A Data Mining Approach to Generating Network Attack Graph for Intrusion Prediction

Zhi-tang Li, Jie Lei, Li Wang, and Dong Li
Computer Science Department
Huazhong University of Science and Technology
430074 Wuhan - Hubei - China
{leeying, leijie, wtwl, lidong}@ hust.edu.cn

Abstract

A network attack graph provides a global view of all possible sequences of exploits which an intruder may use to penetrate a system. Attack graphs can be generated by model checking techniques or intrusion alert correlation. In this paper we proposed a data mining approach to generating attack graphs. Through association rule mining, the algorithm generates multi-step attack patterns from historical intrusion alerts which comprise the attack graphs. The algorithm also calculates the predictability of each attack scenario in the attack graph which represents the probability for the corresponding attack scenario to be the precursor of future attacks. Then the real-time intrusion alerts can be correlated to attack scenarios and ranked by the predictability scores. The ranking result can help identify the appropriate evidence for intrusion prediction from a large volume of raw intrusion alerts. The approach is validated by DARPA 2000 and DARPA 1999 intrusion detection evaluation datasets.

1 Introduction

A network attack graph is a tool used for security vulnerability analysis, intrusion detection and network forensic analysis. It provides a global view of the security vulnerabilities in a system and all possible sequences of exploits which an intruder may use to penetrate the system. Formally, as Sheyner et al. introduced in [7], a network attack graph is a state transition diagram in which each state denotes a state of the attacker, the defender and the system, and the transitions correspond to actions taken by an attacker which lead to a change in the overall state of the system.

Network attack graphs were traditionally created manually, e.g., by Red Teams. Recently some automated approaches to generating attack models are proposed. An attack graph can be either generated by off-the-shelf model checking techniques [7] or by on-line intrusion alert correlation [6, 2].

The approach of using network attack graph for intrusion prediction was advocated by [13, 3]. Generally, a network attack graph can be used for intrusion prediction in the way as follows. First, an attack graph is generated off-line by security experts or automated tools; second, the IDS alerts are real-time collected and correlated to attack scenarios according to the attack graph; then, since we have detected the first few steps of a multi-step attack, according to the attack graph we can predict the next step of the attack: some detected attack scenarios are used as "evidence" and certain inference method is employed to predict the next step of attacks based on the "evidence" and the attack graph.

One problem of this approach is that the intrusion alerts can be overwhelming while most of which may be false positives [4]. Therefore, when facing enormous number of alerts it is difficult to determine which alerts are appropriate to be the "evidence" for intrusion prediction.

In this paper we introduced a data mining approach to generating attack graphs. We used the algorithm of association rule mining to mine attack scenarios from historical IDS alert database and used them to construct attack graphs. During the process of mining attack sequence patterns the algorithm also calculates the *predictability* of each attack scenario in the attack graph which denotes the probability for the corresponding security state to have following attacks. The predictability scores can be used to rank the real-time detected attack scenarios and the ranking result can help with identifying the best evidence for intrusion prediction. Therefore our approach can be used to im-

Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)
0-7695-2874-0/07 $25.00 © 2007 IEEE

prove the efficiency and accuracy of intrusion prediction. The approach are tested by the DARPA 2000 intrusion specific dataset and DARPA 1999 intrusion detection evaluation dataset.

The rest of the paper is organized as follows. Section 2 introduces the related work. Section 3 presents the data mining approach to generating attack graph. Section 4 describes the experiments with DARPA dataset in which we show how the attack graphs are generated and used to support intrusion prediction, and Section 5 concludes the paper.

2 Related Work

Network attack graph allows a security analyst to understand how vulnerabilities in individual network services contribute to overall vulnerability and how an intruder can penetrate the system step by step. Various approaches to generating attack graphs have been proposed. The representative of the early approaches is the model checking approach proposed by Sheyner et al. [7]. In their attack graphs the nodes represent the state of the network in the form of a collection of variables, and the edges represent an attacker's actions that change the state. Given a set of security properties, model checking technique is used to check whether the attack model of a system satisfies the given properties. Philips and Swiler [9] developed a customized search engine to generate the attack graph. Generally this kind of approaches suffers from the scalability problem because of the state explosion problem.

