Combining Genetic-Based Misuse and Anomaly Detection for Reliably Detecting Intrusions in Computer Networks*

I. Finizio, C. Mazzariello, and C. Sansone

Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli “Federico II”
Via Claudio 21, I-80125 Napoli, Italy
{ifinizio, cmazzari, carlosan}@unina.it

Abstract. When addressing the problem of detecting malicious activities within network traffic, one of the main concerns is the reliability of the packet classification. Furthermore, a system able to detect the so-called *zero-day attacks* is desirable. Pattern recognition techniques have proven their generalization ability in detecting intrusions, and systems based on multiple classifiers can enforce the detection reliability by combining and correlating the results obtained by different classifiers.

In this paper we present a system exploiting genetic algorithms for deploying both a misuse-based and an anomaly-based classifier. Hence, by suitably combining the results obtained by means of such techniques, we aim at attaining a highly reliable classification system, still with a significant degree of new attack prediction ability. In order to improve classification reliability, we introduce the concept of rejection: instead of emitting an unreliable verdict, an ambiguous packet can be logged for further analysis. Tests of the proposed system on a standard database for benchmarking intrusion detection systems are also reported.

1 Introduction

The most common and best known tools used to ensure security of companies, campuses and, more in general, of any network, are Firewalls and Antiviruses. Though famous and well known, such tools alone are not enough to protect a system from malicious activities, and basing one's own site's security on the deployment of these instruments relies on the idea that intrusion prevention will suffice in efficiently assuring data availability, confidentiality and integrity. Indeed, an interesting idea about intrusions is that they will sooner or later happen, despite the security policy a network administrator deploys. Based on such assumption, many researchers started to develop systems able to successfully detect intrusions and, in some cases, trace the path leading to the attack source.

* This work has been partially supported by the Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) in the framework of the FIRB Project “Middleware for advanced services over large-scale, wired-wireless distributed systems (WEB-MINDS)”.

F. Roli and S. Vitulano (Eds.): ICIAP 2005, LNCS 3617, pp. 66 – 74, 2005.

On the basis of the information sources analyzed to detect an intrusive activity, the Intrusion Detection Systems (IDS) can be grouped into different categories. In the following, we will concentrate our attention on Network-based IDS (N-IDS) [1]. N-IDS analyze packets captured directly from the network: by setting network cards in promiscuous mode, an IDS can monitor traffic in order to protect all of the hosts connected to a specified network segment.

Depending on the detection technique employed, they can be roughly classified [2] as belonging to two main groups as well. The first one, that exploits signatures of known attacks for detecting when an attack occurs, is known as *misuse* (or *signature*) *detection* based. IDS's that fall in this category are based on a model of all the possible misuses of the network resources. The completeness request is actually their major limit [3].

A dual approach tries to characterize the normal usage of the resources under monitoring. An intrusion is then suspected when a significant difference from the resource's normal usage is revealed. IDS's following this approach, known as *anomaly detection* based, seem to be more promising because of their potential ability to detect unknown intrusions (the so-called *zero-day* attacks). However, there is also a major challenge, because of the need to acquire a model of the normal use general enough to allow authorized users to work without raising false alarms, but specific enough to recognize unauthorized usages [4,5].

The network intrusion detection problem can be also formulated as a binary classification problem: given information about network connections between pairs of hosts, the task is to assign each connection to one out of two classes that represent normal traffic conditions or an attack. Here the term "connection" refers to a sequence of data packets related to a particular service, such as a file transfer via the ftp protocol. In this framework, several proposals have been made in order to extract high-level features from data packets [6,7]. Each network connection can be then described by a "pattern" to be classified and a pattern recognition approach can be followed. The main advantage of such an approach is the ability to generalize exhibited by pattern recognition systems. They are able to detect some novel attacks, since different variants of the same attack will be typically described by very similar patterns. Moreover, the high-level features extracted from connections relative to a totally new attack should exhibit a behavior quite different from those extracted from normal connections. Summarizing, there isn't the need of a complete description of all the possible attack signatures. This overcomes one of the main drawbacks of the misuse detection approach.

Different pattern recognition systems have been reported in the recent past for realizing an IDS, mainly based on neural network architectures [4,8,9]. In order to improve the detection performance, approaches based on multi-expert architectures have been also proposed [10,11,12].

However, one of the main drawbacks occurring when using pattern recognition techniques in real environments is the high false alarm rate they often produce [10]. This is a very critical point in a real environment, as pointed out in [13].

In order to realize an IDS that is capable of detecting intrusion by keeping the number of false alarms as low as possible, in this paper we propose a genetic-based system that tries to combine some of the peculiarities of the misuse and of the anomaly detection approaches.

