INTRUSION DETECTION USING A MIXED FEATURES FUZZY CLUSTERING ALGORITHM

Sarab M. Hameed, Sumaia Saad Sulaiman
Sarab_majeed@yahoo.com, sumasaad@yahoo.com
Department of Computer Science, College of Science, University of Baghdad. Baghdad-Iraq

Abstract

Proliferation of network systems and growing usage of Internet make network security issue to be more important. Intrusion detection is an important factor in keeping network secure. The main aim of intrusion detection is to classify behavior of a system into normal and intrusive behaviors. However, the normal and the attack behaviors in networks are hard to predict as the boundaries between them cannot be well distinct. This paper presents an algorithm for intrusion detection that combines both fuzzy C Means (FCM) and FCM for symbolic features algorithms in one. Experimental results on the Knowledge Discovery and Data Mining Cup 1999 (KDD cup 99) intrusion detection dataset show that the average detection rate of this algorithm is 99%. The results indicate that the proposed algorithm is able to distinguish between normal and attack behaviors with high detection rate.

Keywords: Fuzzy Clustering, Fuzzy C mean, Intrusion Detection, Mixed Features, Symbolic data.

كشف التطفل عن طريق استخدام خوارزمية التجمع الضبابية ذات الميزات المختلطة

سراب مجيد حميد، سمية سعد سليمان
sumasaad@yahoo.com ، Sarab_majeed@yahoo.com
قسم علوم الحاسبات ، كلية العلوم، جامعة بغداد.بغداد-العراق

الخلاصة

انتشار نظم الشبكة والاستعمال المتزايد للإنترنت يجعل قضية الأمن أكثر أهمية. كشف المتطفل هو عامـل مهما في حفظ الشبكة آمنة. الهدف الرئيسي لكشف المتطفلين هو تصنيف سلوك النظـام إلـى سـلوك طبيعـي وهجومي. ومع ذلك، فإن السلوك الطبيعي والهجومي في الشبكات يصعب التنبؤ بها والحدود بينهما لا يمكـن أن تكون متميزة أيضا. يقدم هذا البحث خوارزمية جديدة تجمع بين خوارزمية FCM, FCM ذو الميزات الرمزيـة لكشف التطفل. أظهرت النتائج التجريبية على99 KDD cup بان معدل اكتشاف الخوارزمية المقترحة هـو ٩٩ ٪. تشير النتائج أن الخوارزمية المقترحة قادرة على التمييز بين السلوك الطبيعي والهجومي بمعـدل اكتشـاف عالي.

1. Introduction

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network [1]. In general, intrusion detection systems (IDSs) fall into two categories according to the detection methods they employ, namely misuse detection and anomaly detection. Misuse detection identifies intrusions by matching observed data with pre-defined descriptions of intrusive behavior. Therefore, well-known intrusions can be detected efficiently with a very low false alarm rate. For this reason, the approach is widely adopted in the majority of commercial systems. However, intrusions are usually polymorph, and evolve continuously. Misuse detection will fail easily when facing unknown intrusions [2]. One way to address this problem is to regularly update the knowledge base, either manually which is time consuming and laborious, or automatically with the help of supervised learning algorithms. Unfortunately, datasets for this purpose are usually expensive to prepare, as they require labeling of each instance in the dataset as normal or a type of intrusion. Another way to address this problem is to follow the anomaly detection model proposed by Denning [3]. Anomaly detection is orthogonal to misuse detection. It hypothesizes that abnormal behavior is rare and different from normal behavior. