International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:4, No:3, 2010

Attacks Classification in Adaptive Intrusion Detection using Decision Tree

Dewan Md. Farid, Nouria Harbi, Emna Bahri, Mohammad Zahidur Rahman, Chowdhury Mofizur Rahman

***Abstract*—**Recently, information security has become a key issue in information technology as the number of computer security breaches are exposed to an increasing number of security threats. A variety of intrusion detection systems (IDS) have been employed for protecting computers and networks from malicious network-based or host-based attacks by using traditional statistical methods to new data mining approaches in last decades. However, today's commercially available intrusion detection systems are signature-based that are not capable of detecting unknown attacks. In this paper, we present a new learning algorithm for anomaly based network intrusion detection system using decision tree algorithm that distinguishes attacks from normal behaviors and identifies different types of intrusions**.** Experimental results on the KDD99 benchmark network intrusion detection dataset demonstrate that the proposed learning algorithm achieved 98% detection rate (DR) in comparison with other existing methods.

***Keywords*—**Detection rate, decision tree, intrusion detection system, network security.

I. Introduction

As advances in computer network technology expand for communications and commerce in recent times, the rate of intrusions increase more than double every year. Intrusion detection is the process of identifying actions that attempt to compromise the confidentiality, integrity or availability of computers or networks. The use of data mining algorithms for detecting intrusions is now considered to build efficient and adaptive intrusion detection systems (IDS) that detect unauthorized activities of a computer system or network. IDS was first introduced by James P. Anderson in 1980 [1], and later in 1986, Dr. Dorothy Denning proposed several models for IDS based on statistics, Markov chains, time-series, etc [2]. Anomaly based intrusion detection using data mining algorithms such as decision tree (DT), naïve Bayesian classifier (NB), neural network (NN), support vector machine (SVM), k-nearest neighbors (KNN), fuzzy logic model, and genetic algorithm have been widely used by researchers to improve the performance of IDS [3]-[8]. However, today's commercially available IDS are signature based. Signature based IDS performs pattern matching techniques to match an attack pattern corresponding to known attack patterns in the database and produces very low false positives (FP), but it requires regular updates of rules or signatures and not capable of detecting unknown attacks. On the other hand, anomaly based IDS builds models of normal behavior and automatically detects anomalous behaviors. Anomaly detection techniques identify new types of intrusions as deviations from normal usage [9], but the drawback of these techniques is the rate of false positives (FP). The use of data mining algorithms for anomaly based IDS are to include an intelligent agent in the system that can detect the known and unknown attacks or intrusions.

Intrusion detection systems (IDS) gather and analyze information from a variety of systems and network sources for signs of intrusions. IDS can be host-based or network based systems. Host-based IDS located in servers to examine the internal interfaces and network-based IDS monitor the network traffics for detecting intrusions. Network-based IDS performs packet logging, real-time traffic analysis of IP network, and tries to discover if an intruder is attempting to break into the network. The major functions performed by IDS are: (1) monitoring users and systems activity, (2) auditing system configuration, (3) assessing the data files, (4) recognizing known attacks, (5) identifying abnormal activities, (6) managing audit data, (7) highlighting normal activities, (8) correcting system configuration errors, and (9) stores information about intruders. A variety of IDS have been employed for protecting computers and networks in last decades, but still there some issues that should be consider in the current IDS like low detection accuracy, unbalanced detection rates for different types of attacks, and high false positives. In this paper, we proposed a new decision tree based learning algorithm for classifying different types of network attacks, which improves the detection rates (DR) and reduces false positives (FP) using KDD99 benchmark network intrusion detection dataset in comparison with other existing methods.

The remainders of the paper are organized as follows. Section II presents the various approaches for anomaly based intrusion detection systems. Our proposed algorithm for

Dewan Md. Farid is with the ERIC Laboratory, University Lumière Lyon 2 – 5 av. Pierre Mendes, France – 69676 BRON Cedex, France (phone: +33 0648882531; fax: +33 478772375; e-mail: dewanfarid@gmail.com).

Nouria Harbi is with the ERIC Laboratory, University Lumière Lyon 2– France (e-mail: nouria.harbi@univ-lyon2.fr).

Emna Bahri is with the ERIC Laboratory, University Lumière Lyon 2– France (e-mail: emna.bahri@univ-lyon2.fr).

