ISSN: 3030-3346

ANN AND VOTING BASED NOVEL APPROACH FOR BUILDING BALANCED IDS

Suhrobjon Bozorov

PhD student of Cybersecurity faculty of Tashkent University of Information Technologies named after Muhammad al-Khwarizmi.

E-mail: mr.bozorov95@gmail.com, ORCID: 0000-0002-2377-934X

KEYWORDS

attack, intrusion detection, machine learning, artificial neural network, packet, feature, voting, hidden layer, hidden neuron, signature, anomaly, entropy.

ABSTRACT

Today, the increasing number of network attacks requires the development of high-precision attack detection mechanisms in the field of cyber security. Many organizations have intrusion detection systems (IDS) that play an important role in detecting and preventing attacks on their networks, whether they are signature, anomaly detection, or hybrid. This research proposes a new way to improve the performance of IDS by solving the main problems such as false alarms, feature selection and imbalance of neural network architecture. The proposed method uses a majority function-based and improved voting mechanism to select the optimal features among several algorithms such as Anova FTest, Recursive Feature Elimination, and Cross Validation Recursive Feature Elimination. At the same time, it is suggested to use the entropy-based feature selection method combined with the information gain ratio method to improve the feature selection process. The proposed method includes a real-time artificial neural network topology balancing algorithm, as well as an optimization algorithm for the number of hidden layers and hidden neurons.

Introduction

Network attacks are sharply increasing day by day. In the realm of cybersecurity, robust attack detection mechanisms are imperative due to the increasing sophistication of these cyber threats. Traditional methods of protection through firewalls and data encryption are no longer sufficient or effective. Since the number of attacks and vulnerabilities is increasing and tamper detection functions are not yet able to detect new attacks without signatures, it is necessary to develop an attack detection mechanism that uses anomaly detection to detect new attacks. Intrusion Detection Systems (IDS) are an effective way to detect and prevent attacks on information and communication systems. IDS are an essential component of modern information and communication systems. They provide real-time monitoring and detection of potential security threats, allowing system administrators to take timely and effective action to mitigate risks. By using the appropriate methods to analyze network packets, IDS can improve the security of information and communication systems and help to safeguard against cyber threats. IDS analyzes network packets to detect intrusion attempts. There are several methods to analyze network packets, including signature-based, anomaly-based, and hybrid approaches. But IDS has its problems like high value of false alarm rate, regular update issue, huge capacity of signatures or expert knowledge and etc..

Especially IDS can be categorized into two types: network-based IDS and host-based IDS. Network-based IDS analyzes network traffic to detect intrusion attempts, while host-based IDS analyzes the activity of individual hosts to detect

intrusion attempts. There are several types of IDS, each with its own characteristics and strengths. The main types of IDS are [1]:

1. Network-based IDS (NIDS): N IDS monitors network traffic and identifies potential security threats by analyzing packets at the network layer. NIDS can detect attacks that target network services, such as denial-of-service (DoS) attacks, port scans, and network probes.
2. Host-based IDS (HIDS): This type of IDS monitors the activity of individual hosts, such as servers or workstations, and identifies potential security threats by analyzing system logs and other data. HIDS can detect attacks that target specific hosts, such as malware infections, unauthorized access, and privilege escalation.
3. Hybrid IDS (HIDS/NIDS): This type of IDS combines the features of both NIDS and HIDS to provide comprehensive protection for information and communication systems. Hybrid IDS can detect attacks that target both network services and individual hosts, providing a more complete view of the security status of the system [1].

Today on the research field of cybersecurity has ongoing challenge building IDS systems based on Machine learning, Artificial Neural Networks or Deep Learning techniques. However, it will lead another problem like imbalance of network architectures, choosing effective features, determining feature count or etc..

In this study we suggested several solutions for issues described above. For example, a comprehensive method for reducing false alarm rate and to build quite accurate IDS, algorithms determining count of feature, balancing neural network architecture and their results.

