# A Fuzzy-Genetic Approach to Network Intrusion Detection

Terrence P. Fries
Department of Computer Science
Coastal Carolina University
Conway, South Carolina 29528
1-843-349-2676
tfries@coastal.edu

## ABSTRACT
Computer networks have expanded significantly in use and numbers. This expansion makes them more vulnerable to attack by malicious agents. Many current intrusion detection systems (IDS) are unable to identify unknown or mutated attack modes or are unable to operate in a dynamic environment as is necessary with mobile networks. As a result, it is necessary to find new ways to implement and operate intrusion detection systems. Genetic-based systems offer to ability to adapt to changing environments, robustness to noise and the ability to identify unknown attack methods. This paper presents a fuzzy-genetic approach to intrusion detection that is shown to provide performance superior to other GA-based algorithms. In addition, the method demonstrates improved robustness in comparison to other GA-based techniques.

## Categories and Subject Descriptors
C.2.0 [**Computer Communications Networks**] General - *Security and Protection*; G.1.6 [**Mathematics of Computing**]: Optimization – *constrained optimization, simulated annealing*; I.2.6 [**Artificial Intelligence**]: Learning – *Parameter Learning*; I.5.2 [**Pattern Recognition**] Design Methodology – *Classifier Design and Evaluation*.

## General Terms
Algorithms, Design, Security.

## Keywords
Genetic Algorithms, Fuzzy Sets, Security, Intrusion Detection

## 1. INTRODUCTION
As computer networks expand into more areas of modern society and users require more mobility, networks become more vulnerable to intrusions and malicious attacks. Therefore, it is imperative to find improved, robust, and more reliable ways to protect this valuable resource. Intrusion detection systems (IDS) provide techniques for modeling and recognizing normal and abnormal behavior. An IDS identifies network intrusions such as anomalous behaviors, unauthorized access, malicious attacks.

There are two general categories of IDS: misuse detection and anomaly detection. Misuse detection identifies intruders that exploit weaknesses in the system and application software by comparing them with known patterns such as source address, destination address, source and destination ports, and keywords of the packet payload. These systems have a low false positive rate, however, they lack the robustness to identify attacks not explicitly coded into them. An IDS implementing anomaly detection is able to detect deviations from normal behavior including unknown or novel attacks without prior knowledge, but suffer from a higher false positive rate [3, 4, 11]. Most IDS use pattern-recognition techniques, both supervised and unsupervised, and their combinations to construct meta-classifiers used for intrusion detection. IDS methodologies include statistical models, immune system approaches, protocol verification, file and taint checking, neural networks, whitelisting, expression matching, state transition analysis, dedicated languages, genetic algorithms, and burglar alarms [11, 18]. However, most existing solutions are best applied to well-defined networks and systems. They cannot adapt to dynamic environments, complex behaviors, or unknown behaviors. As a result, they cannot identify new types of intrusions. Other methods using machine-learning algorithms can automatically be retrained, but labeled data is not readily available. New training data must be classified manually which is not practical for very large volume of network data.

There is a need for a reactive technique capable of operating in a dynamic environment and identifying previously unknown intrusions. This is difficult because of the nature of intrusion detection data. It often contains few intrusion instances (positive events) which confound the learning mechanism in its attempts to discover signatures associated with attacks. There are a large number of network transactions in which a small number of false positives can be costly to investigate. In addition, attackers frequently use new intrusion techniques or variants of existing ones.

Genetic algorithms (GA) offer the ability to overcome the shortcomings of many existing IDS techniques. GAs possess properties that make them particularly suitable for intrusion detection including robustness to noise, self-learning capability, and the ability to build initial rules without the need for a priori knowledge. Many IDS techniques experience computational limitations because it is difficult to process the large amounts of network traffic data in real time. GAs have been proven to be capable of providing near optimal solutions for NP-complete problems. Genetic algorithms have advantages over traditional techniques in intrusion detection. GAs are intrinsically parallel because they generate multiple offspring that explore the solution space in multiple directions simultaneously. Parallelism makes them well-suited where solution space is extremely large. The


*GECCO'08*, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-131-6/08/07.

adaptability of GAs allow the system to be easily retrained to evolve new rules [4, 11, 12].

This paper presents a new approach to IDS using a fuzzy-genetic approach. Section 2 discusses the test dataset commonly used to test the performance of intrusion detection systems. Section 3 presents current intrusion detection systems using genetic algorithms. The new fuzzy-genetic intrusion diction algorithm is presented in a Section 4 and the test results and comparisons with existing approaches are presented in Section 5. Section 6 provides a summary and future research direction.

