows. Section 1 introduces the main object and related work. Section 2 introduces the overview of our work and the framework of SATA system. Section 3 presents the problem of mining attack sequence patterns and some concepts used in the problem. Section 4 proposes algorithms to construct attack scenario. Section 5 reports our experiments on the branch of CERNET. In section 6 we conclude with a summary and directions for future work.

2 Overall of Our Work

In this section, we present an overview of our system SATA, which aims to provide a platform for integrated network security data management. Figure 1 presents the main principles we suggest to develop a security event management system SATA (Security alerts & Threats analysis) for intrusion detection. There are six main

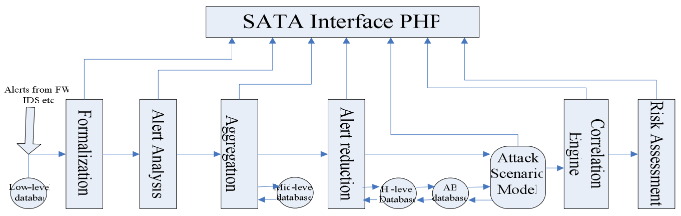


Fig. 1. Framework of SATA

modules in this system: Formalization, alert analysis, aggregation, alert reduction, correlation, and risk assessment. The main target of SATA is to build an integrated and centralized platform of security event management and analysis.

Formalization: SATA receives the alerts generated by different IDS and stores them into the low-level alert database after normalization and format standardization procedures for further analysis.

Alert analysis: provides scores of reliability, priority, asset and a final assessment to the degree of threat for each alert, to indicate how much it poses threat to the current protected network from various aspects.

Aggregation: the modules tries to aggregate the incoming alert with existing alert cluster or single alerts in the alert clusters which are sets of alerts that correspond to the same occurrence of an attack.

Alert reduction: the main component is a filter whose function is to filter the false positives and low-interest alerts.

Correlation engine: contains a construct attack scenarios model that we can recognize the attack strategies by matching attacks' behaviors to attack scenarios.

Risk assessment: the module aims to assess the real-time security state of the protected networks or hosts through a particular risk assessment mechanism.

As mentioned in the above introduction, we will only suggest a specification for the alert management, correlation module. The other modules which are briefly sketched in the conclusion of this paper are not presented.

3 Problem Statement

There are 4 databases in SATA: *low-level* alert database which contains raw alerts, *mid-level* alert database whose contents are the alerts after aggregation process, *and hi-level* alert database which only contains highly improved alerts after alert analysis, reduction and alert aggregation process. The database we are concerned is the *hi-level* alert database, whose alerts are largely less than other database, and also the quality of the alerts in *hi-level* database is more improved. The last database is *AB* database (attack behavior database), which deposits the result of attack behavior analysis from hi-level database.

Each alert record in database *hi-level* consists of the following attributes alert-id (id-number, sensor-id, and signature-id), attack type, timestamp (happen-time, end-time), source (source-IP, source-port), destination (destination-IP, destination-port), risk (reliability, asset, and priority), and protocol.

An itemset is a non-order and non-empty set of items. Without loss of generality, we assume that each alert as an item, mapped to contiguous integers. The itemset in our paper is denoted as a set of attack behaviors reported by different IDS sensors at the same time. We denote these itemsets i by $(i_1, i_2, ... i_m)$, where i_k is an item. A sequence is an ordered list of itemsets. We denote a sequence s by $\{ s_1, s_2, ... s_m \}$, where s_j is an itemset. We also denote 3 sequences: time sequence T $\{ t_1, t_2, ... t_m \}$ where t_i is a time-stamp representing the attack occurring time; alert-id sequence $\{ a_1, a_2, ... a_m \}$, which represent an alert ID. Attack behavior sequence B $\{ b_1, b_2, ... b_m \}$, where b_i is an attack behavior cluster itemset corresponds to different attack behavior types happened in the same time. Attack behavior itemset was denoted as b_i $(k_1, k_2, ... k_m)$, where k_i is a behavior-id item. In a set of sequences, a sequence S is maximal if S is not contained in any other sequence.

