J. Sci. Trans. Environ. Technov. 2009, 3(2): 74-78

Data mining for network intrusion detection system in real time

M.Vanitha, R. Malathi*, T. Lavanya and M. Muthuraman

Department of Information Technology., J.J. College of Arts & Science, Pudukkottai-622 404, Tamil Nadu, India

Abstract

Intrusion detection technology is an effective approach to dealing with the problems of network security. In this paper, we present a data mining-based network intrusion detection framework in real time (NIDS). This framework is a distributed architecture consisting of sensor, data preprocessor, extractors of features and detectors. To improve efficiency, our approach adopts a novel FP-tree structure and FP-growth mining method to extract features based on FP-tree without candidate generation. FP-growth is just accord with the system of real-time and updating data frequently as NIDS. We employ DARPA intrusion detection evaluation data set to train and test the feasibility of our proposed method. Experimental results show that the performance is efficient and satisfactory. Finally, the development trend of intrusion detection technology and its currently existing problems are briefly given.

Keywords : data mining, FP-growth, intrusion detection, network, real time

INTRODUCTION

With the evolution of the technology of information, especially the prevalence of the technology of Internet/ Intranet, security of more and more organization and individual's computer system establishment and information resource have been threatened. Therefore, the security of information has become one of the most important tasks in the domain of information technology. Traditional model of intrusion detection has been established to be inefficient and the cost of research is also high. The technology of data mining takes on particular predominance in the domain of unexpected knowledge acquiring. As a result Data Mining-based Intrusion Detection has become prevalent. In essence, Network security is just network information security. In general, all technologies and theories about secrecy, integrality, usability, reality and controllable of network information are the research domain of network security.

Intrusion is an action that tries to destroy that secrecy, integrality and usability of network information, which is unlicensed and exceed authority. Intrusion Detection is a positive technology of security defend, which gets and analyses audit data of computer system and network from some network point, and to discover whether there is the action of disobeying security strategy and whether be assaulted. Intrusion Detection System is the combination of software and hardware.

FEATURE EXTRACTION FOR NIDS

Feature extraction adopts a FP-tree structure and FP-growth mining method based on FP-tree without candidate generation, which optimized from *Apriori* algorithm. FP-growth is just adapt to the system of real time and updating data frequently like NIDS. *Apriori* is a basal algorithm of generating frequent patterns. *Apriori* employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k+1)-itemsets. *Apriori* is an influential algorithm for mining frequent itemsets for Boolean association rules. Many association-mining algorithms evolve from it. In some application cases the *Apriori* behave not as good as expected (i.e., need to repeatedly scan the itemsets, inefficient, consuming abundant resource of CPU).

FP-growth is optimized algorithm from *Apriori*. FP-growth adopts a divide-and-conquer strategy that compresses the database representing frequent items into a frequent-pattern tree (FP-tree), and proceeds mining of the FP-tree. FP-tree is a good compact tree structure, which contains the complete information of the database in relevance to frequent pattern mining, and its size is usually highly compact and much smaller than its original database. The method is highly compressed so that frequent item-sets generation is integrated and don't need to repeatedly scan the itemsets. Therefore NIDS adopts FP-growth, and the conclusion is whether resource using or efficiency is advanced.

The main steps of FP-growth method are as follows:

(i) Construct conditional pattern base for each node in the FP-tree.

(ii) Construct conditional FP-tree from each conditional pattern-base.

(iii) Recursively mine conditional FP-trees and grow frequent patterns obtained so far.

(iv) If the conditional FP-tree contains a single path, simply enumerate all the patterns.

Let's look at an example of extraction of features.

*Corresponding Author
email: *shamalu2008@rediffmail.com*

Example 1

This example is based on preprocessed data of Table 1. Assume the minimum support count is 2. There are four transactions in this database. Fig.1 is the FP-tree constructed from the Table 1. Table 2 is the first scan of the database candidates 1-itemsets and their support counts. Table 3 shows the result.

