International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-3 Issue-2, December 2013

Intrusion Detection System Based On Improved One versus All Data Stream Classification

Komal Gandle, Pallavi Kulkarni

Abstract*— with the marvelous development of information Technology & Network Security the Intrusion Detection (ID) has rapidly become a crucial component of any network defense strategy. Data Stream Classification is the superlative method for revealing of Intrusion Detection (ID). Improved One Versus All (OVA) is one of the multiclass classification techniques On the basis of this we propose the system on Network Intrusion Detection (NID) for security in network as well as computer. In this paper, improved one versus all decision tree algorithms identifies the behavioral attacks actions and newly arising attacks of intrusions. This paper addresses the excellent advantages of Improved OVA data stream classification such as Low error correlation and concept change. Also propose a new learning algorithm for illuminating of network intrusion Detection.

Index Terms*— Improved OVA decision tree, Intrusion Detection (ID)

I. INTRODUCTION

Improved OVA is the k-class classification problem, learns one binary classifier for each class to distinguish the instances of this class from instances of the all remaining classes [1]. In encyclopedic computer & network technology for infrastructures in current era, the rate of intrusions increases dramatically with irrespective of time. Intrusion detection (ID) is the process of monitoring & analyzing the events occurred in network or system activities for malicious activities or policy violations or security policies, also compromise the discretion, authenticity or availability of computer or networks [2]. Network Intrusion Detection Systems (NIDS) become most important standard component in security infrastructures as they allow network administrators to detect policy violations. By using Improved One Versus All Data Stream Classification algorithm for detecting intrusions is now essential measure and adaptive intrusion detection systems that detect unconstitutional activities of a computer system or networks.

A. Intrusion Detection (ID)

Intrusion detection is the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a resource. The intrusion detection systems are intensely serious component in the network security resource. The process of monitoring and analyzing the events occurring in a computer and/or network system in order to detect signs of security problems is called intrusion detection.

Denning defines the principle for characterizing a system under attack. The system which is not under attack must satisfy the three conditions:

1. Actions of users conform to statistically predictable patterns.
2. Actions of users do not include sequences which violate the security policy.
3. Actions of every process correspond to a set of specifications which describe what the process is allowed to do.

Systems under attack do not meet at least one of the three conditions. Basically, intrusion detection is based upon some rules which are true unrelatedly of the approach permitted by the intrusion detection system. These assumptions are:

1. There exists a security policy which defines the normal and /or the abnormal usage of every resource.
2. The patterns generated during the abnormal system usage are different from the patterns generated during the normal usage of the system; i.e., the abnormal and normal usage of a system results in different system behavior. This dissimilarity in behavior can be used to detect intrusions.

Distinct methods are used for intrusions detection which makes a number of assumptions that are specific only to the particular method in accumulation to the designation of the security policy and the access patterns. These are used in the learning phase of the detector; the attack detection capability of an intrusion detection system depends upon the assumptions made by individual methods for intrusion detection.

B. Working of Intrusion Detection Systems

An intrusion detection system normally consists of three sub systems or modules:

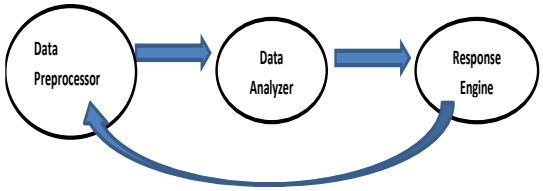

Fig. 1 Architecture of Intrusion Detection Systems

1.2.1 Data Preprocessor

Data preprocessor is responsible for collecting and providing the audit data in a specified format that can be used by the next component (analyzer) to make a decision. Data preprocessor is apprehensive collecting the data from the desired source and converting it into a format that is understandable by the analyzer [14].

Manuscript published on 30 December 2013.
* Correspondence Author (s)
Ms. Komal Gandle*, Assistant Professor, Information Technology Dept. Government College of Engineering, Aurangabad (M.S.) India.
Mrs. Pallavi Kulkarni, Assistant Professor, Computer Science & Engineering Dept. Government College of Engineering, Aurangabad (M.S.) India.