All the attack behavior item in *hi-level* database can be viewed as a global attack behavior sequence, and the sorted attack behavior item can be ordered by the increasing time-stamp $t_1, t_2, ... t_m$, which are contained in the time sequence T.

The problem of mining attack behavior correlation scenario is to find the maximal attack sequences among all global attack sequences that have a certain user-specified minimum frequency and a time order. Each such maximal attack sequence represents a frequent *attack sequence*

EXAMPLE Consider the database shown in Table.1. Because of abundant data in database, we focus on 20 of the whole datum which multi-agents generate. This database has been ordered by real attack occurring time sequence $T\{ t_1, t_2, ... t_m \}$. The alert ID describes attack behaviors in global sequence ordered by attack occurring time. We map the alerts to the integer signature ID because we only concentrate on the attack behavior type attributes. It is more effective to mine attack correlation scenario and save the cost of string matching.

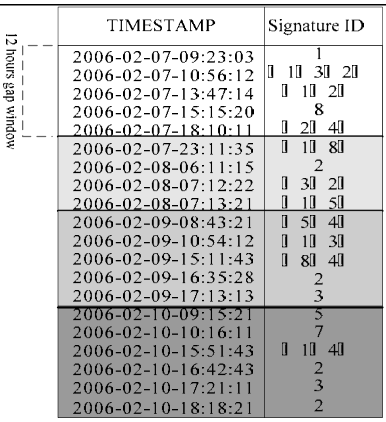
4 Finding a Sequence Pattern

4.1 Transform Global Attack Items to Sequences

As mentioned in figure 1, attack behaviors after formalization module are arranged in the *hi-level* database by the order of attack occurring timestamp. We found that most of attackers usually complete their attacks in 12 hours based on our long time experience and analysis. Because the time interval between the first step of an attack and the last step is usually in a certain time interval, we divide the global attack sequence according to sliding window gap definition. The range of sliding window gap is denoted as the attack time interval. In other words, attacks during one time interval will probably belong to the same attack correlation scenario. We define time interval attack sequences as *Si*. We also divide one day into 2 parts: active-time from 8:00-20:00 and rest-time from 20:00-8:00(the next day).

Table 1. Ordered alerts segment

TIMESTAMP	Alert ID	Signature ID
2006-02-07-09:23:03	1	1
2006-02-07-10:56:12	2	1▯ 3▯ 2
2006-02-07-13:47:14	3	1▯ 2
2006-02-07-15:15:20	4	8
2006-02-07-18:10:11	5	2▯ 4
2006-02-07-23:11:35	6	1▯ 8
2006-02-08-06:11:15	7	2
2006-02-08-07:12:22	8	3▯ 2
2006-02-08-07:13:21	9	1▯ 5
2006-02-09-08:43:21	10	5▯ 4
2006-02-09-10:54:12	11	1▯ 3
2006-02-09-15:11:43	12	8▯ 4
2006-02-09-16:35:28	13	2
2006-02-09-17:13:13	14	3
2006-02-10-09:15:21	15	5
2006-02-10-10:16:11	16	7
2006-02-10-15:51:43	17	1▯ 4
2006-02-10-16:42:43	18	2
2006-02-10-17:21:11	19	3
2006-02-10-18:18:21	20	2


12 hours gap window

TIMESTAMP	Signature ID
2006-02-07-09:23:03	1
2006-02-07-10:56:12	▯ 1▯ 3▯ 2▯
2006-02-07-13:47:14	▯ 1▯ 2▯
2006-02-07-15:15:20	8
2006-02-07-18:10:11	▯ 2▯ 4▯
2006-02-07-23:11:35	▯ 1▯ 8▯
2006-02-08-06:11:15	2
2006-02-08-07:12:22	▯ 3▯ 2▯
2006-02-08-07:13:21	▯ 1▯ 5▯
2006-02-09-08:43:21	▯ 5▯ 4▯
2006-02-09-10:54:12	▯ 1▯ 3▯
2006-02-09-15:11:43	▯ 8▯ 4▯
2006-02-09-16:35:28	2
2006-02-09-17:13:13	3
2006-02-10-09:15:21	5
2006-02-10-10:16:11	7
2006-02-10-15:51:43	▯ 1▯ 4▯
2006-02-10-16:42:43	2
2006-02-10-17:21:11	3
2006-02-10-18:18:21	2

