[Vanve* *et al.*, 5(7): July, 2016]
IC™ Value: 3.00

ISSN: 2277-9655
Impact Factor: 4.116

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

OGEDIDS: OPPOSITIONAL GENETIC PROGRAMMING ENSEMBLE FOR DISTRIBUTED INTRUSION DETECTION SYSTEMS

Shrikant Vanve*, Prof. Sarita Patil
* G.H. Raisoni College of Engg & Management Research, Pune
G.H. Raisoni College of Engg & Management Research, Pune

DOI: 10.5281/zenodo.57737

ABSTRACT

Due to the wide range application of internet and computer networks, the securing of information is indispensable one. In order to secure the information system more effectively, various distributed intrusion detection has been developed in the literature. In this paper, we utilize the oppositional genetic algorithm for Distributed Network Intrusion Detection utilizing the oppositional set based population selection mechanism. This system is mostly useful for detecting unauthorized & malicious attack in distributed network. Here, Oppositional genetic algorithm (OGA) is utilized in OGA ensemble for learning the intrusion detection behavior of networks. Also, OGA ensemble is adapted for distributed intrusion detection system by creating the network profile which classifies normal and abnormal behavior of network. For experimentation, network profile contains different classifier which uses training data set of KDD Cup 99 to generate intrusion rules. For validation, we utilize the confusion matrix, sensitivity, specificity and accuracy and the results are proved that the proposed OGEdIDS are better for intrusion detection.

KEYWORDS: Genetic algorithm, intrusion detection, ensemble learner, KDD cup 99, accuracy.

INTRODUCTION

Nowadays, networked computer systems play an increasingly important role in our society and its economy. They have become the targets of a wide array of malicious attacks that invariably turn into actual intrusions. This is the reason computer security has become an essential concern for network administrators. Intrusion detection as is an approach to counter the computer and networking attacks and misuses. There are two generally accepted categories of intrusion detection techniques: misuse detection and anomaly detection. Misuse detection refers to techniques that characterize known methods to penetrate a system. Anomaly detection refers to techniques that define and characterize normal or acceptable behaviors of the system [1].

Intrusion Detection Systems (IDS) are primarily focused on identifying possible incidents, logging information about them, attempting to stop them, and reporting them to security administrators in real-time, or near real-time, and those that process audit data with some delay (non-real-time). In addition, organizations use IDSs for other purposes, such as identifying problems with security policies, documenting existing threats, and deterring individuals from violating security policies. A typical Intrusion Detection System is shown in figure 1.

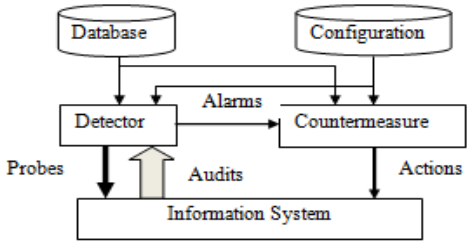

Figure 1. Very simple intrusion detection system

Genetic algorithm is a family of computational models based on principles of evolution and natural selection. These algorithms convert the problem in a specific domain into a model by using a chromosome-like data structure and evolve the chromosomes using selection, recombination, and mutation operators. Genetic Algorithm (GA) has been used in different ways in IDSs. One network connection and its related behavior can be translated to represent a rule to judge whether or not a real-time connection is considered an intrusion. These rules can be modeled as chromosomes inside the population. The population evolves until the evaluation criteria are met. The generated rule set can be used as knowledge inside the IDS for judging whether the network connection and related behaviors are potential intrusions.

In this paper, OGEdIDS, Oppositional Genetic programming Ensemble is developed for Distributed Intrusion Detection Systems. This proposed distributed data mining algorithm utilize the oppositional genetic algorithm for Distributed Network Intrusion Detection utilizing the oppositional set based population selection mechanism. Here, Oppositional genetic algorithm (OGA) is utilized in OGA ensemble for learning the intrusion detection behavior of networks. Also, OGA ensemble is adapted for distributed intrusion detection system by creating the network profile which classifies normal and abnormal behavior of network. The paper is organized as follows: Section 2 presents the literature review and section 3 presents the proposed distributed intrusion detection system. Section 4 presents the results and conclusion is given section 5.

LITERATURE REVIEW

Several Genetic Algorithms (GAs) and Genetic Programming (GP) has been used for detecting intrusion detection of different kinds in different scenarios [8-14]. There are several papers related to IDS which has certain level of impact in network security. Li [2] described a method using GA to detect anomalous network intrusion. The approach includes both quantitative and categorical features of network data for deriving classification rules. However, the inclusion of quantitative feature can increase detection rate but no experimental results are available. Information gain could become more relevant when attribute interactions are taken into account. This phenomenon is associated with rule interestingness.