Ammann, et al. [8] proposed a graph search-based algorithm to construct attack graphs which was then used in the Topological Vulnerability Analysis tool. They assumed that an attacker's privileges always increase during the analysis. Since there are only a polynomial number of privileges an attacker can gain, the algorithm will terminate in polynomial time. Xinming Ou, et al.[12] proposed an approach to generating logical attack graphs which directly illustrate logical dependencies among attack goals and configuration information. Since a logical attack graph always has size polynomial to the network being analyzed, their approach has good scalability.

More recent approaches to generating attack graphs construct the exploit-based attack graph by attack scenario constructing techniques. The work of Ning et al. [6] and Cuppens and Miege [2] examined the prerequisites and consequences of atom attacks and constructing attack scenarios by matching the consequences of the prior attacks with the prerequisites of the following attacks. Qin and Lee [5] proposed a statistical-based alert correlation approach to identifying new alert relationship and constructing attack scenarios without depending on prior knowledge of attack transition patterns.

Our approach to generating attack graphs is based on data mining approach. The concept of using data mining in intrusion detection is proposed by [11]. A recent work by Thurimella et al. [10] had shown that the causal relationships between the attackers and the combination of alarms which are generated in intrusion detection logs as a result of their attack behaviors can also be mined through data mining approach. Therefore, it is possible to mine association rules from intrusion alerts and use these rules(frequent attack sequences) to construct attack graphs, which is the purpose of the work of [4, 10] and our work. Our approach is different with other approaches in two aspects. First, unlike the model checking approaches [7] which need to define a lot of security properties and security states, no prior knowledge needs to be maintained in our approach; Second, our approach can estimate the predictability scores of every security states in the attack graph thus it can help with intrusion prediction in finding the best evidence for future attacks.

3 The Algorithm to Generating Attack Graphs

We use association rule mining to generate attack graphs and estimate the predictability of each attack scenarios. The main goal of association rule mining is to locate non-obvious interrelationships between members of a large data set. The goal of our work is to find associations between the various attack classes from historical IDS alerts generated by real attacks in the network, and use them to constitute an attack graph. Since the IDS alerts follow the time series, we use sequential association rule mining algorithm. An example sequential association rule is in the following form, with two statistics which describe their strength and quality:

$$[x \rightarrow y] \rightarrow z$$
Support=50, Confidence=80

This rule states that whenever x $\rightarrow$ y were present in a given sequence, then z was present as well. The Support value states that this specific sequence of three items represents 50 percent of the transactions which were examined. The Confidence value states that 80 percent of the time that the sequence x $\rightarrow$ y were found, the item z was also found.

The algorithm deals with a large set of IDS alerts. Some preprocessing functions need to be performed before the algorithm runs. First we should improve the

Start time	Signature ID
06-03-02-10:55:12	2
06-03-02-10:56:03	3
06-03-02-11:12:29	9
06-03-02-11:25:43	8
06-03-02-11:29:51	17
06-03-02-11:45:08	14
06-03-02-11:49:08	3
06-03-02-12:11:07	2
06-03-02-12:20:12	9
06-03-02-12:26:31	8
06-03-02-12:39:17	14
06-03-02-12:40:03	5
06-03-02-13:10:17	2

(a) Sample Alert Database

Sequence ID	Candidate attack sequence
1	2,3,9,8,17,14,3
2	3,9,8,17,14,3
3	9,8,17,14,3,2
4	8,17,14,3,2,9
5	17,14,3,2,9,8
6	14,3,2,9,8,14,5
7	3,2,9,8,14,5
8	2,9,8,14,5,2

(b) Candidate Attack Sequences

Figure 1. An Example of Data Preparation

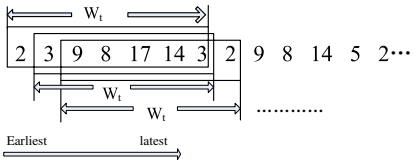


Figure 2. Generating Candidate Attack Sequences with W_t

quality of the data set by alert aggregation and alert verification which reduce the amount of raw alerts and eliminate some false positives. Then the attributes of alerts are mapped to the formats which are more convenient for computation. And last the global alert sequences are divided into separate candidate attack sequences by the sliding Attack Sequence Time Windows (see Figure 2). Suppose the multi-step attacks often happen in a certain range of time interval, the Attack Sequence Time Window is used to define this time interval by which the attack sequences are divided into separate attack sequences(the candidate sequences used for association rule mining).

