Reducing Network Intrusion Detection using Association rule and Classification algorithms

K.KEERTHI[1], P.SREENIVAS[2]

[1] M.Tech(CSE), KAKINADA INSTITUTE OF ENGINEERING AND TECHNOLOGY
[2] Assistant professor, KAKINADA INSTITUTE OF ENGINEERING AND TECHNOLOGY

ABSTRACT:

IDS (Intrusion Detection system) is an active and driving defense technology. This project mainly focuses on intrusion detection based on data mining. Data mining is to identify valid, novel, potentially useful, and ultimately understandable patterns in massive data. This project presents an approach to detect intrusion based on data mining frame work. Intrusion Detection System (IDS) is a popular tool to secure network. Applying data mining has increased the quality of intrusion detection neither as anomaly detection or misused detection from large scale network traffic transaction. Association rules is a popular technique to produce a quality misused detection. However, the weaknesses of association rules is the fact that it often produced with thousands rules which reduce the performance of IDS. This project aims to show applying post-mining to reduce the number of rules and remaining the most quality rules to produce quality signature. This experiment uses KDD Cup 99 dataset to detect IDS rules using Apriori Algorithm, which later performing post-mining using Chi-Squared ($\chi 2$) computation techniques. The quality of rules is measured based on Chi- Square value, which calculated according the support, confidence and lift of each association rule. Decision tree rules are also identified in order to detect attacks in the dataset as well as real time nework traffic dataset. The experimental results demonstrate its effectiveness and efficiency.

I INTRODUCTION

A network intrusion attack often is any use of a network that compromises its stability and even the security of real info that would be stored on computers coupled with it. A wide range of activity is categorized as this definition, including effort to de-stabilize the network overall, gain unauthorized access to files or privileges, simply mishandling and misuse of software. Added security measures can stop these kind of attacks. The intention of intrusion detection is to create system which could automatically scan network activity and detect such intrusion attacks. Once an attack is detected, the machine administrator could well be informed and in consequence take corrective action. Detecting such abusive simply not only provides information on damage assessment, but additionally will help to prevent future attacks. These attacks are normally detected by tools known as intrusion detection system. The most popular and well-known data to have an intrusion detection method is the audit data. An audit trail is the records among the activities on a system kept in chronological order. Since there exist note for one activity (which might even correspond to one system call) inside the system, theoretically it is more than possible manually analyze the source data and detect any abnormal activity inside the system. However, the vastness of the audit data provided by an audit collection system often makes manual analysis impractical. Therefore, an automated audit data analysis tool is considered the only solution.

An intrusion detection system (IDS) is software and/or hardware invented to detect unwanted attempts at accessing, manipulating, and/or disabling of computer system, mainly through a network, typically the internet. One of the main challenges in the security management of large-scale high-speed networks (LSHSN) happens to be the detection of anomalies in network traffic. A secure network must provide the following:

• Data confidentiality: Data that are being transferred in the network should be accessible only to those which have been properly authorized.
• Data integrity: Data should maintain their integrity from the moment they are transmitted towards the moment they are actually received. No corruption or data loss is accepted either from random events or malicious activity.
• Data availability: The network ought to be resilient to Denial of Service attacks.

Anomaly detection: It truly is based on the normal behavior associated with a subject (e.g. an individual or possibly a system). Any action that significantly deviates coming from the normal behavior is held to be as intrusive. Which means when we could generate a normal activity profile to produce a system. Our team can flag all system states varying from established profile. Misuse/Signature detection: misuse detection catches intrusions in regards to the characteristics of known attacks. Any action that conforms to the pattern of a known attack or vulnerability is considered as intrusive. The best issues in misuse detection system are tips to write a signature that encompasses all possible variations of the pertinent attack. And the best way to write signatures that don't also match non-intrusive activity.