Data used for detecting intrusions range from user access patterns. Example, the sequence of commands issued at the terminal and the resources requested to network packet level features such as the source and destination IP addresses type of packets and rate of occurrence of packets to application and system level behavior such as the sequence of system calls generated by a process. This data is referred as the audit patterns.

1.2.2 Intrusion Detector or Analyzer

The analyzer or the intrusion detector is the fundamental component which analyzes the audit patterns to detect attacks. This is an acute component and one of the most examined. There are numerous pattern matching, machine learning, data mining and statistical techniques used as intrusion detectors. The capability of the analyzer to detect an attack frequently determines the strength of the overall system [14].

1.2.3 Response Engine

The response engine deals with the reaction mechanism and determines how to respond when the analyzer detects an attack. The system may decide either to raise an alert without taking any action against the source or may decide to block the source for a predefined period of time. Such an action depends upon the predefined security policy of the network system.

IDS was first taken by James P. Anderson in 1980[7], and later in 1986, Dr. Dorothy Denning projected several models for IDS based on statistics, Markov chains, time-series, etc. Inconsistency based intrusion detection using data mining algorithms like decision tree (DT), Associative Rules, naïve Bayesian Classifier (NB), Pure feed – forward Neural Network (NN), support vector machine (SVM), k-nearest neighbors (KNN), fuzzy logic model, and genetic algorithm have been used widely, because it deals with interesting features deals with the terms of solution quality, speed, adaptability and Extensive computational results will be presented and compared to the latest evolutionary IDS[7].

C. Consistency Based ID

Commercially available IDS having a wide range of data fusion and correlation capabilities and they are Signature based performs pattern harmonizing techniques which match consistency in an attack pattern or signature with known attack patterns in the database. It requires definite rule set or signature can grow very fast. Signature-based attacks can only catch attacks that are known and for which signatures have been created to pretend attacks to avoid false negative (FN). The minor variation in an attack is enough to defeat a signature; it yields very low false positives (FP) [2]. The only solution is immeasurable number of rules, which lost performance of the system and increases complexity and hence THAT ARE NO LONGER ADEQUATE TO CONFLICT WITH NEW ATTACKS ACTIONS.

D. Inconsistency Based ID

Inconsistency based IDS forms models of normal behavior and automatically detects uncharacteristic behaviors. Anomaly detection techniques identify new types of intrusions, but the catch the rate of false positives (FP)[2]. Actually an intrusion attack catches the rate of false negatives (FN). The main issue relates with anomaly based ID system is to classic threshold level to deal with the attack actions. Anomaly based ID systems are computational more expensive because they deal with the rate of keeping the track of or required more updates and preserved several system information. Improved OVA Classification algorithms for anomaly based IDS include an intelligent agent in the system that can recognized identified and unidentified attacks or interruptions. An anomaly based intrusion detection system that employed naïve Bayesian network to perform intrusion detecting on traffic bursts [7]. Classification based on anomaly detection using inductive rules to describe the sequences occurring in normal data [8]. The Fuzzy Intrusion Recognition Engine (FIRE) using fuzzy logic that process the network input data and generate fuzzy sets for every observed feature and then the fuzzy sets are used to define fuzzy rules to detect individual attacks [9]. FIRE creates and applies fuzzy rules to the audit data to classify it as normal or anomalous. In another paper the anomalous network traffic detection with self-organizing maps using DNS and HTTP services for network based IDS that the neurons are trained with normal network traffic then real time network data is fed to the trained neurons [10], if the distance of the incoming network traffic is more than a preset threshold then it raises an alarm.

E. Intrusion detection System

Intrusion detection schemes (IDS) gather and analyze information from a variety of schemes. According to the monitored system the source of input information can be on host or network or host and network. Thus IDS can be host-based or network based schemes. Host-based IDS located in servers to examine the internal interfaces and network-based IDS monitor the network traffics for detecting interruptions.