Fig. 2. Transformation time slide window in *hi-level* database for generate sequential parts

Shown as Figure 2, the global attack sequence has been divided by sliding window to generate several sequential parts. We use these sequential parts to represent the attack behaviors occurring in each certain time interval. Each sequential part represents an attack behavior sequence. The problem of mining attack correlation scenario in *hi-level* database transforms to mine the frequent attack sequence from these attack sequential parts in *AB* database.

Terminology from prefixspan[17] The number of instances of items in a sequence is called the **length** of the sequence. A sequence with length l is called an l-sequence. Suppose all the items in an element are listed alphabetically.

Prefix: Given a sequence $\alpha=\{e_1,e_2,\ldots e_n\}$, a sequence $\beta=\{e'_1,e'_2,\ldots e'_m\}(m\le n)$ is called a prefix of α if and only if (1) $e'_i=e_i$ for $(i\le m-1)$; (2) $e'_m\subseteq e_m$ and all the items in ($e_m-e'_m$) are alphabetically after those in e'_m .

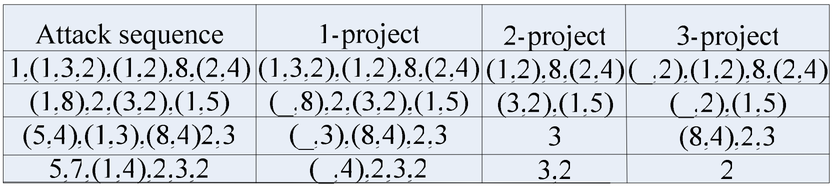
Postfix: $\beta=\{e_1,e_2,\ldots,e_{m-1},e'_m\}(m\le n)$ is α prefix sequence, sequence $\gamma=\{e''_m,e_{m+1},\ldots,e_n\}(m\le n)$ is called postfix of α . Prefix β denoted as $\gamma=\alpha\,|\,\beta$, where $e'_m=(e_m-e'_m)^2$.

4.2 Mining Frequent Attack Sequences

Find length-1 sequential patterns: Scan *AB* database once to find all the frequent items in sequences. Each of these frequent items is a length-1 sequential pattern. One item is frequent if its occurrence in these sequences is greater than a threshold min_frequency. In table 2, the length-1 frequent items are 1, 2, 3, 4, 5, 8.

Search space partition: The complete set of sequential patterns can be partitioned into the following six subsets according to the six prefixes: the ones having prefix 1,2,3,4,5,8. The other parts of the sequence are named postfix subsequence.

Table 2. 1-project sequence pattern



Attack sequence	1-project	2-project	3-project
1,(1,3,2),(1,2),8,(2,4)	(1,3,2),(1,2),8,(2,4)	(1,2),8,(2,4)	(_,2),(1,2),8,(2,4)
(1,8),2,(3,2),(1,5)	(_,8),2,(3,2),(1,5)	(3,2),(1,5)	(_,2),(1,5)
(5,4),(1,3),(8,4)2,3	(_,3),(8,4),2,3	3	(8,4),2,3
5,7,(1,4),2,3,2	(_,4),2,3,2	3,2	2

Table 2 presents the results of the divide sequence patterns by the length-1 1-project sequence pattern.

Find subsets of sequential patterns: After the process of dividing search place, finding all length-2 sequential patterns is our job. We construct corresponding projected databases to mine the subsets of sequential patterns recursively. When getting the length-1 sequential patterns, we can mine the length-2 sequential patterns.

The difference between the real data collected from different sensors to the *AB* database and our examples mentioned above is the amount of real attack behaviors in *AB* database is more than that we mentioned. Notice that the steps of practical attack are mostly more than 3 steps and less than 10 steps, the range of sequential patterns is set as 3 to 10. Differ from the prefixspan, we do not take attention at all sequential pattern which the algorithm could get, but focus on the long length patterns. The projected attack sequential patterns founded in Table 2 are list in Table 4.