II BACKGROUND AND RELATED WORK

Supervised Methods :The main goal as to the supervised methods is to design a predictive model (classifier) to classify or label incoming patterns. The classifier has to be trained with labeled patterns in order to classify new unlabeled patterns. The given labeled training patterns are use to here are the description of classes. Some supervised methods include support vector machines, neural network

and genetic algorithms to name a few. 2.3.2 Unsupervised Methods

Unsupervised methods, also termed as data clustering, use a different approach by grouping unlabeled patterns into clusters based upon similarities. Patterns contained in the same clusters are more a dead ringer for one other than they're to patterns owned by different clusters. Data clustering is extremely useful when little priori details about information is offered. Clustering methods can be classified into two categories: hierarchical clustering algorithms (Figure 1,a) and partitioned clustering algorithms(Figure 1,b).

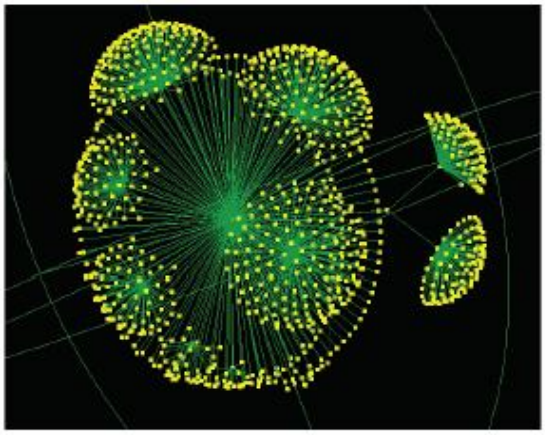
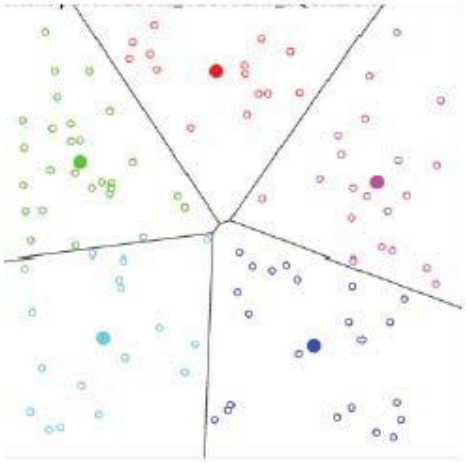


Figure 1 (a) Hierarchical clustering output



(b) Partitional clustering output

James Anderson[2] first proposed that audit trails should be utilized monitor threats. Most of the available system security procedures were geared toward denying admission to sensitive data because of an unauthorized source. Dorothy Denning [4] first proposed the concept of intrusion detection as a chemical solution the topic of providing a sense of security in computer systems. The basic idea is that intrusion behavior involves abnormal usage of sst. Dinner gown model is most definately a rule-based pattern matching system. Some models of normal usage of the internal system could well be constructed and verified against usage of the machine and the significant deviation coming from the normal usage flagged as abnormal usage. This model served as an abstract model for further developments in this particular field and it is generally known as generic intrusion detection model.

IDES [5] used expert system strategies for misuse intrusion detection and statistical techniques for anomaly detection. IDES expert system component evaluates audit records as they are produced. The audit records are viewed as facts, which map to rules in the rule-base.

Problem Definition:

- Existing apriori algorithm needs more database scans for generating important patterns.
- Apriori algorithm uses support and confidence values in order to give more interestingness rules.
- This approach gives more standard error for generating association rules.
- Apriori algorithm takes more time and memory to generate rules during the database scan process.
- Existing system suffering with False positive and False Negative measures.
- This system fails to detect normal behaviour of a system.
- Most intrusion systems based on Rule based approach.
- Rule-Based analysis relies on sets of predefined rules that are provided by an administrator, automatically created by the system.
- Existing algorithms does not handle huge dataset.
- Existing techniques does not implement outlier before the applying association rule mining algorithm.

Ko et al. at UC Davis first proposed to specify the intended behavior of some privileged programs (setuid root programs and daemons in UNIX) using a program policy specification language [42]. During the program execution, any violation of the specified behavior was considered “misuse”. The major limitation of this method is the difficulty of determining the intended behavior and writing security specifications for all monitored programs. Nevertheless, this research opened the door of modeling program behavior for intrusion detection.