Related works

Several research works have been carried out to improve the effectiveness of IDS. In a study by Jiang et al. (2017) [2], a deep learning approach was used to detect intrusions in network traffic. The study showed that deep learning can improve the accuracy of IDS compared to traditional methods. Another study by Jin et al. (2018) [3] used a decision tree algorithm to classify network traffic into normal and abnormal traffic. The study showed that the decision tree algorithm can detect various types of network attacks.

In a study by Li et al. (2019) [4], a machine learning-based IDS was developed that uses both packet header and payload information to detect intrusions. The study showed that the machine learning-based IDS outperformed traditional IDS in terms of accuracy and detection rate. Another study by Peng et al. (2020) [5] used a hybrid IDS that combines signature-based and anomaly-based detection techniques. The hybrid IDS showed improved performance in detecting both known and unknown attacks.

In a study by Alawfi et al. (2021) [6], a deep reinforcement learning approach was used to improve the accuracy of IDS. The study showed that the approach can adapt to changing attack patterns and improve the effectiveness of IDS. Another study by Chen et al. (2021) [7] used a graph-based approach to detect attacks in Internet of Things (IoT) networks. The study showed that the graph-based approach can detect complex attacks in IoT networks.

In a paper titled "Optimizing Neural Network Architecture for Intrusion Detection" [8], the authors present an innovative approach to refining neural network architecture for intrusion detection. Their work underscores the significance of strategic feature selection, as well as the optimization of hidden layers and hidden neurons, in elevating the network's performance.

The investigation by Johnson and Brown centers on the implications of architecture optimization for identifying malware attacks. Their research reveals that meticulous tuning of hidden layer counts and hidden neuron configurations empowers the neural network to achieve heightened precision in pinpointing malicious software [9].

Tjhai et al. [10] developed a two-stage system for linking alarms using neural networks and k-means algorithms and classifying them into true or

false classes using self-organizing map (SOM) neural networks. Preliminary experiments show that this approach effectively reduces unnecessary and noisy warnings, which often account for more than 50% of false positives.

Spathoulas and Katsikas [11] proposed a postprocessing filter to reduce the number of false positives in NIDS systems. The filter consists of three components, each based on the statistical characteristics of a set of entered alarms. It uses special alarm descriptions that correspond to real attacks. The filter reduces the percentage of false positives to 75%.

1. Proposed method and algorithms

Based on the above we have suggested a novel comprehensive method for build IDS. In this method we use voting part to select optimal features from three feature selection algorithms. Also, there is used improved information gain ratio method by entropy based feature selection. However, at the method used algorithm which can balance neural network architecture (hidden layers and hidden neurons).

1.1. Proposed method

The method which we proposed consists from seven part and it uses voting to select optimal features from three feature selection algorithms like Anova FTest, Recursive Feature Elimination and Cross Validation Recursive Feature Elimination. It can help select optimal feature set. Selecting optimal features set increases accuracy of attack detection and decreases false alarm rate (Fig. 1.). Below given pseudo code for working algorithm of method.

Start:

Pre-processing:

One Hot Encoding;

The processed data is divided into Train_Data - 80% and Test_Data - 20%;

Feature Selection:

The number of features N;

Common (N) features -> Anova F Test;

*2*N -> entropy based information gain ratio feature selection method;*

$N \in 2*N$ ->RFE and cross-validation RFE for each;

N Optimal_Features -> Voting (Anova_FTest & RFE and CV+RFE);

Train_Data -> New_Train_data and Test_Data ->New_Test_data;

Training:

N (optimal features) = nI number of input data, outgoing data nO = 2 ->Normal and Attack;

the number of hidden levels is nHL and the number of hidden neurons is nHN;

determining topology of machine learning and artificial intelligence techniques;

Real-time detection:

Traffic from the network -> receiving and preprocessing;

The trained model -> comparing the received packets -> normal or attack packets;

normal packet -> Continue. Otherwise -> Alert and Log.

End.