THE FRAMEWORK AND IMPLEMENT OF NIDS

The overall system distributed architecture framework is designed to support a data mining-based NIDS, as shown in Fig. 2. The architecture adopts a detection mode of real time based on network. The system consists of several divided-modules namely data collection module (sensor), data preprocessor, threads control module, extractors of features, detectors and result return module between user and computer.

Data collection adopts sniffer theory using socket. After gathered, audit data must be preprocessed and cleaned (i.e., adds a sign after data items, as ‘sIP’ ‘dIP’ ‘sPt’ ‘dPt’ ‘po’). Threads control module mainly dominates extractors of features to generate frequent itemsets according to the size of Time Window and the status of data collection. The dominating of extractors depends on how to glide Time Window, as shown in Fig. 3. Datasets are overlapped in two neighboring time window, and the size of those datasets overlapped just is the size of glide Time Window subtracted from the Time Window’s size. How to control the size of Time Window is important. When choosing the size of Time Window, we must take into account the sort

Table 1. Preprocessed audit data

TID	Items
T100	TCP-po,192.168.0.1-sIP, 80-sPt , 192.168.0.2-dIP, 1717-dPt
T200	TCP-po,202.198.16.220-sIP,80-sPt ,10.60.46.58-dIP, 2209-dPt
T300	TCP-po,192.168.0.1-sIP ,3050-sPt,192.168.0.2-dIP,1717-dPt T400
T400	TCP-po,202.198.16.220-sIP,80-sPt ,10.60.46.58-dIP,1717-dPt

Table 2. Frequent items(1-itemsets) and their support counts generated by scan the database

Itemset	Support count
TCP-po	4
80-sPt	3
1717-dPt	3
192.168.0.1-sIP	2
202.198.16.220-sIP	2
192.168.0.2-dIP	2
10.60.46.58-dIP	2
3050-sPt	1
2209-dPt	1

Table 3. Frequent itemsets generated by mining the FP-tree

TID	Items
10.60.46.58-dIP, 202.198.16.220-sIP	2
10.60.46.58-dIP, 80-sPt	2
TCP-po, 1717-dPt	3
.	.
.	.
.	.
10.60.46.58-dIP, 80-sPt, 202.198.16.220-sIP	2
TCP-po, 80-sPt, 1717-dPt	2
TCP-po, 80-sPt, 202.198.16.220-sIP	2
.	.
.	.
.	.
TCP-po, 192.168.0.1-sIP, 192.168.0.2-dIP, 1717-dPt	2
TCP-po, 10.60.46.58-dIP, 202.198.16.220-sIP, 80-sPt	2