## 2. IDS DATASET

All of the methods presented here were tested using the KDD Cup 1999 Dataset [10]. This dataset was derived from the 1998 DARPA Intrusion Detection Evaluation Program held by MIT Lincoln Labs. The dataset was created in a simulated in a military network environment in which a typical U.S. Air Force LAN which was subjected to simulated attacks. Raw TCP/IP dump data was gathered. Approximately 4 GB of compressed TCP dump data from 7 weeks of network traffic comprise about 5 million connection records. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted. The features exhibited in the dataset can be grouped into 3 categories: basic features of individual TCP connections, content features within a connection, and traffic features computed using a two-second time window. Each of the categories and the associated features are shown in Tables 1 – 3, respectively.

The dataset simulates 24 types of attacks in the training data and an additional 14 types in the test data. The attacks include the four most common categories of attack:

- *Denial of service* (DoS) attacks in which the attacker makes some computing or memory resource too busy to handle legitimate requests. These attacks may be initiated by flooding a system with communications, abusing legitimate resources, targeting implementation bugs, or exploiting the system's configuration.
- *User to root* (U2R) attacks in the attacker starts with access to a normal user account and exploits vulnerabilities to gain unauthorized access to the root. The most common U2R attacks cause buffer overflows.
- *Remote to user* (R2L) attacks in which the attacker sends packets to a machine, then exploits machine's vulnerabilities to gain local access as a user. This unauthorized access from a remote machine may include password guessing.
- *Probing* (PROBE) in which the attacker scans a network to gather information or find known vulnerabilities through actions such as port scanning.

**Table 1. Basic features of individual TCP connections** [10]

| ***Feature Name*** | ***Description*** |
|---|---|
| duration | length of connections (in secs.) |
| protocol_type | type of protocol |
| service | network service on destination |
| src_bytes | number of data bytes from source to destination |
| dst_bytes | number of data bytes from destination to source |
| flag | status of connection: normal or error |
| land | 1 if connection is from/to same port |
| wrong_fragment | number of "wrong" fragments |
| urgent | number of urgent packets |

**Table 2. Content features within a connection** [10]

| ***Feature Name*** | ***Description*** |
|---|---|
| hot | number of "hot" indicators |
| num_failed_logins | number of failed login attempts |
| logged_in | 1 if successfully logged in |
| num_compromised | number of "compromised" conditions |
| root_shell | 1 if root_shell is obtained |
| su_attempted | 1 if "su root" command attempted |
| num_root | number of root accesses |
| num_file_creations | number of file creation operations |
| num_shells | number of shell prompts |
| num_access_files | number of operations on access control files |
| num_outbound _cmds | number of outbound commands in an ftp session |
| is_hot_login | 1 if login belongs to hot list |
| is_guest_login | 1 if login is a guest |
| | |

**Table 3. Traffic features using a 2-second window** [10]

| ***Feature Name*** | ***Description*** |
|---|---|
| count | number of connections to same host in past 2 seconds |
| | *Note: The following features refer to these same-host connections.* |
| serror_rate | % of connections with SYN errors |
| rerror_rate | % of connections with REJ errors |
| same_srv_rate | % of connections to same service |
| diff_srv_rate | % of connections to different services |
| srv_count | number of connections to same service in past 2 seconds |
| | *Note: The following features refer to these same-service connections.* |
| srv_serror_rate | % of connections with SYN errors |
| srv_rerror_rate | % of connections with REJ errors |
| srev_diff_host_rate | % of connections to different hosts |

# 3. INTRUSION DETECTION USING GENETIC ALGORITHMS

A number of intrusion detection techniques using genetic algorithms have been proposed due to the GA's capabilities. These approaches focus on selecting key features of network traffic that reduce the complexity of identifying attackers and generating a relatively small set of rules to identify attackers. Specific techniques discussed below include genetic clustering, optimizing a set of rules to identify attackers, and creating a fuzzy inference system.

## 3.1 Genetic Clustering

The Intrusion Detection Based on Genetic Clustering (IDBGC) algorithm provides a two-stage approach that establishes clusters of network traffic features and then detects intruders by classifying traffic into one of the clusters [11].