Here we give an example of how the candidate attack sequences are generated. Figure 1(a) gives a sample sequence of attack signatures in which the signatures have been mapped into integers. Then we divide it into separate attack sequences by Time Window W_t. When the W_t is set to be 1 hour, we got the candidate attack sequences as 1(b).

Figure 3 shows the pseudocode of the algorithm to generating attack graphs and calculating the predictability scores. This algorithm is based on the classic association rule mining algorithm [1]. Different to the classic rule mining algorithm, in our algorithm

Input: A set of candidate attack sequences $\{s_1, s_2, \cdots, s_n\}$ and *min_support*

Output: The Maximal Sequence $\bigcup_k L_k$, and $\prod(\bigcup_k L_k)$

Algorithm:

1: find all the large 1-sequences: L_1={large 1-sequences};

2: once getting the large (k-1)-sequences L_{k-1}, get the large k-sequences L_k;

3: given $l_i \in L_{k-1}$, find the $P_k=\{p_1,p_2,...,p_m\} \subset L_k$, where $\forall p_j \in P_k$, l_i is the beginning subsequence of p_j;

4: find subsequence set S', where $\{\forall s' \in S', \exists p \in P, p$ is the subsequence of $s'\}$ and $\{\forall s_i', s_j' \in S', s_i'$ is not the subsequence of $s_j'\}$;

5: $\pi(l_i) = count(S') / count(l_i)$;

6: $\prod(L_{k-1}) = \{\pi(l_i), l_i \in L_{k-1}\}$;

7: repeat 2,3,4,5,6 until the longest Maximal Sequence is found out;

8: for each L_k, delete the redundant subsequences of L_k from $L_1, L_2, ..., L_{k-1}$;

9: finally, output the Maximal Sequence $\bigcup_k L_k$, and $\prod(\bigcup_k L_k)$.

Figure 3. The Algorithm to Generating the attack graph

when counting the *support* of one subsequence, only the situation that the subsequence appears at the beginning of a candidate sequence counts.

The input of the algorithm is a set of candidate attack sequences and the *min_support* value. And the output of the algorithm is the Maximal Attack Sequences $\bigcup_k L_k$ and the probabilities of having following attacks(Predictability) of each maximal attack sequences $\prod(\bigcup_k L_k)$.

In next section we will show the result of the algorithm and how the algorithm works by the experiment with DARPA intrusion evaluation datasets.

4 Experiment

We tested the algorithm of generating attack graphs by the DARPA 2000 intrusion detection scenario specific dataset and the DARPA 1999 intrusion detection evaluation dataset. The DARPA 2000 dataset has two multi-stage attack scenarios namely LLDOS 1.0 and LLDOS 2.0.2. The DARPA 1999 dataset contains 20 days(workdays of 4 weeks) of different attack traffics. And these two datasets share the same network topology.

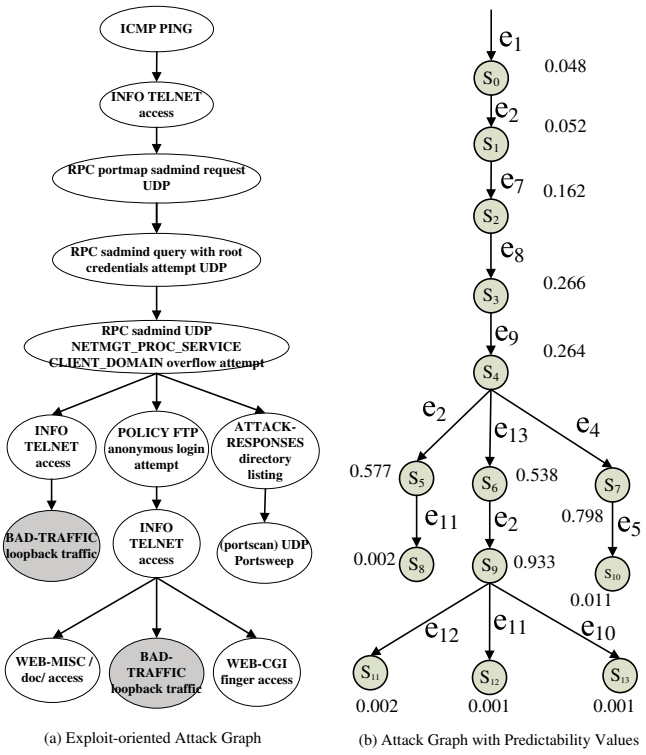
We replayed each of the LLDOS 1.0 and LLDOS 2.0.2 datasets for ten times separately, and each time we replayed one day's attack traffic of DARPA 1999 dataset simultaneously as the background attacks traffics. The network was monitored by Snort V2.4.5 with the rule set V2.4. And the functions of replaying were