Goyal and Kumar [3] described a GA based algorithm to classify all types of smurf attack using the training dataset. Lu and Traore [4] used historical network dataset using GP to derive a set of classification. They used support confidence framework as the fitness function and accurately classified several network intrusions. But their use of genetic programming made the implementation procedure very difficult and also for training procedure more data and time is required. Xia et al. [5] used GA to detect anomalous network behaviors based on information theory. Some network features can be identified with network attacks based on mutual information between network features and type of intrusions and then using these features a linear structure rule and also a GA is derived. The approach of using mutual information and resulting linear rule seems very effective because of the reduced complexity and higher detection rate. Gong et al. [6] presented an implementation of GA based approach to Network Intrusion Detection using GA and showed software implementation. The approach derived a set of classification rules and utilizes a supportconfidence framework to judge fitness function.

PROPOSED OPPOSITIONAL GENETIC PROGRAMMING ENSEMBLE FOR DISTRIBUTED INTRUSION DETECTION SYSTEMS

In this paper, Oppositional Genetic programming Ensemble for Distributed Intrusion Detection Systems is developed. This works aims to develop a distributed data mining algorithm based on the ensemble paradigm that employs an Oppositional Genetic Programming-based classifier as component learner in order to improve the detection capability of the system [7].

Oppositional genetic algorithm (OGA)

This section presents the oppositional genetic algorithm (OGA). The main difference in an oppositional GP, with respect to a genetic algorithm, is its oppositional set based population selection mechanism and the genetic operators (crossover, mutation) adopted. At the beginning, initial population is generated and the initial population is modified based on the oppositional set concept. Then, the fitness of each individual is evaluated. Then, at each generation, every chromosome undergoes one of the genetic operators (reproduction, crossover, mutation) depending on the probability test. If crossover is applied, the mate of the current individual is selected as the neighbor having the best fitness, and the offspring is generated. The current chromosome is then replaced by the best of the two offsprings if the fitness of

the latter is better than that of the former. The evaluation of the fitness of each classifier is calculated on the entire training data. After the execution of the number of generations defined by the user, the individual with the best fitness represents the classifier. Figure 2 shows the algorithmic description of the OGA.

Algorithm: OGA
Start
Generate initial population
Compute opposition set-based population
while not MaxNumberOfGeneration **do**
for each chromosome *i* in the population **do**
produce the offspring by crossing *ti* and *tj*
Produce the offspring by mutation
evaluate the fitness of the offspring
replace t_i with the best of the two offspring if its fitness is better than that of t_i
evaluate the fitness of the new t_i
Keep the best individual in the population
Endfor
end while
end

Figure 2. The algorithm OGA

OGA ensemble: OGA for ensemble learner

The OGA is applied here for ensemble learner. Ensemble is a learning paradigm where multiple component learners are trained for the same task by a learning algorithm, and the predictions of the component learners are combined for dealing with new unseen instances. Let $S = \{(x_i, y_i) | i = 1,....N\}$ be a training set where x_i, called example or tuple or instance, is an attribute vector with m attributes and y_i is the class label associated with x_i. A predictor (classifier), given a new example, has the task to predict the class label for it. Ensemble techniques build *T* predictors, each on a different training set, then combine them together to classify the test set. Boosting was introduced for boosting the performance of any "weak" learning algorithm, i.e. an algorithm that "generates classifiers which need only be a little bit better than random guessing".

The boosting algorithm, called *AdaBoost*, adaptively changes the distribution of the training set depending on how difficult each example is to classify. Given the number *T* of trials (rounds) to execute, *T* weighted training sets $S_1, S_2, \ldots, S_T$ are sequentially generated and *T* classifiers $C^1, \ldots, C^T$ are built to compute a weak hypothesis h_t. Let w_i^t denote the weight of the example x_i, at trial *t*. At the beginning $w_i^1 = 1/n$ for each x_i,. At each round $t = 1, \ldots,$ *T*, a weak learner C^t, whose error ε^t is bounded to a value strictly less than 1/2, is built and the weights of the next trial are obtained by multiplying the weight of the correctly classified examples by $\beta^t = \varepsilon^t / (1 - \varepsilon^t)$ and renormalizing the weights so that $\sum_i w_i^{t+1} = 1$. Thus "easy" examples get a lower weight, while "hard" examples, that tend to be misclassified, get higher weights. This induces AdaBoost to focus on examples that are hardest to classify. The boosted classifier gives the class label *y* that maximizes the sum of the weights of the weak hypotheses predicting that label, where the weight is defined as $\log(1/\beta^t)$. The final classifier h_f is defined as follows:

$$h_f = \arg\max\left(\sum_t^T \log\left(\frac{1}{\beta^t}\right) h_t(x, y)\right)$$