Network-based IDS performs packet classification, real-time network traffic analysis, watches active services and server, and tries to discover as well as report to possibly stop network level attacks [3]. The major advantage of NIDS are attacks information from different subnet can be correlated which lead to attack can be stopped early systems movement, auditing system configuration, assessing the data files, recognizing known attacks identifying strange activities. NIDS have issues like Encrypted data cannot be read and Annoyances to normal traffic if for some reason normal traffic is dropped. In the current IDS also deals with some issues like low-slung detection accuracy, unbalanced detection rates for different types of attacks, and high false positives.

F. Improved OVA classifier

For a k-class classification problem, OVA learn one binary classifier for each class to distinguish instances of this class from instances of the remaining (k-1) classes. To classify an instance, the k binary classifiers are run and one that earnings the highest confidence is chosen. OVA is accurate in component classifiers are well adjusted. In OVA, a false negative is more unsuccessful than a false positive [1]. Because a false negative result in a 1/n probability of correct prediction, while for a false positive, this probability is 1/2. Accordingly, to weight each instance so that the component classifiers are more prepared to say “yes” and have reported that weighted OVA can overtake unweight OVA.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Then classifiers' outputs are combined. Low error correlation lead to high diversity among the component classifiers and hence archives the more accuracy in classification and Handling data streams is a difficult task due to the variations in the data and the frequent occurrences of concept change.

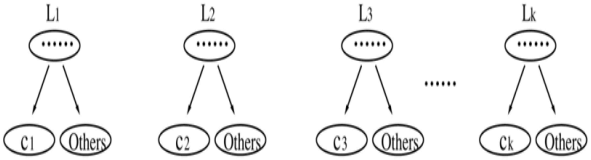

Fig 1: OVA is decomposed into k- class classification problem to k binary classification problem

Features of Improved OVA classifier

- By well training data sets reduces the updating cost
- More accuracy in class distribution as compared with previous standard methods
- Very efficient in deal with concept change
- High diversity gives more accuracy.

II. PROPOSED IMPROVED OVA LEARNING ALGORITHM

A. *Decision Tree*

The decision tree (DT) is very flexible data mining algorithm which is widely used for decision-making, classification problems and regression problems. The real time applications like medical diagnosis, radar signal classification, weather prediction, credit approval, and fraud detection etc. DT can be constructed from enormous volume of dataset with numerous attributes; the key feature is the tree size is independent of the dataset size[4]. A decision tree devises three main components: nodes, leaves, and edges. Each node is labeled with a one attribute versus all attributes data is to be partitioned. Each node has a number of edges, which are labeled rendering to possible values of the attribute. An edge connects either two nodes or a node and a leaf. Leaves are labeled with a decision value for categorization of the data. To make a decision using a decision Tree, start at the root node and follow the tree down the branches until a leaf node representing the class is reached. Each node of decision tree represents a rule set, which categorizes data according to the attributes of dataset. The DT algorithms initially build the tree and then ensure effective classification [5]. By pruning technique, used to reduce the overall size of the tree. The time and space complexity of a decision tree depends on the size of the data set, the number of attributes in the data set, and the shape of the resulting tree. Decision trees are used to classify data with common attributes. The ID3 algorithm constructs decision tree by using information theory, which choose splitting attributes from a data set with the highest information gain [6]. The amount of information associated with an attribute value is associated to the probability of occurrence. The IFN method is empirically to produce more compact and stable models than the "meta-learner" techniques, though preserving a reasonable level of predictive accuracy and stability [15]. Entropy concept is used to calculate information gain, (here to measure the amount of randomness from a data set).

The objective of decision tree classification is to iteratively partition the given data set into subsets where all elements in each final subset belong to the same class.

Probabilities $L_1, L_2, \ldots, L_s$ for different classes in the data set. Entropy $(L_1, L_2, \ldots, L_s) = \sum_{i=1}^{s} L_i \log(1/L_i)$ given a dataset, A, *H (A)* finds the amount of entropy in class based subsets of the dataset. When that subset is split into s new subsets *S = {A1, A2… As}* using some attribute, we have to look at the entropy of those subsets again itself. A subset of dataset is completely ordered and does not need any further split if all instances in it belong to the same class. The information gain of a split is calculated by ID3algorithm and chooses the split which provides maximum information gain.