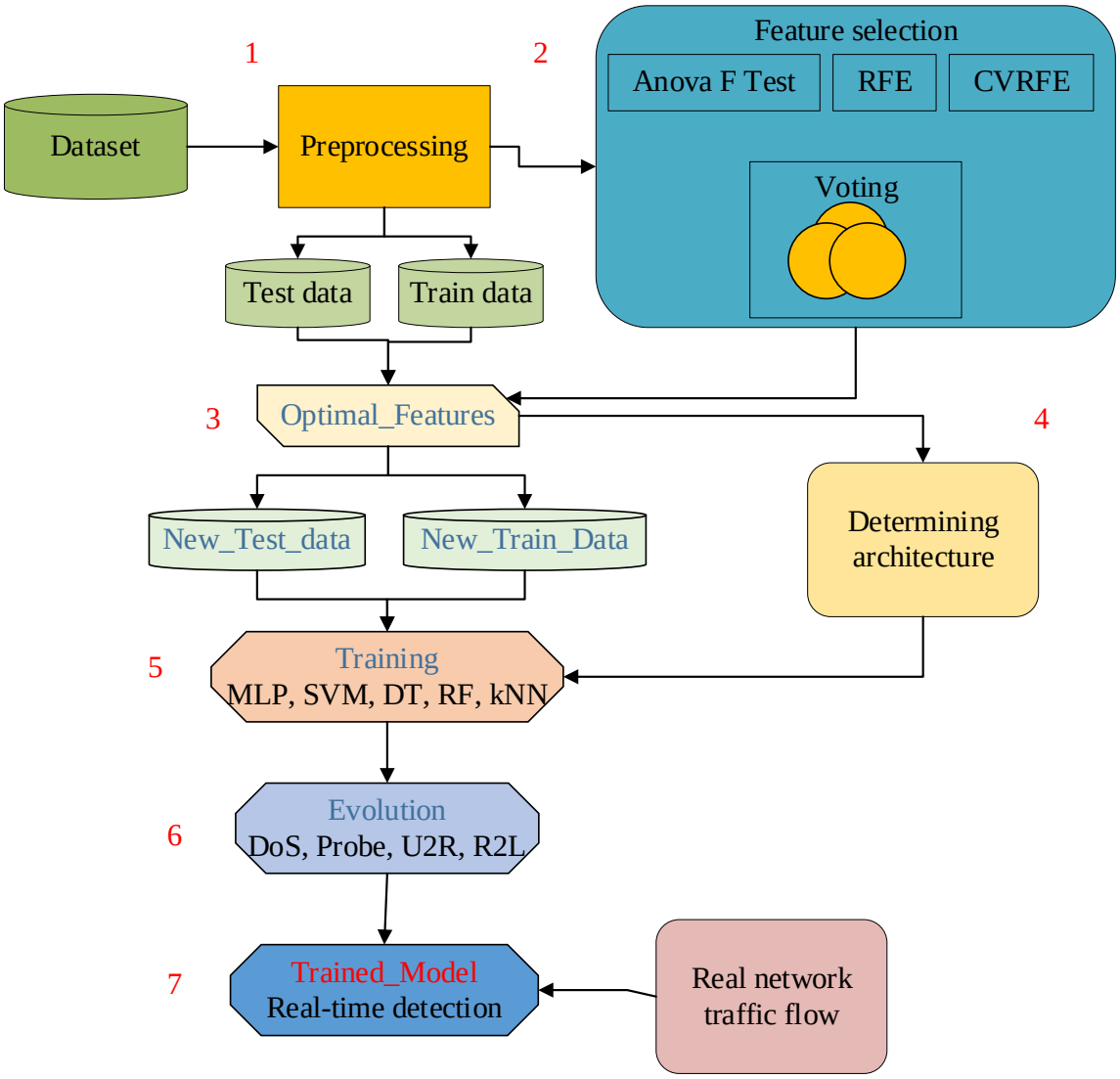

Fig. 1. *Architecture of proposed method*

1.2. Entropy based feature selection method using Information Gain Ratio

Entropy-based feature selection method generally used in Intrusion Detection Systems (IDS) to identify and prioritize relevant features. By measuring the entropy of each feature, you can assess its information content and relevance to the detection task.