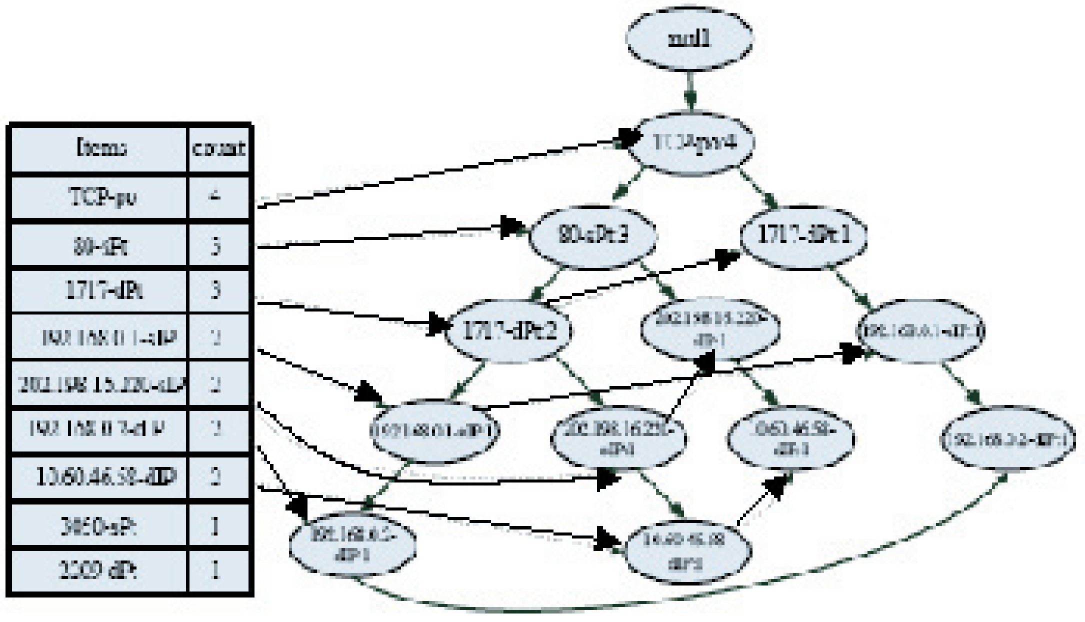


Figure 1. An FP-tree that registers compressed, frequent pattern information

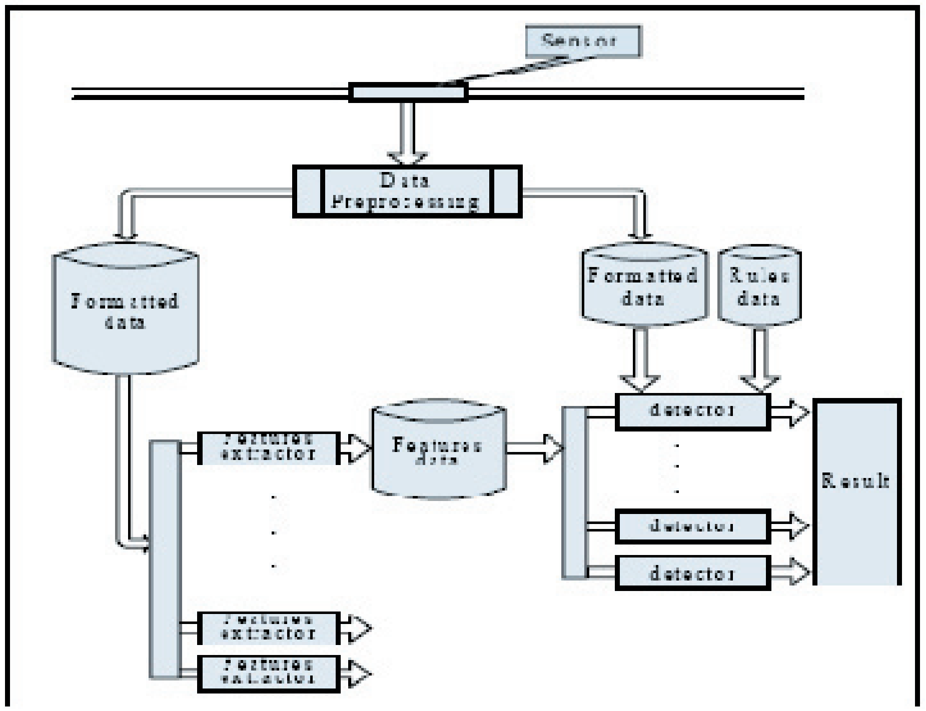


Figure 2. The Architecture of NIDS based on data mining

of detection, the computer and network hardware capability.

Especially, how to choose the size of the overlapped data is extraordinarily important. If the size too small, the system may be miss some attacks. While if the size too large, the system will consume abundant resource of memory and CPU. The core features generation method is just FP-growth. It is composed of two parts: FP-tree constructing, mining the FP-tree. By mining FP-tree, FP-growth method transforms the problem of finding long frequent patterns to looking for shorter ones recursively and then concatenating the suffix. It uses the least frequent items as a suffix, offering good selectivity. The method substantially reduces the search costs.