Gain (A,S) = H(A)-$\sum_{i=1}^{s} p(Ai)H(Ai)$

The C4.5 algorithm [13], which is the upgraded version of ID3 algorithm uses highest *Gain* (Classification and Regression Trees) is a process of generating a binary tree for decision making [5]. CAR Handles missing data and contains a pruning strategy. The SPRINT (Scalable Parallelizable Induction of Decision Trees) algorithm uses an impurity function called *gini* index to find the best split [12].

gini (D) = 1-$\Sigma\, p_j^2$

Where, *pj* is the probability of class *Cj* in data set *D*. The goodness of a split of *D* into subsets *D1* and *D2*is defined by

$gini_{split}(D) = n1/n(gini(D1)) + n2/n(gini(D2))$

B. *Proposed Learning Algorithm*

In a given dataset, first the algorithm initializes the weights for each instance of dataset; *Wi*= 1/*n*, where *n* is the number of total examples in dataset and calculate the prior probability *P(Cj)* for each class by summing the weights that how repeatedly each class occurs in the dataset. Also for each attribute, *Ai*, the number of occurrences of each attribute value *Aij* can be counted by summing the weights to determine *P(Aij)*. Correspondingly, the conditional probabilities *P(Aij | Cj)* are expected for all values of attributes by summing the weights how repeatedly each attribute value occurs in the class *Cj*. Then algorithm uses these probabilities to update the weights for each example in the dataset. It is achieved by multiplying the probabilities of the different attribute values from the examples.

Suppose the example *ei* has independent attribute values *{Ai1, Ai2,…,Aip}*. *P(Aik | Cj)*, for each class *Cj* and attribute *Aik*. We then estimate *P(ei | Cj)* by *P(ei | Cj) = P(Cj) Πk=1→p P(Aij | Cj)* To update the weight, we can estimate the probability of *ei* in each class *Cj*. Probability *P(Cj | ei)* is found for each class. Now the weight of the example is updated with the highest next probability for that example.

Finally, the algorithm calculates the information gain by using updated weights and builds a tree for decision making [6]. Following describes the main procedure of algorithms:

Algorithm: Tree Construction

Input: dataset D

Output: decision tree T

Procedure:

1. Initialize all the weights in D, Wi=1/n, where n is the total number of the examples.
2. Calculate the prior probabilities P(Cj) for each class Cj In D P(Cj) $= \frac{\sum_{cl} w_i}{\sum_{i=1}^{n} w_i}$
3. Calculate the conditional probabilities P (Aij | Cj) for each attribute values in D.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

$$P(Aij \mid Cj) = \frac{P(Aij)}{\sum_i w_i}$$

4. Calculate the next probabilities for each example in D.
   P(ei | Cj) = P(Cj) Π P(Aij | Cj)
5. Update the weights of examples in D with Maximum Possibility (MP) of next probability P(Cj|ei);
   Wi= PML (Cj|ei)
6. Find the splitting attribute with highest information gain using the updated weights, Wi in D.
7. T = Create the root node and label with splitting attribute.
8. For each branch of the T, D = database created by applying splitting predicate to D, and continue steps 1 to 7 until each final subset belongs to the same class or leaf node created.
9. When the decision tree construction is completed the algorithm terminates.

III. REQUIREMENT ANALYSIS

A. *Data set collection*

To verify the reliability and feasibility of proposed system the KDD99 standard network intrusion detection system dataset is used in comparison with other standing approaches [11]. KDD99 becomes more popular dataset due to the following features:

- Redundant records form training set are eliminated
- Duplicate records are removed form test set which leads to increase the performance of the system.
- It gives tremendous accurate in learning technology.
- Reliability which gives adequate number of instance in both train set and test sets.

B. *Types of Attacks*

1) Probing attacks

The attacker scans a network to collect information or find known vulnerabilities through machines and the services of the networks are available can use of exploitation. Probe attacks are having different types like abuses the computer legitimate features or social engineering techniques. This requires less technical expertise [14].