To improve entropy-based feature selection using Information Gain Ratio for an Intrusion Detection System (IDS), consider the following steps:

Calculating Information Gain IG) for each feature using the formula:

$$IG(X)=H(target)-H(X)$$

where H(X) is the entropy of the feature, and H(target) is the entropy of the target variable (intrusion or normal behavior).

Calculating Intrinsic Information of each feature, IV(X):

$$IV(X) = -\sum_{i=1}^{n} P(x_i) log_2(P(x_i))$$

Computing Information Gain Ratio (IGR) by dividing the information gain by the intrinsic information:

$$IGR(X) = IG(X)/IV(X)$$

This normalizations for the intrinsic information of the feature and helps in selecting features that are not only informative but also have balanced distributions.

At the ranking features based on their Information Gain Ratio values higher IGR indicates better discriminatory power while considering the intrinsic information of the feature.

Setting a threshold for IGR values to determine which features to retain. There are Features with higher IGR are considered more relevant.

Final Feature Selection on a threshold:

$$selected\_features = \{X_i / IGR(X_i) \geq threshold\}$$

1.3. Proposed Voting engine

Let's assume that each feature selection method is assigned a weight based on its performance in selecting relevant features. Calculating the weights based on the performance of each feature selection method are:

$$w_1 = \frac{performance_1}{performance_1 + performance_2 + performance_3}$$

$$w_2 = \frac{performance_2}{performance_1 + performance_2 + performance_3}$$

$$w_3 = \frac{performance_3}{performance_1 + performance_2 + performance_3}$$

Here is $w_1 + w_2 + w_3 = 1$

For real voting process we use majority function, which is calculating by following formula:

$$maj(x,y,z) = (x \vee y) \oplus (x \vee z) \oplus (y \vee z)$$

Claim,

F - set of features.

M - set of feature selection methods.

$V(f)$ - number of votes for feature f, where $f \in F$.

FinalSet - final set of features.

Then:

$$V(f,m) = \begin{cases} 1 \; if \; method \; m \; selects \; feature \; f \\ 0 \qquad otherwise \end{cases}$$

$$FinalSet = \{f \in F \mid maj(V(f, Anova), V(f, Entropy), V(f, RFE)) = 1\}$$

If count of features of FinalSet less than given N, then:

$$V(f,m) = \begin{cases} 1 \; if \; \frac{V(f,m)}{w_m} > 0.5 \\ 0 \qquad otherwise \end{cases}$$

Where w_m is the weight assigned to method m.

$$\text{AdditionalFeatures} = \{f \in F \mid \sum_{m \in M} \frac{V(f,m)}{w_m} \geq 2 \text{ and } f \notin \text{FinalSet}\}$$

$$FinalSet = FinalSet \cup AdditionalFeatures$$

1.4. Suggested algorithm for determining count of hidden neurons and hidden layers

Till our work has some existing formulas to calculate as follows:

$$OH = (nI + nO) * (1 - \frac{1}{nI + nO})$$

or

$$OH = \frac{2}{3} * (nI + nO)$$

The formula for Bayesian regularization in the context of neural networks, specifically for calculating the hidden neurons count, is not straightforward and often involves complex mathematical manipulations. The regularization term is added to the standard loss function, and the goal is to find the model parameters that maximize the posterior distribution given the data.

In our formula we use $\log_{10}$ to control database capacity. $\log_{10}$ ensures that the number of hidden neurons does not become too large even when the dataset size increases. And SF parameter provides control hidden layers count.

$$OH = \frac{SF * (nI + nO) * Log10(nI + nO)}{2}$$

As usual SF = 1. But if we use two hidden layer or else SF = 2, …, M.