Table 3. Attack sequential patterns

prefix	Attack sequential patterns
1	{1,(3,2),1} {1,3,1} {1,3,2} {(1,3),8,2} {1,2,1} {1,2,3} {1,2,2} {1,8,2}
2	NULL
3	{3,8,2}
4	{4,3,2} {4,2,3}
5	{5,1,3} {5,1,2} {5,1,2,3} {5,3,2} {5,2,3} {5,4,2} {5,4,2,3}
8	{8,2,3}

In table 3, we consider that 1-prefix sequential pattern contain some sequences that contain itemset. Take {1,(3,2),1}for example, it means that 2 sensors generate 2 alerts in the same time. To construct attack scenario effectively and accurately, we divide the sequence into 2 sequence: {1,3,1}, {1,2,1}.

SAMP The SAMP (Sequence attack behavior mining based on prefixspan) algorithm is described as follows:

```
Algorithm (SAMP)
Input: a sequence database AB, the min_frequency, window gap
       active-time, window gap rest-time
Output: The frequent attack behavior sequence
Method:
1. Scan the hi-level database, order the attack behaviors by
   time-stamp;
```

```
2. foreach the timestamp, put the attack behaviors itemsets
   among the interval of windows in a sequence labeled by
   sequence id;
3. put attack behavior sequence into AB database;
4. Call PrefixspanSome({},0,S,min,max);
```

Subroutine prefixspanSome(b, l, $S|_b$, min, max)

Parameters: b: a sequential pattern; l: the length of b;
$S|_b$: b–projected database, if $b \neq \{\}$; otherwise, the sequence database S. max: the maximal length of the patterns; min: the minimal length of the patterns ,

Method:

```
1. Scan $S|_b$ once, find the set of frequent items a such that
   a or {a} can be append to the prefix b to form a
   sequential pattern
2. Append a to b, to form a new sequential prefix pattern b′;
    if ($l \neq min$) JUMP 3;else { scan b′
     if b′ contain itemset, output b′ and replace the itemset
     to each item in it once;otherwise output b′;
3. Construct prefix b′-projected database $S|_{b'}$
4. If $l \leq max$, call prefixSpansom(b′,l+1, $S|_{b'}$)
     Otherwise subroutine termination.
```

4.3 Construct Attack Scenario Tree

The process of constructing AS-tree (Attack Scenario tree) can be more effective for recognizing the strategies of attacks. We represent every length-1 frequent item which can be found in attack sequential patterns as a root note. After setting each length-*l*+1 item as the child of length-*l* item, the branch of AS-tree is ordered by the frequency of the first embranchment item of each branch of the AS-tree. The character of the AS-tree is the frequency of first embranchment item in right branch is always larger than left, what can make the matching phrase much more effectively.

Algorithm CAST (Construct Attack Scenario Tree) can be described as follows

Input: attack sequence patterns
Output: Its sequence and frequent pattern tree, AS-tree
Method: The AS-tree is constructed in the following steps.

```
1. Scan the attack sequence patterns database once, arrange the
   sequence patterns by the length-1 prefix frequency in order.
2. Create the root of an AS-tree T, and label it as "NULL".
3. Call insert_tree([S/a],T)
4. delete T.
```

Subroutine Insert_AStree([*S*/*a*],*T*)
Parameters: [*S*/*a*]: a is the first element and S is the remaining list

```
1. if T has a child b such that b.name=a.name, then
   b.frequency=a.frequency+b.frequency
2. else create a new node b, make b.frequency=a.frequency, its
   parent link be linked to T as right child of T.
```

```
3. if b have a child c formerly, compare a.frequency and
   c.frequency,
4. if a.frequency ≥ c.frequency exchange a note with b sub-
   branch, jump to step 3,otherwise do not change;
5. if S is nonempty, call insert_tree(S,b)
```

Example: Figure 3 represents the AS-tree attack sequence patterns.

5 Recognize Attack Strategies

After constructing a sequence frequent attack tree from history data, the next step of our work is recognizing attack steps. When AB database receives an alert, we calculate the correlativity and priority to match with certain attack sequence tree easier.