Mohammad Zahidur Rahman is with the Department of Computer Science and Engineering, Jahangirnagar University, Bangladesh (e-mail: rmzahid@juniv.edu).

Chowdhury Mofizur Rahman is with the United International University, Bangladesh (e-mail: crmuiu@gmail.com).

anomaly based network intrusion detection system is introduced in Section III. In Section IV, the experimental results are expressed. Finally, our conclusions and future works are mentioned in Section V.

II. Anomaly Based Intrusion Detection Systems

In 1980, the concept of IDS began with Anderson's seminal paper [1]; he introduced a threat classification model that develops a security monitoring surveillance system based on detecting anomalies in user behavior. In Anderson's model threats are classified as external penetrations, internal penetrations, and misfeasance. External penetrations are intrusions in computer system by outside intruders, who do not have any authorized access to the system that they attack. Internal penetrations are intrusions in computer system by inside intruders. Inside intruders are users in the network and have some authority, but seek to gain additional ability to take action without legitimate authorization. Misfeasance is defined as the misuse of authorized access of both to the system and to its data. In 1986, Dr. Dorothy Denning mentioned several models for commercial IDS development based on statistics, Markov chains, time-series, etc [2]. In Denning model, user's behavior that deviates sufficiently from the normal behavior is considered anomalous. In the early 1980's, Stanford Research Institute (SRI) developed an Intrusion Detection Expert System (IDES) that continuously monitored user behavior and detected suspicious events [10]. Later SRI developed an improved version of IDES called the Next-Generation Intrusion Detection Expert System (NIDES) [11], [12] that could operate in real time for continuous monitoring of user activity or could run in a batch mode for periodic analysis of the audit data, an audit data is a record of activities generated by the operating system that are logged to a file in chronologically sorted order. NIDES enable the system to compare the current activities of the user/system/network with the audited intrusion detection variables stored in the profile and then raise an alarm if the current activity is sufficiently far from the stored audited activity. In 1988, a statistical anomaly-based IDS was proposed by Haystack [13], which used both user and group-based anomaly detection strategies. In this system, a range of values were considered normal for each attribute and during a session if an attribute fell outside the normal range then an alarm raised. It was designed to detect six types of intrusions: attempted break-ins by unauthorized users, masquerade attacks, penetration of the security control system, leakage, denial of service, and malicious use. Statistical Packet Anomaly Detection Engine (SPADE) [14] is a statistical anomaly intrusion detection system that is available as a plug-in for SNORT that an open source network intrusion detection and prevention system (NIDPS) developed by Sourcefire [15], [16].

In 1996, Forrest et al. proposed an analogy between the human immune system and intrusion detection that involved analyzing a program's system call sequences to build a normal profile [17], which analyzed several UNIX, based programs like sendmail, ipr, etc. If the sequences deviated from the normal sequence profile then it considered as an attack. The system they developed was only used off-line using previously collected data and used a quite simple table-lookup algorithm to learn the profiles of programs. In 2000, Valdes et al. [18] developed an anomaly based intrusion detection system that employed naïve Bayesian network to perform intrusion detecting on traffic bursts. In 2003, Kruegel et al. [19] proposed a multisensory fusion approach using Bayesian classifier for classification and suppression of false alarms that the outputs of different IDS sensors were aggregated to produce single alarm. In the same year, Shyu et al. [20] proposed an anomaly based intrusion detection scheme using principal components analysis (PCA), where PCA was applied to reduce the dimensionality of the audit data and arrive at a classifier that is a function of the principal components. In another paper, Yeung et al. [21] proposed an anomaly based intrusion detection using hidden Markov models that computes the sample likelihood of an observed sequence using the forward or backward algorithm for identifying anomalous behavior from normal behaviors. Lee et al. [22] proposed classification based anomaly detection using inductive rules to characterize sequences occurring in normal data. In 2000, Dickerson at al. [23] developed 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. FIRE creates and applies fuzzy rules to the audit data to classify it as normal or anomalous. In another paper, Ramadas et al. [24] presented 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, if the distance of the incoming network traffic is more than a preset threshold then it rises an alarm. Another network based anomaly detection using data mining techniques developed by Minnesota Intrusion Detection System (MINDS) in 2004 [25].