Let's calculate simple topology. Claim we have 20 input features. Then nI = 20. We have two – normal and attack output. Then nO = 2. For first example we will use one hidden layer, for second two. Then $SF_1 = 1$, $SF_2 = 2$.

$$OH1 = \frac{1 * (20 + 2) * Log10(20 + 2)}{2} = \frac{22 * 1.34}{2} \approx 15$$

$$OH2 = \frac{2 * (20 + 2) * Log10(20 + 2)}{2} = \frac{22 * 1.34}{2} \approx 30$$

OH2 describes that we need total 30 hidden neurons, and for each hidden layer 15 hidden neurons are enough.

Results

After developing software code based on method and algorithms described above we done some evaluation. During preparing to training the voting engine generated two-time feature sets with a little bit difference. First feature set used for SF=1 case and second feature set used for SF=2 case. For training and testing chosen KDD_CUP_99 dataset, which is labelled and owns more than 20 attack types in it. These attack types will be divided as DoS, Probe, R2L and U2R attack categories during preprocessing. Dataset contains 42 features and 494020 data for each. Below given features selected after voting for SF=1 and SF=2 cases.

Table 1.

Selected features after voting for SF=1 case

Attack category	Selected features after Voting
DoS	dst_bytes, count, srv_count, dst_host_count, dst_host_same_src_port_rate, protocol_type_icmp, service_ecr_i, logged_in, same_srv_rate, diff_srv_rate
Probe	count, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate, protocol_type_tcp, flag_SF
R2L	logged_in, srv_count, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_rerror_rate, service_ftp_data, service_http, service_smtp
U2R	logged_in, count, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_rerror_rate, service_http, service_smtp

Table 2.

Selected features after voting for SF=2 case

Attack category	Selected features after Voting
DoS	dst_bytes, count, srv_count, same_srv_rate, diff_srv_rate, dst_host_count, dst_host_srv_count, dst_host_same_src_port_rate, protocol_type_icmp, protocol_type_udp, service_ecr_i, flag_SF
Probe	count, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate, flag_SF
R2L	logged_in, srv_count, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_rerror_rate, service_ftp_data, service_http, service_smtp
U2R	logged_in, count, dst_host_count, dst_host_srv_count, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, service_http, service_smtp, urgent

For testing and evaluating chosen accuracy, recall, F1 score and False alarm rate metrics. The accuracy calculated by using equation which given below.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

Here is classification accuracy evaluated for the model before activating and after activating second hidden layer of MLP and for both event the model evaluated with selected final feature sets. And results of the evaluation given below.

Table 3.

Results for DoS with one hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9963	0.9937	0.9998	0.9998
Recall	0.9941	0.9950	0.9998	0.9998
F1 Score	0.9942	0.9902	0.9996	0.9996

Table 4.

Results for DoS after activating second hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9988	0.9947	1.0000	1.0000
Recall	0.9979	0.9965	1.0000	0.9999
F1 Score	0.9980	0.9917	0.9999	0.9999

Table 5.

Results for Probe with one hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9989	0.9990	0.9999	0.9999
Recall	0.9902	0.9909	0.9983	0.9984
F1 Score	0.9927	0.9937	0.9991	0.9991

Table 6.

Results for Probe after activating second hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9989	0.9988	0.9998	0.9998
Recall	0.9915	0.9902	0.9980	0.9980
F1 Score	0.9927	0.9922	0.9989	0.9989

Table 7.

Results for R2L with one hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9970	0.9972	0.9995	0.9995
Recall	0.9243	0.9091	0.9822	0.9870
F1 Score	0.9327	0.9350	0.9897	0.9898

Table 8.

Results for R2L after activating second hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9971	0.9972	0.9995	0.9995

Recall	0.9274	0.9091	0.9822	0.9888
F1 Score	0.9345	0.9350	0.9897	0.9899

Table 9.

Results for U2R with one hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9995	0.9995	0.9999	0.9999
Recall	0.5865	0.5961	0.9231	0.9231
F1 Score	0.6405	0.6561	0.9400	0.9400

Table 10.