Ensemble techniques have been shown to be more accurate than component learners constituting the ensemble, thus such a paradigm has become a hot topic in recent years and has already been successfully applied in many application fields. A key feature of the ensemble paradigm, often not much highlighted, concerns its ability to solve problems in a distributed and decentralized way. We adopt such a paradigm to derive a network profile for modeling distributed intrusion detection systems using OGA as component learner.

An OGA ensemble offers several advantages over a monolithic GA that uses a single GP to solve the intrusion detection task. First, it can deal with very large data sets. Second, it can make an overall system easier to understand, modify and implement in a distributed way. Finally, it is more robust than a monolithic GP, and can show graceful performance degradation in situations where only a subset of GPs in the ensemble are performing correctly. One of the disadvantages of such an approach is the loss of interaction among the individual GPs during the learning phase. In fact, the individual GPs are often trained independently or sequentially. This does not take into account the interdependence that exists among the data and could bring to an ensemble overfitting them and having weak characteristics of generalization. To this aim, we propose a method that emphasizes the OGA in the ensemble during the building of the solution.

Adapting OGA ensemble for distributed intrusion detection system

Once the OGA ensemble is developed, it is then applied for distributed intrusion detection system. The pseudo-code of the OGEdIDS algorithm is shown in figure 3. Each island is furnished with a OGA algorithm enhanced with the boosting technique AdaBoost. OGA, a population initialized with random individuals, and operates on the local audit data weighted according to a uniform distribution. The selection rule, the replacement rule and the asynchronous migration strategy are specified in the OGA algorithm. Each island generates the OGA ensemble by running for a certain number of iterations, necessary to compute the number of boosting rounds. During the boosting rounds, each classifier maintains the local vector of the weights that directly reflect the prediction accuracy on that site. At each boosting round the hypotheses generated by each classifier are exchanged among all the processors in order to produce the ensemble of predictors. In this way each island maintains the entire ensemble and it can use it to recalculate the new vector of weights. After the execution of the fixed number of boosting rounds, the classifiers are used to evaluate the accuracy of the classification algorithm for intrusion detection on the entire test set.

Given a network constituted by P nodes, each having a data set S_j
For $j = 1, 2, \ldots, P$ (for each island in parallel)
Initialize the weights associated with each tuple
Initialize the population Q_j with random individuals
end parallel for
For $t = 1,2,3, \ldots, T$ (boosting rounds)
For $j = 1, 2, \ldots, P$ (for each island in parallel)
Train OGA on S_j using a weighted fitness according to the weight distribution
Compute a weak hypothesis
Exchange the hypotheses among the P islands
Update the weights
end parallel for
end for t
Output the hypothesis

Figure 3. The OGEdIDS algorithm

RESULTS AND DISCUSSION

This section presents the experimental results and validation of the proposed OGEdIDS algorithm.

Data sets description

We performed experiments over the KDD Cup 1999 Data set [15]. Though this, data set has been judged not representative of a realistic IDS scenario, it is a reference data set, extensively used to compare results of different intrusion detection techniques. The data set comes from the 1998 DARPA Intrusion Detection Evaluation Data and contains a training data consisting of 7 weeks of network based attacks inserted in the normal data, and 2 weeks of network-based attacks and normal data for a total of 4,999,000 of connection records described by 41 characteristics.

The main categories of attacks are four: DoS (Denial of Service), R2L (unauthorized access from a remote machine), U2R (unauthorized access to a local superuser privileges by a local unprivileged user), PROBING (surveillance and

ISSN: 2277-9655
Impact Factor: 4.116

probing). However a smaller data set consisting of the 10% the overall data set is generally used to evaluate algorithm performance. In this case the training set consists of 494,020 records among which 97,277 are normal connection records, while the test set contains 311,029 records among which 60,593 are normal connection records. Table 1 shows the distribution of each attack type in the training and the test set. Note that the test set is not from the same probability distribution as the training data, in fact it includes specific attack types not in the training data. This makes the task more realistic.

Table 1 shows the distribution of each intrusion type in the training and the test set.