III. Proposed Learning Algorithm

A. Decision Tree Learning

The decision tree (DT) is very powerful and popular data mining algorithm for decision-making and classification problems. It has been using in many real life applications like medical diagnosis, radar signal classification, weather prediction, credit approval, and fraud detection etc. DT can be constructed from large volume of dataset with many attributes, because the tree size is independent of the dataset size. A decision tree has three main components: nodes, leaves, and edges. Each node is labeled with an attribute by which the data is to be partitioned. Each node has a number of edges, which are labeled according 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 decision tree represents a rule set, which categorizes data according to the attributes of dataset. The DT building algorithms may initially build the tree and then prune it for more effective classification. With pruning technique, portions of the tree may be removed or combined to reduce the overall size of the tree. The time and space complexity of constructing 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 builds decision tree using information theory, which choose splitting attributes from a data set with the highest information gain [26]. The amount of information associated with an attribute value is related to the probability of occurrence. The concept used to quantify information is called entropy, which is used to measure the amount of randomness from a data set. When all data in a set belong to a single class, there is no uncertainty, and then the entropy is zero. 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. The entropy calculation is shown in equation 1. Given probabilities $p_1, p_2,..,p_s$ for different classes in the data set

$$Entropy: H(p_1,p_2,...p_s) = \sum_{i=1}^{s} (p_i \log(1/p_i)) \quad (1)$$

Given a data set, D, $H(D)$ finds the amount of entropy in class based subsets of the data set. When that subset is split into s new subsets $S = \{D_1, D_2, ..., D_s\}$ using some attribute, we can again look at the entropy of those subsets. A subset of data set is completely ordered and does not need any further split if all examples in it belong to the same class. The ID3 algorithm calculates the information gain of a split by using equation 2 and chooses that split which provides maximum information gain.

$$Gain\ (D,S) = H(D) - \sum_{i=1}^{s} p(D_i)H(D_i) \quad (2)$$

The C4.5 algorithm [27], which is the upgraded version of ID3 algorithm uses highest *Gain Ratio* in equation 3 for splitting purpose that ensures a larger than average information gain.

$$GainRatio(D,S) = \frac{Gain(D,S)}{H\left(\frac{|D_1|}{|D|},...,\frac{|D_s|}{|D|}\right)} \quad (3)$$

The C5.0 algorithm improves the performance of building trees using boosting, which is an approach to combining different classifiers. But boosting does not always help when the training data contains a lot of noise. When C5.0 performs a classification, each classifier assigns a vote and the example is assigned to the class with the most number of votes. CART (Classification and Regression Trees) is a process of generating a binary tree for decision making [28]. CART 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 [29].

$$gini\ (D) = 1 - \sum p_j^2 \quad (4)$$

Where, p_j is the probability of class C_j in data set D. The goodness of a split of D into subsets D_1 and D_2 is defined by

$$gini_{split}(D) = n_1/n(gini(D_1)) + n_2/n(gini(D_2)) \quad (5)$$

B. Proposed Learning Algorithm

In a given dataset, first the algorithm initializes the weights for each example of dataset; W_i equal to $1/n$, where n is the number of total examples in dataset. Then the algorithm estimates the prior probability $P(C_j)$ for each class by summing the weights that how often each class occurs in the dataset. Also for each attribute, A_i, the number of occurrences of each attribute value A_{ij} can be counted by summing the weights to determine $P(A_{ij})$. Similarly, the conditional probabilities $P(A_{ij} \mid C_j)$ are estimated for all values of attributes by summing the weights how often each attribute value occurs in the class C_j. After that the algorithm uses these probabilities to update the weights for each example in the dataset. It's performed by multiplying the probabilities of the different attribute values from the examples. Suppose the example e_i has independent attribute values $\{A_{i1}, A_{i2},...,A_{ip}\}$. We already know $P(A_{ik} \mid C_j)$, for each class C_j and attribute A_{ik}. We then estimate $P(e_i \mid C_j)$ by

$$P(e_i \mid C_j) = P(C_j) \prod_{k=1 \to p} P(A_{ij} \mid C_j) \quad (6)$$

To update the weight, we can estimate the likelihood of e_i in each class C_j. The probability that e_i is in a class is the product of the conditional probabilities for each attribute value. The posterior probability $P(C_j \mid e_i)$ is then found for each class. Now the weight of the example is updated with the highest posterior probability for that example. Finally, the algorithm calculates the information gain by using updated weights and builds a tree for decision making. 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, $W_i=1/n$, where n is the total number of the examples.
2. Calculate the prior probabilities $P(C_j)$ for each class C_j in D. $P(C_j) = \frac{\sum_{Ci} W_i}{\sum_{i=1}^{n} W_i}$