Cluster in $D$-dimensional Euclidean space partitions a set of $n$ instances into $K$ groups based on some similarity metric. The objective is to reduce the size of data to a manageable one. One obtains a set of $K$ clusters represented by $c_1, c_2, \ldots c_K$ such that

$c_i \neq \varnothing, i = 1, \ldots, K$,

$c_i \cap c_j = \varnothing, \quad i, j = 1, \ldots, K$ and $i \neq j$ and

$$C = \bigcup_{i=1}^{K} c_i$$

The first stage of the algorithm establishes a set of original clusters using the nearest neighbor method. It groups similar instances into a cluster and then filters noisy data based on similarity or dissimilarity metrics.

In the second stage, the original clusters are combined using a GA to obtain a near optimal result. The cluster that includes the most activities is labeled as normal. To perform the genetic optimization, each chromosome contains $K$ bits where the $i$th bit represents cluster $c_i$ . If $c_i$ is selected the $i$th position in the chromosome is '1', otherwise it is '0'. The GA then creates new random clusters $c_m$ using crossover and mutation. The fitness of cluster $c_m$ based on intra-cluster distance and inter-cluster distance. Intra-cluster distance is a measure degree of nearness among clusters in $c_m$, and inter-cluster distance is a measure degree of separation among clusters in $c_m$ and outside clusters. Each item of connection data is then classified into one of the clusters, either normal or abnormal. Using the KDD Cup 1999 dataset for testing, the average detection rate was close to 60% while the average false positive rate was only 0.4%

## 3.2 Optimizing a Set of Rules

Bankovic et. al. [4] have proposed a method that creates a set of rules to identify network intrusions such as anomalous behaviors, unauthorized access, and malicious attacks. The first step uses a dimension reduction technique, principal component analysis (PCA). PCA, also known as Karhunen-Loève transform, extracts a subset of features that preserve the most relevant information by identifying a few orthogonal linear combinations of the original variables with largest variance. The 41 features in the KDD Cup 1999 dataset were reduced to only three:

*duration* number of seconds of the connection

*src_bytes* number of data bytes from source to destination

s*error_rate* percentage of connections with "SYN" errors

The second step uses a genetic algorithm to create a set of rules to classify behavior as normal or abnormal. Every selected feature represents one gene in chromosome resulting in the chromosome structure. Each chromosome represents a rule for intrusion detection in an *if-then* clause format. The conditional part of rule (antecedent) is composed of the features connected by *and* an function. The consequent of each rule is a confirmation of an intrusion. For example, the chromosome shown in Figure 1 corresponds to the rule:

***if*** (*duration* = "1" ***and*** *src_bytes* = "0"
***and*** *error_rate* = "50") ***then*** *intrusion*

Although the type of the attack is not of great importance in intrusion detection, it is important for forensics in order to recover from an attack. To accomplish this, the confirmation in the rule consequent can be replaced by an identification of the attack type. The expanded chromosome shown in Figure 2 corresponds to the rule:

***if*** (*duration* = "1" ***and*** *src_bytes* = "0"
***and*** *error_rate* = "50")
***then*** (*attack_name* = "*portsweep*")

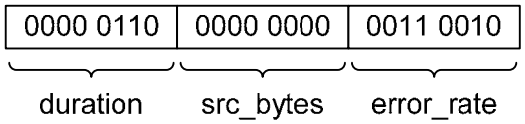


**Figure 1. Chromosome example**

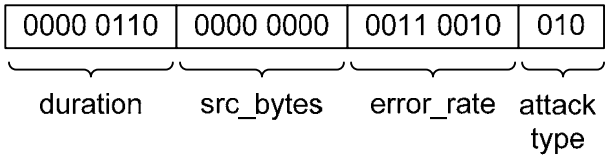


**Figure 2. Chromosome with type of attack**

An initial population of chromosomes is randomly created and then a GA is used to optimize the set of rules. To determine the fitness of each rule, the following fitness function is used:

$$fitness = \frac{\alpha}{A} - \frac{\beta}{B}$$

where $\alpha$ is the number of correctly identified attacks, $A$ is the total number of attacks in the training dataset, $\beta$ is the number of

normal connections incorrectly characterized as attacks (false positives), and $B$ is the total number of normal connections in the training dataset. The resulting fitness values are in the range [-1, 1].

When tested using the KDD Cup 1999 dataset, this method produced a 94% attack detection rate with no false positives, that is, normal connections were classified 100% correctly.