Detectors are used for comparing and identifying attacks by corresponding attack patterns, with setting master attribute and assistant attribute. It adopts classification by decision tree. A decision tree is a flow-chart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and leaf nodes represent classes or class distributions.

When the system is running, the system chooses the number of collecting data based on the Time

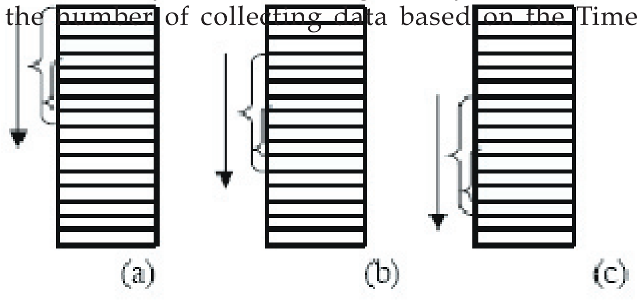


Figure 3. Gliding of Time Window

Window's size and the form of data updating. When the condition accord with the requirement, all the extractors will be work. When and how extractors work lies on data preprocessor. Because the system is a real time detection system, and can not loose data during detection, the data preprocessor is the core time line. The thread control module drives extractors to work in multithreading when the condition accord with the requirement. If an extractor is working and the data preprocessor's condition accord with the requirement, another extractor is constructed immediately. The system sets some buffers in the memory. Thus the system is rather adapted to real-time requirement.

THE EXPERIMENTS AND RESULTS

In the experiment, we partly use *2000 DARPA Intrusion Detection Scenario Specific Data Sets* to train and test our NIDS prototype. It provided a standard corpus for evaluating intrusion detection systems. It also introduced more stealthy attacks, insider attacks and attacks against the windows operating system.

Attacks fall into four main categories:

Ø DOS: denial of service

Ø R2L: unauthorized access from a remote machine

Ø U2R: unauthorized access to local super user (root) privileges

Ø Probing: surveillance and other probing

We built a NIDS and implemented it by Microsoft Visual C++ 6.0. Fig. 4. shows some part of label data patterns. The size of time window is 350, and the minimum support count is 10. The size of the datasets overlapped is 50 percent of the size of time window. Table 4 shows the Performance of our system.

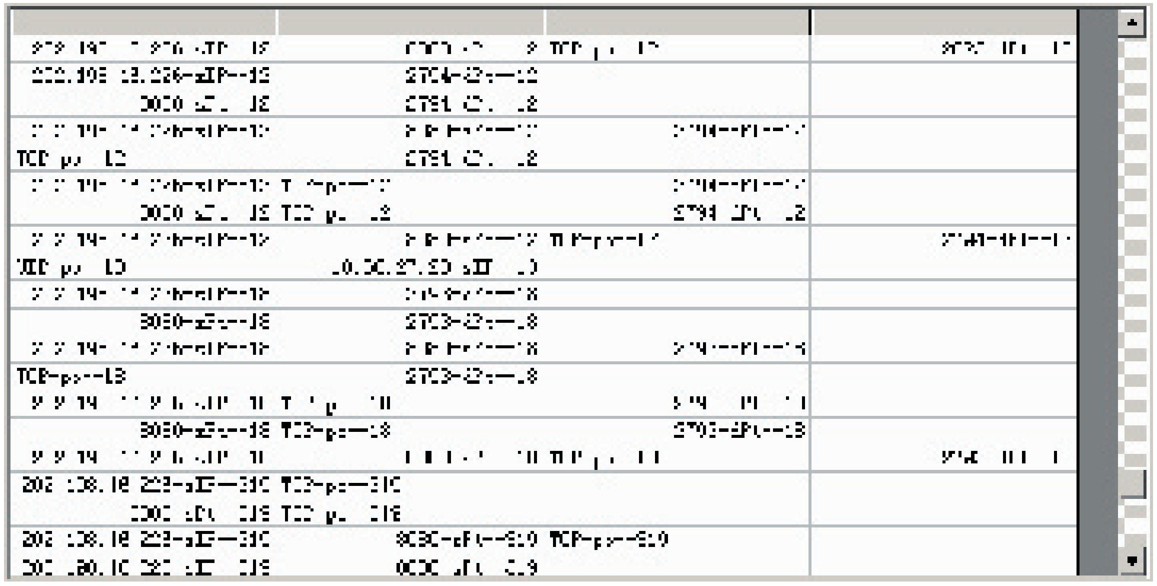
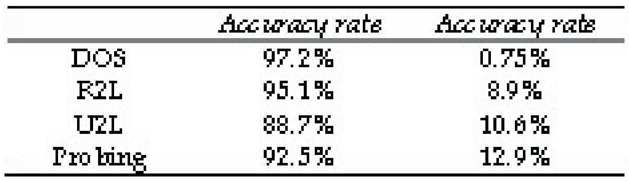