3. Calculate the conditional probabilities $P(A_{ij} \mid C_j)$ for each attribute values in D. $P(A_{ij} \mid C_j) = \frac{P(A_{ij})}{\sum_{C_i} W_i}$
4. Calculate the posterior probabilities for each example in D.
$$P(e_i \mid C_j) = P(C_j) \prod P(A_{ij} \mid C_j)$$
5. Update the weights of examples in D with Maximum Likelihood (ML) of posterior probability $P(C_j|e_i)$;
$$W_i = P_{ML}(C_j|e_i)$$
6. Find the splitting attribute with highest information gain using the updated weights, W_i 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 belong to the same class or leaf node created.
9. When the decision tree construction is completed the algorithm terminates.

IV. Experimental Analysis

A. Intrusion Detection Dataset

The KDD99 dataset was used in the 3rd International Knowledge Discovery and Data Mining Tools Competition for building a network intrusion detector, a predictive model capable of distinguishing between intrusions and normal network connections [30]. In 1998, DARPA intrusion detection evaluation program, a simulated environment was set up to acquire raw TCP/IP dump data for a local-area network (LAN) by the MIT Lincoln Lab to compare the performance of various intrusion detection methods. It was operated like a real environment, but being blasted with multiple intrusion attacks and received much attention in the research community of adaptive intrusion detection. The KDD99 dataset contest uses a version of DARPA98 dataset. In KDD99 dataset, each example represents attribute values of a class in the network data flow, and each class is labeled either normal or attack. The classes in KDD99 dataset categorized into five main classes (one normal class and four main intrusion classes: probe, DOS, U2R, and R2L).

1) Normal connections are generated by simulated daily user behavior such as downloading files, visiting web pages.

2) Denial of Service (DoS) attack causes the computing power or memory of a victim machine too busy or too full to handle legitimate requests. DoS attacks are classified based on the services that an attacker renders unavailable to legitimate users like apache2, land, mail bomb, back, etc.

3) Remote to User (R2L) is an attack that a remote user gains access of a local user/account by sending packets to a machine over a network communication, which include send-mail, and Xlock.

4) User to Root (U2R) is an attack that an intruder begins with the access of a normal user account and then becomes a root-user by exploiting various vulnerabilities of the system. Most common exploits of U2R attacks are regular buffer-overflows, load-module, Fd-format, and Ffb-config.

5) Probing (Probe) is an attack that scans a network to gather information or find known vulnerabilities. An intruder with a map of machines and services that are available on a network can use the information to look for exploits.

In KDD99 dataset these four attack classes (DoS, U2R, R2L, and probe) are divided into 22 different attack classes that tabulated in Table I.

TABLE I.
Different Types of Attacks in KDD99 Dataset

4 Main Attack Classes	22 Attack Classes
Denial of Service (DoS)	back, land, neptune, pod, smurt, teardrop
Remote to User (R2L)	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster
User to Root (U2R)	buffer_overflow, perl, loadmodule, rootkit
Probing	ipsweep, nmap, portsweep, satan

There are 41 input attributes in KDD99 dataset for each network connection that have either discrete or continuous values and divided into three groups. The first group of attributes is the basic features of network connection, which include the duration, prototype, service, number of bytes from source IP addresses or from destination IP addresses, and some flags in TCP connections. The second group of attributes in KDD99 is composed of the content features of network connections and the third group is composed of the statistical features that are computed either by a time window or a window of certain kind of connections. Table II shows the number of examples of 10% training examples and 10% testing examples in KDD99 dataset. There are some new attack examples in testing data, which is no present in the training data.

TABLE II.
Number of Examples in Training and Testing KDD99 Data

Attack Types	Training Examples	Testing Examples
Normal	97277	60592
Denial of Service	391458	237594
Remote to User	1126	8606
User to Root	52	70
Probing	4107	4166
Total Examples	494020	311028

B. Experimental Analysis

In order to evaluate the performance of proposed algorithm for network intrusion detection, we performed 5-class classification using KDD99 dataset. All experiments were performed using an Intel Core 2 Duo Processor 2.0 GHz processor (2 MB Cache, 800 MHz FSB) with 1 GB of RAM. The results of the comparison of proposed algorithm with ID3 and C4.5 algorithms are tabulated in Table III and Table IV.