## 3.3 Fuzzy Inference Systems

The use of a GA for generating intrusion identification rules can be expanded to create a set of fuzzy rules that comprise a fuzzy inference system [1, 2, 7, 13, 16]. The resulting fuzzy inference system is capable of detecting intrusive behaviors in computer networks. The fuzzy classifier system uses fuzzy *if-then* rules that are similar to ones in the prior example except that the antecedent features are weight with fuzzy linguistic variables and the consequent classification is qualified with a certainty factor. A typical rule, $R_i$ is:

***if*** $x_1$ *is* $A_{i1}$ ***and*** … ***and*** $x_n$ *is* $A_{in}$
***then*** Class $C_i$ *with* $CF = CF_i$

where $R_i$ is the label of the $i$ th rule
$x_1, \ldots, x_n$ are the attributes/features
$A_{i1}, \ldots, A_{in}$ are antecedent fuzzy sets
$C_i$ is the consequent class
$CF_i$ is the certainty factor of the rule $R_i$

A fuzzy set, or linguistic variable, is defined by a membership function which describes the degree of membership in the set [19]. The membership function provides a mapping to any real value in the range 0 to 1, inclusive. The use of fuzzy sets accommodates values that possess a degree of inaccuracy. The fuzzy linguistic variables for the IDS rules are represented by the triangular membership functions shown in Figure 3.

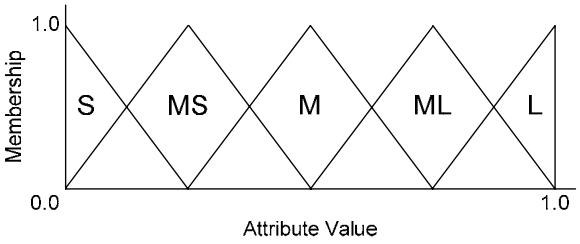


**Figure 3. Membership functions of linguistic variables (S: small, MS: medium small, M: medium, ML: medium large, L: large).**

After an initial set of fuzzy rules are randomly created, a GA is then used to optimize the set of fuzzy rules using the training set from the KDD Cup 1999 dataset. When optimized rules were tested using the KDD Cup 1999 dataset, they method produced a 99% attack detection rate with less than 4% false positives.

# 4. A FUZZY-GENETIC INTRUSION DETECTION

All of the existing methods based on genetic algorithms suffer from either a poor anomaly detection rate or a high number of false positives. This section presents a fuzzy-genetic approach for an intrusion detection system. The algorithm is comprised of two distinction parts. First, a genetic algorithm is used to establish an optimal subset of the communication features. Second, a set of fuzzy rules is optimized using a genetic algorithm.

## 4.1 Feature Subset Selection

The reduction of features is necessary to improve performance in terms of learning time, classification accuracy, and comprehensibility of the learned rules [8]. However, genetic clustering is too restrictive by reducing the features to a subset of only three. The inappropriateness of the method is evidenced by the 60% identification rate. Clearly, an algorithm is needed that is capable of producing a more robust subset of features.

Rather than use a traditional dimension reduction technique which relies on statistical analysis, a GA will be used for feature subset selection with each chromosome corresponding to a candidate feature subset. Each chromosome is encoded as a string of 0's and 1's with the number of bits equal to the total number of features. Each bit represents a particular feature. If the bit is a '1', it indicates the attribute is to be used for training, while a '0' indicates the attribute is not to be used. The GA then determines the optimal set of features to be used for training the rule set.

The GA fitness function is based on the accuracy of the hypothesized subset and the number of attributes. The accuracy is the ratio of correct identifications to the total number of communications in the test set and the cost is the number of features used in the subset. The fitness of subset $s$ is:

$$\text{fitness}(s) = w_{accuracy} * \text{accuracy}(s) + w_{cost} * \text{cost}(s)$$

where $w_{accuracy}$ and $w_{cost}$ are the respective weights of the factors.

When tested with the KDD Cup 1999 dataset, the GA produced a subset containing 8 features:

| | |
|---|---|
| *duration* | number of seconds of the connection |
| *src_bytes* | number of data bytes from source to destination |
| *num_failed_logins* | number failed login attempts |
| *root_shell* | 1 if root shell is obtained; 0 otherwise |
| *num_access_files* | number of operations on access control files |
| s*error_rate* | percent of connections with "SYN" errors |
| *same_srv_rate* | percent of connections to same service |
| *srv_count* | number of connections to same service in past 2 seconds |