Definition. Cor-degree: $Cor(h_i,h_j)$ $h_i,h_j \in H,(1 \le i,j \le n)$. Alert h_i,h_j is described by p attributes: $x_1,x_2,......x_p$, $y_1,y_2,......y_p$ respectively. The correlativity between h_i,h_j is described as:

$$Cor(h_i,h_j) = \frac{\sum_{i,j=1}^{p} w_{ij} Cor(x_i,y_j)}{\sum_{i,j=1}^{p} w_{ij}}.$$

Definition. pre-degree: $Pre(h_i,h_j)$, $h_i,h_j \in H,(1 \le i,j \le n)$. Alert h_i,h_j is described by p attributes $x_1,x_2,.....x_p$, $y_1,y_2,.....y_p$ respectively. The correlativity between h_i,h_j is described as:

$$Pre(h_i,h_j) = \frac{\sum_{i,j=1}^{p} k_{ij} Pre(x_i,y_j)}{\sum_{i,j=1}^{p} k_{ij}}$$

The weighted values of w_{ij} and k_{ij} are set empirically and can be tuned in practice. The formulation $Cor(h_i,h_j)$ defines the degree of matching between a new alert and a history alert in a attack scenario and the formulation $Pre(h_{i-1},h_j)$ describe the degree of correlation between the new attack action and the previous attack behavior of the possible-matched alert in the attack scenario. When the value of $Cor(h_i,h_j)$ and $Pre(h_{i-1},h_j)$ are more than certain thresholds, we consider the new attack is belong to a scenario and then predict next possible attack behavior according to the known attack patterns and report the security warning reports to users.

6 Experiments

Experiments 1: To evaluate the effectiveness of our techniques, we conducted the experiments in a branch of CERNET(China Education and Research Network).We ran the SATA system that persisted for 4 weeks to collect history data for SAMP algorithm, and then continue our experiment for 2 weeks to evaluate our method of constructing attack scenarios. During four weeks test, our system received 96300 alerts and then reduced to 2369 after the process of aggregation and verification phase, and last constructed 42 attack scenarios.

The final result of attack scenario construct of alerts in the experiment was presented in Figure4. Newattack number is denoted the amount of the new attacks which IDS generated and the newattack time is denoted three of the latest alerts which has correlation with incidents contained in a attack scenario.

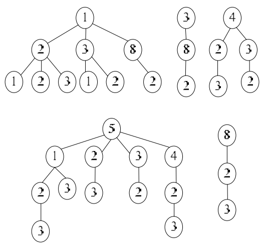
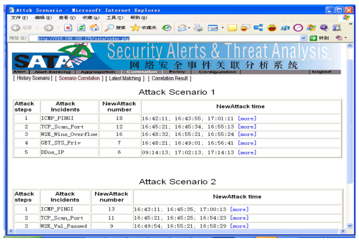


Fig. 3. AS-tree attack sequence


Attck Scenario - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
http://218.199.102.176/sata/index.php
SATA
Security Alerts & Threat Analysis
网络安全事件关联分析系统
Alert | Alert-Ranking | Aggregation | Correlation | Policy | Configuration | Logout
[History Scenario] [Scenario Correlation] [Latest Matching] [Correlation Result]

Attack Scenario 1

Attack steps	Attack Incidents	NewAttack number	NewAttack time
1	ICMP_PINGI	18	16:42:11; 16:43:55; 17:01:11 [more]
2	TCP_Scan_Port	12	16:45:21; 16:45:34; 16:55:13 [more]
3	W2K_Wins_Overflow	16	16:48:32; 16:55:21; 16:55:24 [more]
4	GET_SYS_Priv	7	16:48:21; 16:49:01; 16:56:41 [more]
5	DDos_IP	6	09:14:13; 17:02:13; 17:14:13 [more]

Attack Scenario 2

Attack steps	Attack Incidents	NewAttack number	NewAttack time
1	ICMP_PINGI	13	16:43:11; 16:45:35; 17:00:13 [more]
2	TCP_Scan_Port	11	16:45:21; 16:45:25; 16:54:23 [more]
3	W2K_Val_Passwd	9	16:49:54; 16:55:21; 16:58:29 [more]