In particular, starting from the features proposed in [6], a genetic algorithm is used to generate two distinct sets of rules. The first set is devoted to individuate normal traffic connections (as in an anomaly detection approach), while the second one is suited for detecting attacks (following the misuse detection paradigm). A connection is then classified as an attack if it matches almost one of the rules of the second set and no one of the first set. On the other hand, a connection is labeled as normal if it matches almost one of the rules devoted to detecting normal traffic and no one of those generated for characterizing attacks. In all the other cases, the connection is rejected by the system, since it cannot be correctly classified with an high reliability. This permits to reduce the number of false alarms. Note that reject, in this case, means that the data about a 'rejected' connection are only logged for further processing, without raising an alert for the system manager.

It is worth noticing that other rule-based classifiers have been employed in an IDS (for example RIPPER [14], used by Lee and Stolfo in [6]). They, however, follow only the misuse detection approach, thus giving rise to a false alarm rate that cannot be acceptable in a typical real environment.

The organization of the paper is as follows: in Section 2 the proposed system is presented, while in Section 3 tests on a standard database used for benchmarking IDS are reported, together with a comparison of the proposed system with other pattern recognition systems. Finally, some conclusions are drawn.

2 A Genetic Approach for Generating Rules

As stated in the introduction, the proposed system is a rule-based one. Two sets of rules are generated, each one devoted to individuate a specific class, namely attacks or normal traffic. In order to classify a new traffic connection, the results of the two rule-based classifiers are suitably combined by means of a decision engine. In particular, if the feature vector describing a connection matches one of the rules related to the normal traffic and does not match any of the rules related to the attack class, it is attributed to the normal traffic class. Vice versa, if it matches almost a rule describing attacks and no one of the rules describing normal traffic, an alert is raised. In all the other cases, data about the connections are just logged for further processing (see Fig. 1).

Each set of rules is generated by means of a genetic algorithm based on a particular structure of the chromosomes. Such a structure was developed for suitably representing the boundaries of the region of the *n*-dimensional space containing the feature vectors representing the network connection belonging to the class the chromosome refers to.

Each chromosome consists of *n* genes. Each gene is associated to an element of the feature vector to be classified and is composed by a pair of values, x_{iMIN} and $x_{iMAX.}$ Such values represent, respectively, the lower and the upper limit of an interval. If the values of all the elements of the feature vector fall within the limits specified by the corresponding genes of a chromosome, this feature vector is attributed to the class the chromosome refers to. The minimum value that x_{iMIN} can assume is -∞, while the maximum value that x_{iMAX} can assume is +∞. The conversion from a rule to a chromosome and vice versa is immediate since they are simply two different ways to represent decision regions (see Fig. 2).

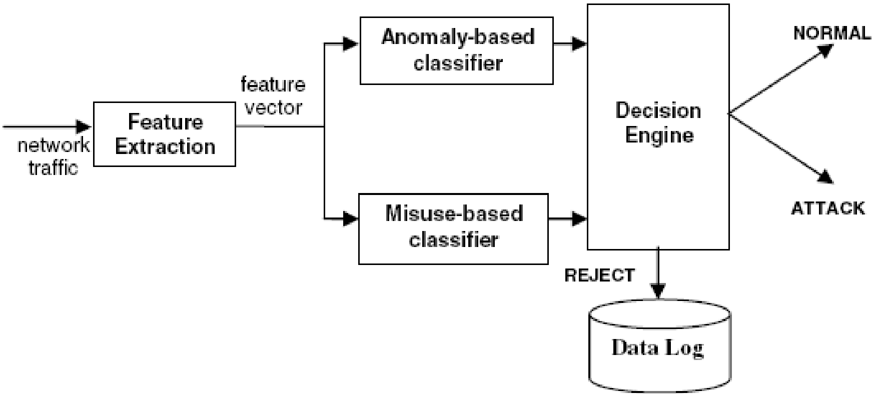

Fig. 1. A sketch of the proposed architecture for intrusion detection

X_{1MIN}	X_{2MIN}	....	X_{iMIN}	....	X_{nMIN}
X_{1MAX}	X_{2MAX}	....	X_{iMAX}	....	X_{nMAX}

⇕

$(x_{1MIN}<= x_1<= xb_{1MAX})$ AND $(x_{2MIN}<= x_2<= x_{2MAX})$ AND ... AND $(x_{iMIN}<= x_i<= x_{iMAX})$ AND ... AND $(x_{nMIN}<= x_n<= x_{nMAX})$