## 4.2 Optimization of a Fuzzy Rule Set

While the fuzzy inference system described in the previous section only has a 4% rate of false positives, this is a substantial

number when one considers the enormous quantity of communication packets received by a network. In light of the volume of communications, this produces far too many false positives to be practical. Analysis of the results of this method reveals that the problem is related to the use of relatively narrow triangular fuzzy sets for the antecedents. This adversely affects the rules by limiting the acceptable values for a feature in a particular rule. By using predefined, inflexible fuzzy sets, the ability to match the rule antecedents is limited. This precipitates two possibilities: either the rule is limited and the antecedents are not matched in some cases, or additional rules are required for all possible ranges of the antecedent values.. To rectify these faults, this research uses trapezoidal fuzzy sets for the antecedents. Trapezoidal fuzzy sets are defined by the tuple ($a$, $b$, $c$, $d$) which describe the key points as shown in Figure 4. They have been chosen because they allow full membership over any range in the universe of discourse, thus, providing great flexibility in the weighting.

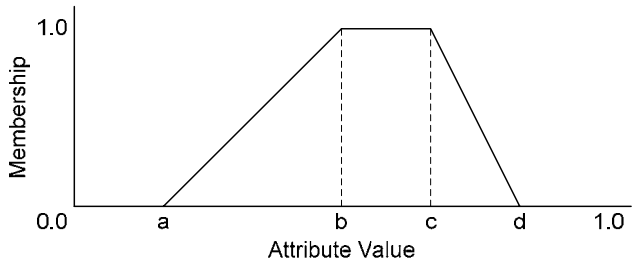


**Figure 4. Trapezoidal fuzzy set**

The fuzzy-genetic intrusion detection algorithm uses fuzzy *if-then* rules that are similar to ones already discussed with the exception that the antecedent features are weighted with trapezoidal fuzzy sets. A typical rule, $R_i$ is:

***if*** $x_1$ *is* $A_{i1}$ ***and*** … ***and*** $x_n$ *is* $A_{in}$
***then*** Class $C_i$ *with* $CF = CF_i$

where $R_i$ is the label of the $i$ th rule
$x_1, \ldots, x_n$ are the features
$A_{i1}, \ldots, A_{in}$ are trapezoidal fuzzy sets
$C_i$ is the consequent class (intrusion category)
$CF_i$ is the certainty factor of the rule $R_i$

To address the limitations regarding the use of predefined fuzzy sets, the proposed method uses a more generalized fuzzy set whose parameters ($a$, $b$, $c$, $d$) are determined by the genetic algorithm at the same time it optimizes the set of fuzzy rules. To minimize the length of the chromosomes, the fuzzy set parameters ($a$, $b$, $c$, $d$) are each limited to one of eight values in the range 0 to 7. This requires only 3 bits to encode each in the gene corresponding to a single feature. Thus, the gene for feature $x_j$ requires 12 bits as shown in Figure 5. The chromosome for rule $R_i$ requires a 12-bit gene for each feature $x_j$ as shown in Figure 6.

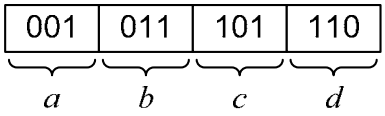


**Figure 5. Gene for feature $x_j$**

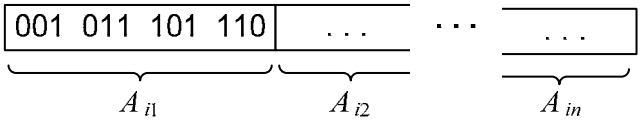


**Figure 6. Chromosome for rule $R_i$**

Once the set of fuzzy rules have been established, they are used to identify communication attempts as normal or an intruder. If the intruder is detected, the rule will provide a classification of the intrusion in the consequent $C_i$. The system is adaptable in a dynamic environment. When a new intruder is found, the system can be retrained using the genetic algorithms.

# 5. TEST RESULTS

The fuzzy-genetic intrusion detection approach was tested using the KDD Cup 1999 Dataset [10]. Feature subset selection and optimization of the fuzzy rule set, both using genetic algorithms, were performed using the 10% training subset of the KDD dataset. The training data contains approximately 500,000 connection records. Once trained, the new IDS was tested using the full 5 million connection records in the KDD dataset. The full dataset contains 14 types of intrusion attacks not present in the training data. Table 4 provides a comparison of the fuzzy-genetic IDS with the other methods described in Section 3 of this paper.