Fig. 4. Attack Scenario correlation interface patterns

Experiment 2: In this section, we simulate several types of practical attacks on attack testing network to test the correlation-ship between attack scenario "BSD 4.2 UNIX mail exploit", "Dos on DNS" with real attack. The attack behaviors with $cor(h_i,h_j)$ and $pre(h_i,h_{i-1})$ more than 0.68 are considered as correlated attack behaviors in the same attack scenario. The valid of our methods can be shown by the results of each attack tests presented in Table 4,5 The $cor(h_i,h_j)$ and $pre(h_i,h_{i-1})$ of each new attack actions and attact scenarios are all up to 0.68. This experiment shows the accuracy and availability of our approach.

Table 4. Attack scenario "BSD 4.2 UNIX mail exploit" and new attacks

BSD 4.2 UNIX mail exploit	Attack actions	$Cor\ (h_i, h_j)$	*pre(hi,hi-1)*
Consist_root_mail	Consist_root_mail	0.84	
Set_root_setuid	Set_root_setuid	0.78	0.76
Touch_file	Touch_file	0.74	0.76
Mail_root	Mail_root	0.78	0.79
Run_root_shell	Run_root_shell	0.68	0.71

Table 5. Attack scenario "DOS on DNS" and new attacks

Dos on DNS	Attack actions	$Cor\ (h_i, h_j)$	*pre(hi,hi-1)*
Lookup_target_DNS	Lookup_target_DNS	0.91	
Ping_DNS	Ping_DNS	0.93	0.92
Nmap_DNS	Nmap_DNS	0.78	0.91
Winnuke_target_run	Winnuke_target_run	0.82	0.78

7 Conclusion and Future Work

In this paper, we proposed the approach we used to design the attack scenario construction module of correlation function within SATA (Security Alert and Threat Analysis) project. Attack scenario construction is conducted based on the Algorithm SAMP and CAST to mine frequent attack sequential patterns from attack sequences database transformed from alerts database and construct attack scenario.

The main points of our novel approach can be mentioned as followed: our approach transforms the alerts database into attack sequences to solve problems of multi-agent and multi-stage attack scenario construction. Our approach gets rid of the predefined knowledge limitation of attack conditions and consequences, which most of approaches must rely on. Our approach can discover new attack relationships as long as the alerts of the attacks have calculable correlation. The results of the experiments conducted on the branch of CERNET can demonstrated the potential of our method in multi-agent attack scenario construction and correlation analysis.

There are several interesting and important future directions. We will continue to study the no-limitation of prerequisites approaches for alert correlation, which is an interesting area to study. To further improve the affectivity and accuracy of the correlation and scenario construction is another important direction of our future work.

References

1. Valdes, A., Skinner, K.: Probabilistic alert correlation. In: Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID) (October 2001)
2. Lee, W., Qin, X.: Statistical Causality Analysis of INFOSEC Alert Data. In: RAID (2003)
3. Debar, H., Wespi, A.: Aggregation and correlation of intrusion-detection alerts. In: Recent Advances in Intrusion Detection (2001)
4. Templeton, S.J., Levitt, K.: A requires/provides model for computer attacks. In: Proceedings New Security Paradigm Workshop, Ballycotton, Ireland, vol. 31, ACM, New York (2001)
5. Ning, P., Cui, Y., Reeves, D.S.: Constructing attack scenarios through correlation of intrusion alerts. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington, DC (November 18-22, 2002)
6. Cuppens, F.: Managing alerts in multi-intrusion detection environment. In: Proceedings 17th annual computer security applications conference, New Orleans, pp. 22–31 (2001)
7. Cuppens, F., Miege, A.: Alert correlation in a cooperative intrusion detection framework. In: Proceedings of the 2002 IEEE symposium on security and privacy, p. 202e15 (2002)
8. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H.: PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. In: Proceedings of IEEE Conference on Data Engineering, pp. 215–224 (2001)