Fig. 2. Structure of a chromosome and the associated rule

Thus, each chromosome represents a hyper-region in the *n*-dimensional space. The aim of the proposed algorithm is to identify the region which the feature vectors belonging to a given class (normal or attack) lie into, that is, to select the corresponding chromosomes. The first step consists in the generation of an initial population of chromosomes, by assigning to each gene pairs of pseudo-random values. The assigned values are randomly selected from the set of all the values assumed within the whole training set by the corresponding elements, with the addition of the -∞ and +∞ values. The constraint to be observed is that, for each gene, the value of x_{iMIN} cannot be greater than the value of x_{iMAX}. After the computation of the fitness value for each generated chromosome, the reproduction process starts. A hybrid method was designed for the selection of the parents. This method can be considered as a binary selection double tournament with steady-state replacement. The algorithm randomly selects two pairs of chromosomes and, within each pair, compares the fitness. The fittest chromosomes of each selected pair are selected for reproduction and their two children take the place of the loser chromosomes. This technique promotes the elitism. In fact, the best chromosomes in each binary tournament are always winning and never substituted. After each step of the reproduction only the new individuals' fitness is recomputed and they are immediately ready for the reproduction. By using such a mating strategy, it is possible to use a promising individual just as soon as it is created. The used fitness function is:

$$Fitness = \frac{k_1 \cdot (neg + 1 + \left(\frac{elem}{k_3} \right))}{k_2 \cdot pos + 1} \tag{1}$$

where k_1, k_2 and k_3 are three parameters whose optimal values were fixed by an experimental investigation, *pos* and *neg* are respectively the number of feature vectors belonging to the training set which are correctly classified by the rule associated to the chromosome and the number of feature vectors belonging to the same set which are misclassified, *elem* is the number of elements of a feature vector not generalized to *any*. We assume that an element is generalized to *any* when the corresponding gene covers the whole set of real numbers, that is, when x_{iMIN}=-∞ and $x_{iMAX=}$+∞. Lower fitness values correspond to better chromosomes; therefore, in the comparison between two chromosomes, the one with the lower fitness is chosen. We insert the number of elements whose corresponding gene is not generalized to *any* in the fitness function in order to favour less complicated rules. Having fixed the value of all the other parameters, in fact, a rule with a simpler structure is associated to a chromosome characterized by a smaller value of the *elem* parameter. To obtain a behaviour as independent as possible from the training set used, we have adopted a uniform crossover strategy. For each reproduction step, a mask made by a pair of values for each gene of the chromosome is randomly generated. Each element of the mask contains the value 1 if it should prevail the present value in the first chromosome involved in the reproductive process, and 0 in the opposite case. The first one of two crossovers is carried out on the basis of such a mask. The second crossover is carried out using a mask obtained by complementing the previous one. It must be guaranteed that x_{iMIN}<= x_{iMAX} also in the crossover phase.

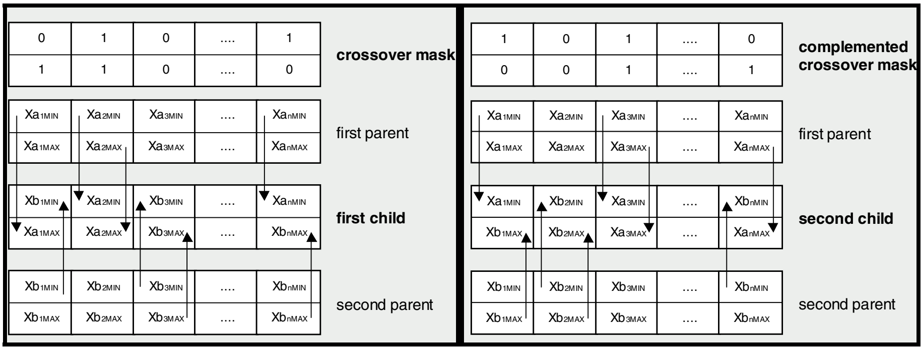
0 1 0 1
1 1 0 0
crossover mask
Xa1MIN Xa2MIN Xa3MIN .... XanMIN
Xa1MAX Xa2MAX Xa3MAX .... XanMAX
first parent
Xb1MIN Xa2MIN Xb3MIN .... XanMIN
Xa1MAX Xa2MAX Xb3MAX .... XbnMAX
first child
Xb1MIN Xb2MIN Xb3MIN .... XbnMIN
Xb1MAX Xb2MAX Xb3MAX .... XbnMAX
second parent
1 0 1 0
0 0 1 1
complemented crossover mask
Xa1MIN Xa2MIN Xa3MIN .... XanMIN
Xa1MAX Xa2MAX Xa3MAX .... XanMAX
first parent
Xa1MIN Xb2MIN Xa3MIN .... XbnMIN
Xb1MAX Xb2MAX Xa3MAX .... XanMAX
second child
Xb1MIN Xb2MIN Xb3MIN .... XbnMIN
Xb1MAX Xb2MAX Xb3MAX .... XbnMAX
second parent