Dataset	normal	probe	dos	u2r	r2l	Total
Train ("kddcup.data_10_percent")	97280	4107	391458	52	1124	494021
Test ("corrected")	60593	4166	229853	228	16189	311029

Performance metrics

To evaluate the proposed system, three standard metrics of sensitivity, specificity and accuracy developed for network intrusions, have been used. Table 1 shows the confusion matrix which is a table with two rows and two columns that report the number of false positives, false negatives, true positives, and true negatives. This allows more detailed analysis than mere proportion of correct guesses (accuracy). Accuracy is not a reliable metric for the real performance of a classifier, because it will yield misleading results if the data set is unbalanced (that is, when the number of samples in different classes vary greatly).

Sensitivity=TP/(TP+FN)

Specificity= TN/(FP+TN)

Accuracy=(TP+TN)/(TP+TN+FP+FN)

Performance evaluation

This section presents the performance evaluation of the proposed IDS using KDD cup data. Figure 4 shows the confusion metrics-based graph for the different percentage of training data. The better performance is achieved by the proposed IDS when the percentage of training data is equal to 60%. Here, TP, FP, TN and FN of the methods are 790, 54, 737, and 107. Similarly Figure 5 shows the performance graph of the proposed IDS using sensitivity, specificity and accuracy. Here, the better accuracy of 90.41% is achieved when the percentage of training data is equal to 60%.