[illegible]

Figure 4. Part of the label data sets

Table 4. The accuracy rate and false alarm rate



	Accuracy rate	Accuracy rate
DOS	97.2%	0.75%
R2L	95.1%	8.9%
U2L	88.7%	10.6%
Probing	92.5%	12.9%

CONCLUSION

In this paper, we outlined and implemented the architecture of the data mining-based network intrusion detection system in real-time (NIDS). We analyze a frequent patterns mining algorithm that integrate *Apriori* candidate generation into FP-growth method. FP-growth adopts a divide-and-conquer strategy that compresses the database representing frequent items into a frequent-pattern tree (FP-tree), and proceeds mining of the FP-tree. The method is highly compressed and frequent itemsets generation is integrated and don't need to repeatedly scan the itemsets. Therefore extractor of features adopts FP-growth, and the conclusion is that both resource consuming and efficiency are satisfied. We also expect more such attempts in the future. We are also developing unsupervised anomaly detection algorithms to reduce the reliance on labeled training data. Experiments on *DARPA* shows the performance of the NIDS is satisfied. Our future work includes researching some new algorithms and refining the existing system.

REFERENCES

Gehrke, J., Ramakrishnan, R. and Ganti, V. 2000. RAINFOREST - a Framework for Fast Decision Tree Construction of Large Datasets. Data Mining and Knowledge Discovery: 4(2/3): 127-162.

Ghosh A. and Schwartzbard, A. 1999. A study in using neural networks for anomaly and misuse detection. *In: Proc. of the Eighth USENIX Security Symposium,* August 23-26, 1999, Washington D.C.

Han, J. Pei, J. and Yin, Y. 2000. Mining Frequent Patterns without Candidate Generation, *In: Proc. 2000 ACM SIGMOD International Conference on Management of Data*. Dallas T.X. May 2000.

Han, J. and Kamber, M. 2006. Data Mining: Concepts and Techniques. The Morgan Kaufmann Series in Data Management Systems, Jim Gray, Series Editor, Morgan Kaufmann Publishers, San Franscisco.

Lee, W. Stolfo, S. J. and Mok, K.W. 1999. Mining in data flow environments: Experiences network in intrusion detection. *In: Proc. of the 1999 Conference on Knowledge Discovery and Data Mining (KDD-99)*. San Diego, California, USA, pp. 114-214.

Lee, W. Stolfo, S. J. Chan, P.K. Eskin, E. Fan, W. Hershkop, S. Miller, M. and Zhang, J. 2001. Real time data mining-based intrusion detection. *In:* DARPA Information Survivability Conference and Exposition (DISCEX II'01), Anaheim, California.

Pornoy, L. 2000. Intrusion detection with unlabeled data using clustering. In Undergraduate Thesis, Columbia University, Department of Computer Science.

Zamboni, D. 2001. Using clustering to detect abnormal behavior in a distributed intrusion detection system. Unreleased Technical Report, Purdue University. August.