Fig. 3. The crossover mechanism

At the end of the crossover, the new chromosomes undergo the mutation process. A mutation mechanism with incremental probability was proposed, in order to fully exploit the peculiarity of both crossover and mutation. We initialize the mutation probability to a very low value, then we progressively increase it during the genetic analysis. This technique offers the advantage of fully exploiting the mutation capacity of moving a solution far from local maxima. Then, it avoids the problems of a slow convergence and, even though in a smaller measure, of a premature convergence. Once the convergence has been reached, the obtained chromosomes are chosen and translated into a rule. If the rule associated to such chromosomes is able to correctly classify all the training set data, the process ends, otherwise it restarts and tries to identify a new chromosome able to classify the feature vectors not covered by the previously selected chromosome. At the end of the process, the set of rules that describes the whole decision region is composed by all the rules corresponding to the selected chromosomes.

3 Experimental Results

The proposed system has been tested on a subset of the database created by DARPA in the framework of the 1998 *Intrusion Detection Evaluation Program.* It is made up of a large number of network connections related to normal and malicious traffic. This database was pre-processed at the Columbia University giving rise to a feature vector of 41 elements for each connection, according to the set of features defined in [6] and tailored for the intrusion detection problem. Each connection in the database is labelled as belonging to normal traffic or to an attack. It is worth noticing that the attack class is made up of different variants, each one exploiting different vulnerabilities of a computer network.

The 1999 KDD intrusion detection contest used a version of this dataset. Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a local area network (LAN) simulating a typical U.S. Air Force LAN. They operated the LAN as if it was a true Air Force environment, but peppered it with multiple attacks. Even if this database has been collected in 1998, and some criticisms have been expressed on it [15], it is still widely used for testing the performance of an IDS [9,16].

The results obtained by means of the proposed system are reported in terms of *i*) the overall error rate on the classified connections, *ii*) the false alarm rate and the missed detection rate on the classified connections, and *iii*) the reject rate.

In the following we present the results obtained by our classification method applied to two different network services (*smtp* and *http*) among those present in the DARPA database. Other services have also been experimented, but the obtained results are not reported here for the sake of brevity. The choice of designing a different classifier system for each service follows the so-called modular approach presented in [11], where the authors experimentally demonstrate the advantage, in terms of recognition performance, of an IDS that develops a different classification module for each one of the network services to be protected.

For each service, a separate feature selection process was performed in order to reduce the data dimensionality. In particular, we have adopted a Sequential Forward Selection strategy, with the Minimum Estimated Probability classification criterion.

Moreover, different values of the parameters k_1, k_2 and k_3 (see eq.1) have been tested. The selected value has been chosen so as to maximize the obtainable results on the training set.

3.1 *Smtp* Service

In this case the training data was made up of 9723 patterns related to different attack variants and to the normal class. The Test Set (TS) for this service is made up of 3261 patterns with 3207 normal connections and 54 attack packets. After the feature selection process, each connection was described by a 6-dimensional feature vector. In particular, *duration*, *flag*, *src_bytes*, *hot*, *count* and *dst_src_host_same_src* features were selected (see [6] for their meanings).

Table 1 shows the results achieved by the proposed system on the TS. These results have been obtained by averaging ten different trials of the genetic algorithm for generating the two sets of rules.

As it is evident, we reach an ideal performance in terms of both missed detections and false alarms on the connections classified by the system. This excellent result is paid with about a 9% of reject rate.

Table 1. Results obtained by the proposed system on the TS for the *smtp* service

Overall error	*False alarm rates*	*Missed Detection rate*	*Reject rate*
0.00 %	0.00 %	0.00 %	9.12 %

Moreover, it is interesting to note that among the classified connections the proposed system is able to correctly detect over the 96% of the attacks to the *smtp* service.

3.2 *Http* Service

The training data for this service in the DARPA database are made up of 64292 patterns. However, in [9] it has been demonstrated that a dataset of about 15% of the whole *http* data is sufficient for training classifiers. Therefore, only 8866 samples have been considered as training data. The test set for this service is made up of 40442 patterns with 1195 attack packets and 39247 normal connections. After the feature selection process each connection was described by a 6-dimensional feature vector. The selected features were the same of the *smtp* service.