TABLE III.
Comparison of the Results using 41 Attributes

Method	Normal	Probe	DOS	U2R	R2L
Proposed Algorithm (DR %)	98.76	98.21	98.55	98.11	97.16
Proposed Algorithm (FP %)	0.07	0.44	0.05	0.12	6.85
ID3 (DR %)	97.63	96.35	97.41	43.21	92.75
ID3 (FP %)	0.10	0.55	0.04	0.14	10.03
C4.5 (DR %)	98.53	97.85	97.51	49.21	94.65
C4.5 (FP %)	0.10	0.55	0.07	0.14	11.03

TABLE IV.
COMPARISON OF THE RESULTS USING 19 ATTRIBUTES

Method	Normal	Probe	DOS	U2R	R2L
Proposed Algorithm (DR %)	99.19	99.15	99.26	98.43	98.05
Proposed Algorithm (FP %)	0.06	0.48	0.04	0.10	6.32
ID3 (DR %)	98.71	98.22	97.63	86.11	94.19
ID3 (FP %)	0.06	0.51	0.04	0.12	7.34
C4.5 (DR %)	98.81	98.22	97.73	56.11	95.79
C4.5 (FP %)	0.08	0.51	0.05	0.12	8.34

V. CONCLUSION

This paper presents a new learning algorithm for anomaly based network intrusion detection using decision tree, which adjusts the weights of dataset based on probabilities and split the dataset into sub-dataset until all the sub-dataset belongs to the same class. In this paper, we developed the performance of IDS using decision tree. In conventional decision tree algorithm weights of every example is set to equal value which contradicts general intuition, but in our approach weights of every example change based on posterior probability. The experimental results on KDD99 benchmark dataset manifest that proposed algorithm achieved high detection rate on different types of network attacks. The future research issues will be to test it extensively in real world problem domains.

ACKNOWLEDGMENT

Support for this research received from ERIC Laboratory, University Lumière Lyon 2 – France, and Department of Computer Science and Engineering, Jahangirnagar University, Bangladesh.