Leonid Portnoy [53] presented method for detecting Intrusion based on feature vector collected from network, without being given any information about classification of these vectors. He designed a system that implemented clustering technique and able to detect a large number of intrusions while keeping false positive rate reasonable low. Data clustering technique has advantage over the signature based classifier. First that no manual classification of training data is needs to done. The second is that we do not aware of new types of intrusions in order for the system to be able to detect them.

3. PROPOSED FRAMEWORK

This section, it gives an overview of the data set used for intrusion detection. This data set contains seven weeks of training data and two weeks of testing data. The raw data was about four gigabytes of compressed binary TCP dump data from the of network traffic generated. This was processed into about five million connection records, each of

which is a vector of extracted feature values of that network connection. As we know, a connection is a sequence of TCP packets to and from some IP addresses, starting and ending at some well defined times. This data set of the five million connection records was used as the data set for the 1999 KDD intrusion detection contest and is called the KDD Cup 99 data. In particular, MIT Lincoln Lab's DARPA intrusion detection evaluation datasets have been employed to design and test intrusion detection systems. In 1999, recorded network traffic from the DARPA 98 Lincoln Lab dataset [4] was summarized into network connections with 41-features per connection. This formed the KDD 99 intrusion detection benchmark in the International Knowledge Discovery and Data Mining Tools Competition.

The KDD 99 intrusion detection datasets are based on the 1998 DARPA initiative, which provides designers of intrusion detection systems (IDS) with a benchmark on which to evaluate different methodologies [3]. To do so, a simulation is made of a factitious military network consisting of three 'target' machines running various operating systems and services. Additional three machines are then used to spoof different IP addresses to generate traffic. Finally, there is a sniffer that records all network traffic using the TCP dump format. The total simulated period is seven weeks.

Each connection was labeled as normal or as exactly one specific kind of attack. All labels are assumed to be correct. There were a total of 37 attack types in the data set. The simulated attacks fell in exactly one of the four categories : User to Root; Remote to Local; Denial of Service; and Probe.

• **Denial of Service (dos):** Attacker tries to prevent legitimate users from using a service.
• **Remote to Local (r2l):** Attacker does not have an account on the victim machine, hence tries to gain access.
• **User to Root (u2r):** Attacker has local access to the victim machine and tries to gain super user privileges.
• **Probe:** Attacker tries to gain information about the target host.

Apriori Algorithm:
Pass 1

- Generate the candidate itemsets in C_1
- Save the frequent itemsets in L_1

Pass *k*

- Generate the candidate itemsets in C_k from the frequent itemsets in L_{k-1}
  - Join L_{k-1} p with L_{k-1}q, as follows: **insert into** C_k **select** $p.\text{item}_1$, $p.\text{item}_2$, . . . , $p.\text{item}_{k-1}$, $q.\text{item}_{k-1}$ **from** L_{k-1} p, L_{k-1}q **where** $p.\text{item}_1 = q.\text{item}_1$, . . . $p.\text{item}_{k-2} = q.\text{item}_{k-2}$, $p.\text{item}_{k-1} < q.\text{item}_{k-1}$
  - Generate all (k-1)-subsets from the candidate itemsets in C_k
  - Prune all candidate itemsets from C_k where some (k-1)-subset of the candidate itemset is not in the frequent itemset L_{k-1}
- Scan the transaction database to determine the support for each candidate itemset in C_k
- Save the frequent itemsets in L_k

The Apriori Algorithm requirements are Confidence and Support; these two values determine the degree of association that must hold. The Support shows how many times the items in the rule crop up together and it is the relation of transactions that include all the items in the antecedent and consequent to the number of whole transactions and the confidence shows the probability of both the antecedent and the consequent coming into view in the same transaction. Confidence is the relation of the rule support to the number of transactions that include the antecedent and it is the conditional probability of the consequent given the antecedent. Usually when both of the measures are high we need a third determent, in this case it is Lift to evaluate the quality of rules. Lift shows the power of a rule over the random co-occurrence of the antecedent and the consequent, given their individual support. It offers information about the development, the increase in probability of the consequent given the antecedent. Lift is defined as follows:

$$Lift = \frac{(Rule\ Support)}{(Support\ (Antecedent) * Support\ (Consequent))}$$

Chi-Square Pruning technique

Chi-Squared ($\chi 2$) is an analysis technique that helpful in determining the association rules statistical significance level,. Computing the chi-square statistic for the couple of variables (A, B) involves constructing two contingency tables. The experimental contingency table for (A, B) has four cells, parallel to the four possible Boolean combinations of A, B. The value in every cell is the number of explanation (samples) that competition the Boolean combination for that cell.