**Table 4. Performance Comparison of ID Methods**

| | Intrusion Detection Rate | False Positives |
|---|---|---|
| Genetic Clustering | 60% | 0.4% |
| Rule Optimization | 94% | 0% |
| Fuzzy Inference System | 98% | 6% |
| Fuzzy-Genetic IDS | 99.6% | 0.2% |

The proposed fuzzy-genetic intrusion detection system had the best intrusion detection rate of those tested. Only the rule optimization method had a lower rate of false positives, however, it had a lower detection rate making it less effective and the difference in false positive rates is negligible. In addition, the new system was able to correctly identify each of the 14 types of intrusions not in the training data. This demonstrates the robustness of the new IDS.

## 6. SUMMARY

This paper has presented a fuzzy approach to network intrusion detection using genetic algorithms. GAs have several advantages over traditional methods: robustness, unsupervised learning, ability to find a near optimal solution in large dimensional problem spaces, and intrinsic parallel operation. The proposed method has been shown to provide performance superior to other GA-based algorithms. In addition, the method demonstrates improved robustness in comparison to other GA-based techniques.

## 7. REFERENCES


[1] Abadeh, M. S., Habibi, J., Barzegar, Z., and Sergi, M. A Parallel Genetic Local Search Algorithm for Intrusion Detection in Computer Networks. Engineering Applications of Artificial Intelligence 20 (8), December 2007, 1058-1069.

[2] Abadeh, M. S, Habibi, J., and Lucas, C. Intrusion Detection Using a Fuzzy Genetics-Based Learning Algorithm. Journal of Network and Computer Applications 30 (1), January 2007, 414-428.

[3] Abraham, A., Jain, R., Thomas, J., and Han, S. Y. D-SCIDS: Distributed Soft Computing Intrusion Detection System. Journal of Network and Computer Applications 30 (1), January 2007, 81-98.

[4] Bankovic, Z., Stepanovic, D., Bojanic, S., and Nieto-Taladriz, O. Improving Network Security Using Genetic Algorithm Approach. Computers and Electrical Engineering 33 (5-6), September-November 2007, 438-451.

[5] Budynek, J., Bonabeau, E., and Shargel, B. Evolving Computer Intrusion Scripts for Vulnerability Assessment and Log Analysis. In Proceedings of the 2005 Genetic and Evolutionary Computation Conference (GECCO '05), 2005.

[6] Goldberg, D. E. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989.

[7] Haghighat, A. T., Esmaeili, M., Saremi, A., Mousavi, V. R. Intrusion Detection via Fuzzy-Genetic Algorithm Combination with Evolutionary Algorithms. In Proceedings of the 6th IEEE/ACIS Conference on Computer and Information Sciences), 2007.

[8] Helmer, G., Wong, J. S. K., Honavar, V., and Miller, L. Automated Discovery of Concise Predictive Rules for Intrusion Detection. Journal of System Software 60 (3), February 2002, 165-175.

[9] Holland, J. H. Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.

[10] KDD99 Cup Dataset, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 1999.

[11] Liu, Y., Chen, K., Liao, X., and Zhang, W. A Genetic Clustering Method for Intrusion Detection. Pattern Recognition 37 (5), May 2004, 927-942.

[12] Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs, 3rd Edition. New York: Springer, 1996.

[13] Ozyer, T., Alhajj, R., and Barker, K. Intrusion Detection by Integrating Boosting Genetic Fuzzy Classifier and Data Mining Criteria for Rule Pre-Screening. Journal of Network and Computer Applications 30 (1), January 2007, 99-113.

[14] Shon, T., Kovah, X., and Moon, J. Applying Genetic Algorithm for Classifying Anomalous TCP/IP Packets. Neurocomputing 69 (16-18), October 2006, 2429-2433.

[15] Shon, T., and Moon, J. A Hybrid Machine Learning Approach to Network Anomaly Detection. Information Sciences 177 (18), September 2007, 3799-3821.

[16] Toosi, A. N., and Kahani, M. A New Approach to Intrusion Detection Based on an Evolutionary Soft Computing Model Using Neuro-Fuzzy Classifiers. Computer Communications 30 (10), July 2007, 2201-2212.

[17] Tsang, C.-H., Kwong, S., and Wang, H. Genetic-Fuzzy Rule Mining Approach and Evaluation of Feature Selection Techniques for Anomaly Intrusion Detection. Pattern Recognition 40 (9), September 2007, 2373-2391.

[18] Verwoerd, T., and Hunt, R. Intrusion Detection Techniques and Approaches. Computer Communications 25 (15), September 2002, 1356-1365.

[19] Zadeh, L. Fuzzy sets. Information and Control 8(3), March 1965, 338-353.