Rank	*Attack Scenario*	Time Range	*Source→Target*	*Predictability*
1	$e_{13} \rightarrow e_2$	10:53:35-10:59:26	197.192.91.223→172.16.112.100 172.16.113.168→197.182.91.233	0.933
2	$e_2 \rightarrow e_4$	10:59:27	172.16.112.100→196.37.75.158	0.798
3	e_4	12:14:58	195.73.151.50→172.16.112.100	0.768
4	$e_2 \rightarrow e_7 \rightarrow e_8 \rightarrow e_9$	11:04:28-11:32:38	202.77.162.213→172.16.115.20	0.264
5	e_2	10:50:08-11:06:29	172.16.112.194→194.7.248.153	0.016

Table 1. The Ranked Attack Scenarios

fastened and each time the procedure lasted for 4 minutes by average. Then the Snort alerts of twenty times were merged into one alert dataset and preprocessed by the alert aggregation module. The goal of our algorithm is to generate attack graph from this alert dataset which should contain the attack scenarios of both the LLDOS 1.0 attack and the LLDOS 2.0.2 attack and calculate the predictability scores of every attack scenarios.

We ran the algorithm with the Attack Sequence Time Window setting to 4 minutes. And when the *Support* value was set to 5 (percent), the algorithm generated attack graph as Figure 4. The attack graph contains 18 types of Snort alerts. Figure 4(a) is the exploit-oriented attack graph in which each node represents an exploit named by the Snort signatures. Figure 4(b) is the generated attack graph in which each node denotes a security state and each transition is an exploit which lead the security state from one change to another.



(a) Exploit-oriented Attack Graph (b) Attack Graph with Predictability Values

Figure 4. The Generated Attack Graph

We can see from Figure 4 that the generated attack graph contains the attack scenarios of both LLDOS 1.0 and LLDOS 2.0.2 and some other attacks. This shows that our algorithm can generate the attack graph as long as the dataset contains the frequent attack scenarios. The LLDOS 1.0 attack scenario can be represented by the attack sequence "$e_1 \rightarrow e_2 \rightarrow e_7 \rightarrow e_8 \rightarrow e_9 \rightarrow e_2 \rightarrow e_{11}$", and the LLDOS 2.0.2 attack scenario is as follows: "$e_1 \rightarrow e_2 \rightarrow e_7 \rightarrow e_8 \rightarrow e_9 \rightarrow e_{13} \rightarrow e_2 \rightarrow e_{11}$". Actually in this alert database the exploit e_{11} (BAD-TRAFFIC loopback traffic) is the indicator of *MStream* DOS attack, because in the Snort ruleset v2.4.5 there is no rule specified to detect *MStream* DOS attack, while whenever it is executed the attacker use fake IP addresses which would trigger the Snort alerts with the signature of "BAD-TRAFFIC loopback traffic".

In the result attack graph the predictability scores of every security states are given. For example, if the attacker reaches the s_7 security state (the following exploits have been executed: $e_1 \rightarrow e_2 \rightarrow e_7 \rightarrow e_8 \rightarrow e_9 \rightarrow e_4$), where $\pi(s_7) = 0.798$, this means the probability of following attacks to this state is 0.798.

After the attack graph was generated, we can use it for intrusion prediction. We replayed the LLDDOS 1.0 and LLDOS 2.0.2 attack traffic simultaneously and used Snort to detect the attacks. The snort alerts were real-time collected and correlated into attack scenarios. Then the alerts are ranked by their predictability scores every five minutes. The time range of the experiment lasted for nearly 4 hours from 10:44:53 to 14:38:56, and 1573 alerts are received totally. Table 1 shows the ranking result at 12:20:00 before the DoS attack against 131.84.1.31 was launched and at this time 293 alerts are received.