IMPROVED C45 ALGORITHM:

IMPROVED C45:

Attribute Selection:
Apply Attribute selection to each attribute(L, attribute list) to find the "best" splitting criterion; Gain measures how well a given attribute separates training examples into targeted classes. The one with the highest information is selected. Given a collection S of c outcomes The expected information needed to classify a tuple in D is given by
In the kdd99 dataset we have two class labels ie normal and anomaly.Hence

Modified Information or entropy is given as

$\text{ModInfo(D)} = -S_i \sum_{i=1}^{m} l\,\text{og}\sqrt{S_i}$,m different classes

$$\text{ModInfo(D)} = -S_i \sum_{i=1}^{2} l \log \sqrt{S_i}$$

$$= -S_1 \log \sqrt{S_1} - S_2 \log \sqrt{S_2}$$

Where S_1 indicates set of samples which belongs to target class 'anamoly', S_2 indicates set of samples which belongs to target class 'normal'.

Information or Entropy to each attribute is calculated using

$$Info_A(D) = \sum_{i=1}^{v} |D_i| / |D| \times ModInfo(D_i)$$

The term Di /D acts as the weight of the jth partition. ModInfo(D) is the expected information required to classify a tuple from D based on the partitioning by A.

Information gain is defined as the difference between the original information requirement) and the new requirement .That is,

$$Gain(A) = Mod \inf o(D) - \inf o_A(D)$$

Finding Best Split:

In order to decide which attribute is best split measure ,correlation coefficient is used as threshold as

$$r = \sum XY - \overline{XY} / \sqrt{SD_x} . \sqrt{SD_y}$$

Let A= MaxGain{AttributeList
If(r>0 and A>r })
{
A is positively alerted and the node is selected.
}
Elseif(r<0 and A>r)
{
A is negatively alerted and the node is discarded.
}
Elseif(r=0 and A>r)
{
A is unalerted and next highest MaxGain is selected.
}
Else
A is discarded

Depending on the alert type severity, the decision on the root node and the child nodes are selected.

Recurs on the sub lists obtained by splitting on a best, and add those nodes as children of node.

Experimental Results:

All experiments were performed with the configurations Intel(R) Core(TM)2 CPU 2.13GHz, 2 GB RAM, and the operating system platform is Microsoft Windows XP Professional (SP2).

Improved Apriori Results:

Best rules found:

1. protocol_type=tcp dst_bytes=0 logged_in=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
2. protocol_type=tcp dst_bytes=0 num_failed_logins=0 logged_in=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
3. protocol_type=tcp dst_bytes=0 logged_in=0 num_file_creations=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
4. protocol_type=tcp dst_bytes=0 logged_in=0 num_access_files=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
5. protocol_type=tcp dst_bytes=0 logged_in=0 num_outbound_cmds=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
6. protocol_type=tcp dst_bytes=0 logged_in=0 is_host_login=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
7. protocol_type=tcp dst_bytes=0 logged_in=0 is_guest_login=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
8. protocol_type=tcp dst_bytes=0 num_failed_logins=0 logged_in=0 num_file_creations=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
9. protocol_type=tcp dst_bytes=0 num_failed_logins=0 logged_in=0 num_access_files=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)
10. protocol_type=tcp dst_bytes=0 num_failed_logins=0 logged_in=0 num_outbound_cmds=0 dst_host_count=255 370 ==> class=anomaly 357 conf:(0.96)

=== Evaluation ===

Elapsed time: 8.87s

Improved C45 results:

Size of the tree : 1631

Time taken to build model: 1.32 seconds
Time taken to test model on training data: 0.09 seconds