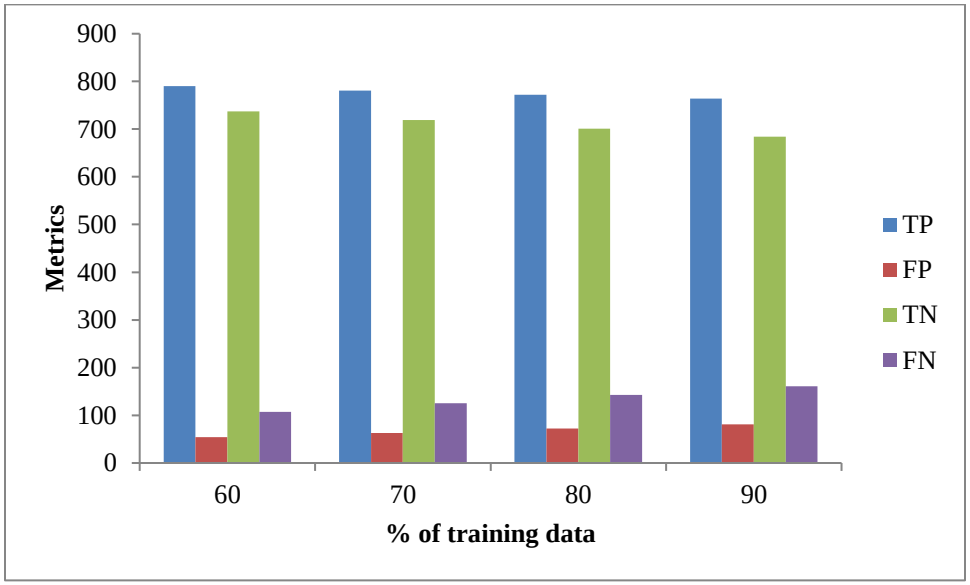

Figure 4 Performance graph of confusion metrics

ISSN: 2277-9655
Impact Factor: 4.116

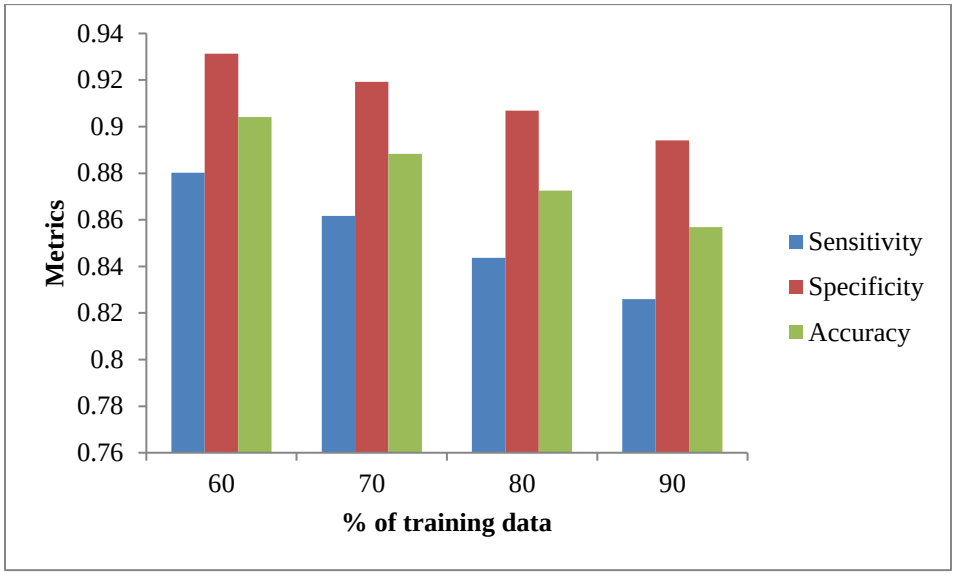

Figure 5 Performance graph using sensitivity, specificity and accuracy

CONCLUSION

A distributed intrusion detection approach based on Oppositional Genetic Programming and extended with the ensemble paradigm, to classify malicious or unauthorized network activity has been presented. Here, OGP ensembles are built using a distributed approach based on a hybrid model that combines the ensemble learner and the OGA. The combination of these two models provides an effective implementation of distributed Intrusion Detection Systems, namely, OGEdIDS. A main advantage of the distributed architecture is that it enables for flexibility, extensibility, and efficiency since each node of the network works with its local data, and communicates with the other nodes, to obtain the results, only the local model computed, but not the data. For experimentation, KDD Cup 99 is used and the validation is performed using confusion matrix, sensitivity, specificity and accuracy and the results are proved that the proposed OGEdIDS are better for intrusion detection.

REFERENCES

[1] M. Botha, R. Solms, "Utilizing Neural Networks for Effective Intrusion Detection", ISSA, 2004.
[2] W. Li, "Using Genetic Algorithm for Network Intrusion Detection". "A Genetic Algorithm Approach to Network Intrusion Detection". SANS Institute, USA, 2004
[3] Anup Goyal, Chetan Kumar, "GA-NIDS: A Genetic Algorithm based Network Intrusion Detection System", 2008.
[4] W. Lu, I. Traore, "Detecting New Forms of Network Intrusion Using Genetic Programming". Computational Intelligence, vol. 20, pp. 3, Blackwell Publishing, Malden, pp. 475-494, 2004.
[5] T. Xia, G. Qu, S. Hariri, M. Yousif, "An Efficient Network Intrusion Detection Method Based on Information Theory and Genetic Algorithm", Proceedings of the 24th IEEE International Performance Computing and Communications Conference (IPCCC '05), Phoenix, AZ, USA. 2005.
[6] R. H. Gong, M. Zulkernine, P. Abolmaesumi, "A Software Implementation of a Genetic Algorithm Based Approach to Network Intrusion Detection", 2005.
[7] Folino, G., Pizzuti, C., and Spezzano, G. "An Ensemble Based Evolutionary Framework For Coping With Distributed Intrusion Detection", Genetic Programming and Evolvable Machines 11, 2, 2010, 131-146.
[8] Chi Hoon Lee, Sung Woo Shin and Jin Wook Chung, "Network Intrusion Detection Through Genetic Feature Selection", SNPD, IEEE, 2006.
[9] Saqib Ashfaq, M.Umar Farooq and Asim Karim, "Efficient Rule Generation for Cost- Sensitive Misuse Detection Using Genetic Algorithms", IEEE, 2006.

ISSN: 2277-9655
Impact Factor: 4.116

[10] Melanie Middlemiss and Grant Dick, "Weighted Feature Extraction Using a Genetic Algorithm for Intrusion Detection", 2003 Congress on Evolutionary Computation (cec-03) 2003, pp.1669-1675.
[11] Nalini N and Raghavendra Rao G., "Network Intrusion Detection via a Hybrid of Genetic Algorithms and Principal Component Analysis", IEEE, 2006.
[12] Hua Zhou, Xingu Meng and Li Zhang, "Application of Support Vector Machine and Genetic Algorithm to Network Intrusion Detection", IEEE, 2007.
[13] Yong Wang, Dawu Gu, Xiuxia Tian and Jing Li, "Genetic Algorithm Rule Definition for Denial of Services Network Intrusion Detection", International Conference on Computational Intelligence and Natural Computing, IEEE, 2009, pp.99-102.
[14] Chen Zhongmin, Feng Jianyuan, Xu Sheng and Xu Renzuo, "The research of Intrusion Detection Technology Based on Genetic Algorithms", International Conference on Net-works Security, Wireless Communications and Trusted Computing, IEEE, 2009.
[15] KDD Cup 1999: Data; http://www.kdd.org/kddcup/index.php?section=1999&method=data