1. Entry1: This attack scenario represents the exploit of using Ftp to upload mstream DOS soft-

ware to 172.16.112.100 which is the third step of LLDOS 2.0.2 attack. The predictability of this scenarios is 0.933 which means it is very possible that further attacks related to this scenario would take place. This is the evidence of the coming Mstream DoS attack of LLDOS 2.0.2 attack scenario.

2. Entry2: This represents that 172.16.112.100 is probable to be compromising to 196.37.75.158.

3. Entry3: This represents that 195.73.151.50 is probable to be compromising to 172.16.112.100.

4. Entry4: This represents the fouth step of LLDOS 1.0 has been reached, and the attacker has gained the administrator's privilege of 172.16.115.20 and is ready to launch Mstream DoS attack.

5. Entry5: This represents that 172.16.112.194 is trying to telnet to 194.7.248.153.

We can see from the result that the attack scenarios ranked to be the top five of the ranking list are all very probable to be the precursors of future attacks. This shows that our approach can find out the evidence for future attacks.

5 Conclusion

In this paper we proposed a data mining approach to generating network attack graph. The algorithm calculates the predictability scores of every attack scenarios in the attack graph as well while generating the attack graph, which represent the probabilities of the corresponding security states to have following attacks.Then the realtime intrusion alerts can be correlated to attack scenarios by the attack graph and ranked by their predictability scores. The ranking result lets the attack scenarios which are most probable to have following attacks appear at the top of the list. Thus the approach can help with intrusion prediction in identifying the best "evidence" of future attacks.

The experiments with the DARPA 2000 intrusion detection scenario specific dataset and the DARPA 1999 intrusion detection evaluation dataset had shown that the approach could generate attack graphs from historical IDS alert database and the approach is useful in intrusion prediction.

The limitation of our approach lies in that the attack graphs generated by the data mining approach are usually partial and it depends on the completeness of data source to generate the overall attack graphs of the network system.

References

[1] S. A. Agrawal R., Imielinski T. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 207–216, 1993.

[2] F. Cuppens and A. Miege. Alert correlation in a cooperative intrusion detection framework. Proceedings 2002 IEEE Symposium on Security and Privacy, page 202, Berkeley, CA, USA, 2002. IEEE Comput. Soc.

[3] C. W. G. Goldman and R. P. Plan recognition in intrusion detection systems. In *Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX II' 01)*, Anaheim,California, 2001.

[4] K. Julisch and M. Dacier. Mining intrusion detection alarms for actionable knowledge. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 366–375, 2002.

[5] W. Lee and X. Qin. Statistical causality analysis of infosec alert data. In *Proceedings of the International Symposium on the Recent Advances in Intrusion Detection(RAID 2003)*, pages 73–94. Springer-Verlag, 2003.

[6] P. Ning, Y. Cui, D. S. Reeves, and D. Xu. Techniques and tools for analyzing intrusion alerts. *ACM Transactions on Information and System Security*, 7(2):274, 2004.

[7] S. J. R. L. Oleg Sheyner, Joshua Haines and J. M. Wing. Automated generation and analysis of attack graphs. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy (SP 2002)*, pages 273–284, 2002.

[8] D. W. P. Ammann and S. Kaushik. Scalable, graph-based network vulnerability analysis. In *Proceedings of 9th ACM Conference on Computer and Communications Security*, Washington, DC, November 2002.

[9] C. Phillips and L. P. Swiler. A graph-based system for network-vulnerability analysis. In *Proceedings of the 1998 workshop on New security paradigms*. ACM Press.

[10] J. J. T. Thurimella and Ramakrishna. A framework for the application of association rule mining in large intrusion detection infrastructure. In *Proceedings of the International Symposium on the Recent Advances in Intrusion Detection(RAID 2006)*, pages 1–18, 2006.

[11] S. S. Wenke Lee. Data mining approaches for intrusion detection. In *Proceedings of the 7th USENIX Security Symposium*, pages 79–94, 1998.

[12] W. F. B. Xinming Ou and M. A. McQueen. A scalable approach to attack graph generation. In *ACM Conference on Computer and Communications Security 2006*, Alexandria, Virginia, USA.

[13] W. L. Xinzhou Qin. Attack plan recognition and prediction using causal networks. In *Proceedings of The 20th Annual Computer Security Applications Conference (ACSAC 2004)*, pages 370–379, Tucson,Arizona, December 2004.