Table 2. Results obtained by the proposed system on the TS for the *http* service

Overall error	*False alarm rates*	*Missed Detection rate*	*Reject rate*
0.08 %	0.08 %	0.06 %	9.67 %

Table 2 shows the results achieved by the proposed system on the TS. Also in this case, the results reported here have been averaged on ten different trials of the genetic algorithm for generating the two sets of rules. In this case the system exhibits a quite negligible percentage of false alarms and missed detections. On the other hand, it must be noted that among the classified traffic connections, about the 44% of attacks were detected.

In order to make a comparison with other systems, it can be noted that the multi-stage classification system proposed in [12] achieved on the *http* connections a slightly higher false alarm rate (0.09%), while the multi-expert system proposed in [11] exhibited a false alarm plus missed detection rate of 0.54%. This confirms that our system is able to keep the number of false alarms low.

4 Conclusions

In this paper we proposed a genetic-based system for intrusion detection. A genetic algorithm is used for building two rule-based classifiers, a misuse-based one and an anomaly-based one. By suitably combining their *opinions* about each analyzed network connection, a decision engine improved the ability of the system in avoiding detection errors.

The proposed system showed a very encouraging behavior from the detection capability point of view. In particular, in case of the *smtp* service, we observe an error rate which is equal to 0%. On the other hand, we have a not negligible number of rejected packets.

Therefore, as a future development of the proposed architecture, we will work on the analysis of the rejected packets with slower but more accurate algorithms, in order to further improve the detection capability of the system.

References

1. G. Vigna, R. Kemmerer, “Netstat: a network based intrusion detection system”, Journal of Computer Security, vol. 7, no. 1, 1999.
2. S. Axelsson, Research in Intrusion Detection Systems: A Survey, TR 98-17, Chalmers University of Technology, 1999.
3. R. Kumar, E.H. Spafford, “A Software Architecture to Support Misuse Intrusion Detection”, in Proceedings of the 18th National Information Security Conference, pp. 194-204, 1995.
4. A.K. Ghosh, A. Schwartzbard, “A Study in Using Neural Networks for Anomaly and Misuse Detection”, Proc. 8'th USENIX Security Symposium, Aug. 26-29 1999, Washington DC.
5. T. Lane, C.E. Brodley, “Temporal Sequence learning and data reduction for anomaly detection”, ACM Trans. on Inform. and System Security, vol. 2, no. 3, pp. 295-261, 1999.
6. W. Lee, S.J. Stolfo, “A framework for constructing features and models for intrusion detection systems”, ACM Transactions on Inform. System Security, vol. 3, no. 4, pp. 227-261, 2000.
7. M. Esposito, C. Mazzariello, F. Oliviero, S.P. Romano, C. Sansone, “Real Time Detection of Novel Attacks by Means of Data Mining Techniques”, Proceedings of the 7th International Conference on Enterprise Information Systems, Miami (USA), May 24-28, 2005 (in press).

8. S. C. Lee, D.V. Heinbuch, “Training a neural Network based intrusion detector to recognize novel attack”, IEEE Trans. Syst, Man., and Cybernetic, Part-A, vol. 31, pp. 294-299, 2001.
9. M. Fugate, J.R. Gattiker, “Computer Intrusion Detection with Classification and Anomaly Detection, using SVMs”, International Journal of Pattern Recognition and artificial Intelligence, vol. 17, no. 3, pp. 441-458, 2003.
10. G. Giacinto, F. Roli, L. Didaci, “Fusion of multiple classifiers for intrusion detection in computer networks”, Pattern Recognition Letters, vol. 24, pp. 1795-1803, 2003.
11. G. Giacinto, F. Roli, L. Didaci, “A Modular Multiple Classifier System for the Detection of Intrusions”, Lecture Notes in Computer Science vol. 2709, pp. 346-355, 2003.
12. L. P. Cordella, A. Limongiello, C. Sansone, “Network Intrusion Detection by a Multi Stage Classification System”, Lecture Notes in Computer Science vol. 3077, Springer, Berlin, pp. 324-333, 2004.
13. S. Axelsson, “The Base-Rate Fallacy and the Difficulty of Intrusion Detection”, ACM Trans. on Information and System Security, vol. 3, no.3, pp. 186-205, 2000.
14. W.W. Cohen, “Fast effective rule induction”. In Proc. of the 12th International Machine Learning Conference, Morgan Kaufmann, 1995.
15. J. McHugh, “Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory”, ACM Transactions on Information and System Security, vol. 3, no. 4, pp. 262-294, 2000.
16. Y. Liu, K. Chen, X. Liao, W. Zhang, “A genetic clustering method for intrusion detection“, Pattern Recognition vol. 37, 2004.