2) Denial of Service (DoS)

The attacker makes computing or memory resource too busy or too full to handle legitimate requests, thus denying legitimate users access to a machine [14]. DoS attacks are having different types like abusing the computer legitimate features, by exploiting the system misconfigurations, by targeting the implementation bugs.

3) User to Root (U2R)

The attacker starts out with access to a normal user account on the system and able to exploit vulnerability to gain the root access of the system [14]. U2R attacks are like regular buffer overflows are causes the regular programming mistakes and environment assumptions.

4) Remote to Local (R2L)

The attacker sends the packets to the machine over the network and then exploits the machine vulnerability to illegally gain local access as a user. These attacks are carried out mostly by using social engineering.

IV. CONCLUSION

This paper presents a new learning algorithm for anomaly based network intrusion detection using decision tree, on the basis of improved one versus all criteria. In this paper, we advanced the performance of IDS using decision tree. In conventional decision tree algorithm weights of every example is set to equal value which challenges general intuition, but in our approach weights of every instance concept transformation based on probability.

ACKNOWLEDGMENTS

Our thanks to experts who have contributed towards development of the work.

REFERENCES

[1] Hashemi, S.; Ying Yang; Mirzamomen, Z.; Kangavari, M.; "Adapted One-versus-All Decision Trees for Data Stream Classification," Knowledge and Data Engineering, IEEE Transactions on ,vol.21,no.5,pp.624-637,May 2009 doi: 10.1109/TKDE.2008.181

[2] Wenke Lee and Salvatore J. Stolfo Data Mining Approaches for Intrusion Detection Computer Science Department
Columbia University
500 West 120th Street, New York, NY 10027
{wenke,sal}@cs.columbia.edu

[3] Theodoros Lappas and Konstantinos Pelechrinis;" Data Mining Techniques for (Network) Intrusion Detection Systems" Department of Computer Science and Engineering UC Riverside, Riverside CA 92521

[4] Ruoming Jin:" Efficient Decision Tree Construction on Streaming Data", Ohio State University, Columbus OH 43210jinr@cis.ohiostate.edu

[5] L. Breiman, J. H. Friedman, R. A. Olshen and C.J. Stone, "Classification and Regression Trees," Statistics probability series, Wadsworth, Belmont, 1984.

[6] J. R. Quinlan, "Induction of Decision Tree," Machine Learning Vol. 1,pp. 81-106, 1986

[7] Barbara, Daniel, Couto, Julia, Jajodia, Sushil, Popyack, Leonard, Wu,and Ningning, "ADAM: Detecting intrusion by data mining," IEEEWorkshop on Information Assurance and Security, West Point, New York, June 5-6, 2001.

[8] W. Lee, S.J. Stolfo, "Data mining approaches for intrusion detection," In
Proc. of the 7th USENIX Security Symposium (SECURITY-98),Berkeley, CA, USAedwdf, 1998, pp. 79-94.

[9] J.E. Dickerson, J.A. Dickerson, "Fuzzy network profiling for intrusion detection," In Proc. of the 19th International Conference of the NorthAmerican Fuzzy Information Processing Society (NAFIPS), Atlanta,GA, 2000, pp. 301-306

[10] M. Ramadas, S.O.B. Tjaden, "Detecting anomalous network traffic with
self-organizing maps," In Proc. of the 6th International Symposium onRecent Advances in Intrusion Detection, Pittsburgh, PA, USA, 2003, pp.36-54.

[11] The KDD Archive KDD99 cup dataset, 1999 http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[12] John Shafer, Rakesh Agarwal, and Manish Mehta, SPRINT: A Scalable Parallel Classifier for Data Maining," in Proceedings of the VLDB conference, Bombay, India, September 1996.

[13] J. R. Quinlan, "C4.5: Programs for Machine Learning," MorganKaufmann Publishers, San Mateo, CA, 1993.

[14] Srinivas Mukkamala, Andrew Sung and Ajith Abraham;" Designing Intrusion Detection Systems: Architectures, Challenges and Perspectives", Department of Computer Science, New Mexico Tech, USA

[15] Mark Last," Improving Stability of Decision Trees", Department of Information Systems Engineering, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)