=== Error on training data ===

Correctly Classified Instances	5119	96.7492 %
Incorrectly Classified Instances	172	3.2508 %
Mean absolute error	0.0153	
Relative absolute error	10.2084 %	
Total Number of Instances	5291	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.994	0.037	0.966	0.994	0.98	0.997	normal
0.988	0.027	0.969	0.988	0.978	0.997	anomaly
0.069	0	1	0.069	0.129	0.977	neptune
0.073	0	1	0.073	0.136	0.977	teardrop
0.174	0	1	0.174	0.296	0.992	back
0.16	0	1	0.16	0.276	0.984	land
0.154	0	1	0.154	0.267	0.99	smurf
Weighted Avg. 0.967	0.031	0.968	0.967	0.958	0.997	

=== Confusion Matrix ===

a	b	c	d	e	f	g	<-- Tree classified as
2703	15	0	0	0	0	0	a = normal
30	2399	0	0	0	0	0	b = anomaly
9	18	2	0	0	0	0	c = neptune
17	21	0	3	0	0	0	d = teardrop
13	6	0	0	4	0	0	e = back
12	9	0	0	0	4	0	f = land
14	8	0	0	0	0	4	g = smurf

6. CONCLUSION AND FUTURE WORK

In this work, we have proposed an efficient scalable Improved decision tree construction algorithm which results in high processing speed and small scale.Proposed work also tested association between attacks using apriori and improved apriori algorithms.Because of this reason, it is most suitable for large datasets. Our proposed algorithms improved apriori and c45 algorithms has many advantages, but the important thing is that it requires only one pass over the training dataset for the entire construction of decision tree. So it significantly reduces the IO cost. Moreover, our algorithm provides a general framework that can be used with any existing decision tree construction algorithms and requires only one time sorting for the numerical attribute. Hence, it reduces the sorting cost of numerical attributes and execution time of partitioning phase in the decision tree construction process. From the experimental evaluation, we have got a promising result, since our proposed algorithm outperforms the Existing C45 algorithm in execution time.

Type of algorithm→	C45	IMPROVED C 4.5
Accuracy	95	96

REFERENCES:

[1] Quinlan J R, ”Simplifying Decision Tree,” Internet Journal of Man-Machine Studies,1987,27,pp.221-234

[2] Yang Xue-bing,Zhang Jun, ”Decision Tree Algorithm and its core technology,” Computer Technology and Development,2007, 17(1),pp.43-45

[3] Qu Kai-she,Wen Cheng-li,Wang Jun-hong, ”An improved algorithm of ID3 algorithm,” Computer Engineering and Applications, 2003,(25),pp.104-107

[4] Mao Cong-li□Yi Bo, ”The most simple decision tree generation algorithm based on decision-making degree of coordination ,“ Computer Engineering and Design,2008,29(5),pp.1250-1252

[5] Huang Ai-hui, ”Improvement and application of decision tree C4.5 algorithm ,“ Science Technology and Engineering,2009, (1),pp.34-37

[6] J. Gehrke, R. Ramakrishnan, and V. Ganti, "Rainforest, a framework for fast decision tree construction of large datasets", in Springer Netherlands-Data mining and knowledge discovery vol.4. Issue(2-3) July 2000.

[7] M. Kantardzic “Data Mining. Concepts, Models, Methods and Algoritms”. John Wiley and Sons Inc, 2003.

[8] Xu.M.Wang, J. and Chen.T. “Improved decision tree algorithm: ID3+” Intelligent Computing in Signal Processing and Pattern Recognition, Vol.345, pp.141-149, 2006.

[9] Quinlan, J. R. “C4.5: Programs for Machine Learning” Morgan Kaufmann, San Mateo, CA 1993.

[10] Lewis, R.J. “An Introduction to Classification and Regression Tree (CART) Analysis” Annual Meeting of the Society for Academic Emergency Medicine, Francisco 2000.

[11] Ruoming Jin, Ge Yang and Gagan Agrawal, “Shared memory parallelization of Data mining algorithms: Techniques, Programming interface and Performance”, IEEE Transactions on Knowledge & data engineering, 2005.

[12] Song Xudong, Cheng Xiaolan “Decision tree Algorithm based on Sampling” IFIP International conference on Netwok and Parallel Computing-Workshops 2007.