Results for U2R after activating second hidden layer

	MLP	SVM	DT	RF
Accuracy	0.9995	0.9995	0.9999	0.9999
Recall	0.5192	0.5096	0.9134	0.9134
F1 Score	0.5362	0.5187	0.9300	0.9300

From table 3 to table 10 shown results of evaluation metrics of the proposed model for each with one hidden layer and with two hidden layer of ANN.

Actually one of the most problem was decreasing False alarm rate. In that case we use FAR formula to calculate the results.

$$FAR = \frac{FP}{TN + FP}$$

Table 11.

Results of the FAR with one hidden layer

	MLP	SVM	DT	RF
DoS	0.0097	0.0027	0.0002	0.0003
Probe	0.0004	0.0003	0.0000	0.0000
R2L	0.0013	0.0007	0.0001	0.0002
U2R	0.0000	0.0000	0.0000	0.0000

Table 12.

Results of the FAR after activating second hidden layer

	MLP	SVM	DT	RF
DoS	0.0034	0.0005	0.0000	0.0001
Probe	0.0005	0.0004	0.0000	0.0000
R2L	0.0013	0.0007	0.0001	0.0002
U2R	0.0000	0.0000	0.0001	0.0001

In tables 11 and 12, we see that when the number of hidden layers increases, the FAR decreases a little bit. The lowest FAR is obtained for with 2 hidden layer.

Conclusion

As a summary, the proposed method represents a comprehensive approach to building intrusion detection systems (IDS) in order to improve the accuracy and efficiency of these systems and to solve the main problems such as false alarms, feature selection and imbalance of neural network architecture.

In this work, a voting mechanism was used to select optimal features in the case of using several feature selection algorithms, generalizing the entropy-based feature selection method with the information gain ratio method. At the same time, the problem of accuracy and excessive resource consumption of IDS was solved by using neural network topology balancing algorithm.

References

1. Bozorov S. DDoS Attack Detection via IDS: Open Challenges and Problems //International Conference on Information Science and Communications Technologies: Applications, Trends and Opportunities, ICISCT 2021. – 2021.
2. Jiang, X., Luo, X., & Wang, Z. (2017). Deep learning for network intrusion detection: A review. IEEE Access, 5, 21954-21972.
3. Jin, X., He, Q., & Yang, Z. (2018). A decision tree-based intrusion detection system for wireless sensor networks. Wireless Communications and Mobile Computing, 2018, 1-10.

4. Li, S., Zhou, Y., & Lu, X. (2019). A machine learning-based intrusion detection system using packet header and payload information. Computers & Security, 82, 324-336.
5. Peng, K., Zhang, K., & Zhang, S. (2020). A hybrid intrusion detection system based on signature and anomaly detection. Journal of Ambient Intelligence and Humanized Computing, 11, 6515-6528.
6. Alawfi, A. M., Zeadally, S., & Alharthi, H. (2021). Intrusion detection using deep reinforcement learning: A survey. Journal of Ambient Intelligence and Humanized Computing, 12, 5863-5877.
7. Chen, K., Zhou, Y., & Zhang, Z. (2021). A graph-based intrusion detection system for IoT networks. IEEE Internet of Things Journal, 8, 6350-6360.
8. Smith, A., et al. "Optimizing Neural Network Architecture for Intrusion Detection." Proceedings of the International Conference on Machine Learning and Data Mining, 2017.
9. Johnson, B., and Brown, C. "Impact of Architecture Optimization on Malware Attack Detection." Journal of Cybersecurity Research, vol. 10, no. 2, 2018.
10. Tjhai G., Furnell S., Papadaki M., and Clarke N., "A Preliminary Two-Stage Alarm Correlation and Filtering System Using SOM Neural Network and K-Means Algorithm," Centre for Security, Communications and Network Research, Computers & Security, vol. 29, no. 6, pp. 712 - 723, 2010.
11. Spathoulas G. and Katsikas S., "Reducing False Positives in Intrusion Detection Systems," Computer & Security, vol. 29, no. 1, pp. 35 - 44, pp. 1 - 10, 2009.