REFERENCES

[1] James P. Anderson, "Computer security threat monitoring and surveillance," Technical Report 98-17, James P. Anderson Co., Fort Washington, Pennsylvania, USA, April 1980.
[2] Dorothy E. Denning, "An intrusion detection model," IEEE Transaction on Software Engineering, SE-13(2), 1987, pp. 222-232.
[3] Barbara, Daniel, Couto, Julia, Jajodia, Sushil, Popyack, Leonard, Wu, and Ningning, "ADAM: Detecting intrusion by data mining," IEEE Workshop on Information Assurance and Security, West Point, New York, June 5-6, 2001.
[4] N.B. Amor, S. Benferhat, and Z. Elouedi, "Naïve Bayes vs. decision trees in intrusion detection systems," In Proc. of 2004 ACM Symposium on Applied Computing, 2004, pp. 420-424.
[5] Mukkamala S., Janoski G., and Sung A.H., "Intrusion detection using neural networks and support vector machines," In Proc. of the IEEE International Joint Conference on Neural Networks, 2002, pp.1702-1707.
[6] J. Luo, and S.M. Bridges, "Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection," International Journal of Intelligent Systems, John Wiley & Sons, vol. 15, no. 8, 2000, pp. 687-703.
[7] YU Yan, and Huang Hao, "An ensemble approach to intrusion detection based on improved multi-objective genetic algorithm," Journal of Software, vol. 18, no. 6, June 2007, pp. 1369-1378.
[8] Shon T., Seo J., and Moon J., "SVM approach with a genetic algorithm for network intrusion detection," In Proc. of 20th International Symposium on Computer and Information Sciences (ISCIS 2005), Berlin: Springer-Verlag, 2005, pp. 224-233.
[9] Lazarevic, A., Ertoz, L., Kumar, V., Ozgur,. A., Srivastava, and J., "A comparative study of anomaly detection schemes in network intrusion detection," In Proc. of the SIAM Conference on Data Mining, 2003.
[10] Dorothy E. Denning, and P.G. Neumann "Requirement and model for IDES- A real-time intrusion detection system," Computer Science Laboratory, SRI International, Menlo Park, CA 94025-3493, Technical Report # 83F83-01-00, 1985.
[11] D. Anderson, T. Frivold, A. Tamaru, and A. Valdes, "Next generation intrusion detection expert system (NIDES)," Software Users Manual, Beta-Update Release, Computer Science Laboratory, SRI International, Menlo Park, CA, USA, Technical Report SRI-CSL-95-0, May 1994.
[12] D. Anderson, T.F. Lunt, H. Javitz, A. Tamaru, and A. Valdes, "Detecting unusual program behavior using the statistical component of the next generation intrusion detection expert system (NIDES)," Computer Science Laboratory, SRI International, Menlo Park, CA, USA, Technical Report SRI-CSL-95-06, May 1995.
[13] S.E. Smaha, and Haystack, "An intrusion detection system," in Proc. of the IEEE Fourth Aerospace Computer Security Applications Conference, Orlando, FL, 1988, pp. 37-44.
[14] N. Ye, S.M. Emran, Q. Chen, and S. Vilbert, "Multivariate statistical analysis of audit trails for host-based intrusion detection," IEEE Transactions on Computers 51, 2002, pp. 810-820.
[15] Martin Roesch, "SNORT: The open source network intrusion system," Official web page of Snort at http://www.snort.org/
[16] L. C. Wuu, C. H. Hung, and S. F. Chen, "Building intrusion pattern miner for sonrt network intrusion detection system," Journal of Systems and Software, vol. 80, Issue 10, 2007, pp. 1699-1715.
[17] S. Forrest, S.A. Hofmeyr, A. Somayaji, T.A. Longstaff, "A sense of self for Unix processes," in Proc. of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, USA, 1996, pp. 120-128.
[18] A. Valdes, K. Skinner, "Adaptive model-based monitoring for cyber attack detection," in Recent Advances in Intrusion Detection Toulouse, France, 2000, pp. 80-92.
[19] C. Kruegel, D. Mutz, W. Robertson, F. Valeur, "Bayesian event classification for intrusion detection," in Proc. of the 19th Annual Computer Security Applications Conference, Las Vegas, NV, 2003.
[20] M.L. Shyu, S.C. Chen, K. Sarinnapakorn, L. Chang, "A novel anomaly detection scheme based on principal component classifier," in Proc. of the IEEE Foundations and New Directions of Data Mining Workshop, Melbourne, FL, USA, 2003, pp. 172-179.
[21] D.Y. Yeung, Y. Ding, "Host-based intrusion detection using dynamic and static behavioral models," Pattern Recognition 36, 2003, pp. 229-243.
[22] W. Lee, S.J. Stolfo, "Data mining approaches for intrusion detection," In Proc. of the 7th USENIX Security Symposium (SECURITY-98), Berkeley, CA, USA, 1998, pp. 79-94.
[23] J.E. Dickerson, J.A. Dickerson, "Fuzzy network profiling for intrusion detection," In Proc. of the 19th International Conference of the North American Fuzzy Information Processing Society (NAFIPS), Atlanta, GA, 2000, pp. 301-306.
[24] M. Ramadas, S.O.B. Tjaden, "Detecting anomalous network traffic with self-organizing maps," In Proc. of the 6th International Symposium on Recent Advances in Intrusion Detection, Pittsburgh, PA, USA, 2003, pp. 36-54.
[25] L. Ertoz, E. Eilertson, A. Lazarevic, P.N. Tan, V. Kumar, J. Srivastava, P. Dokas, "The MINDS: Minnesota intrusion detection system," In: Next Generation Data Mining, MIT Press, Boston, 2004.
[26] J. R. Quinlan, "Induction of Decision Tree," Machine Learning Vol. 1, pp. 81-106, 1986.
[27] J. R. Quinlan, "C4.5: Programs for Machine Learning," Morgan Kaufmann Publishers, San Mateo, CA, 1993.
[28] L. Breiman, J. H. Friedman, R. A. Olshen and C.J. Stone, "Classification and Regression Trees," Statistics probability series, Wadsworth, Belmont, 1984.
[29] 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.
[30] The KDD Archive. KDD99 cup dataset, 1999. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html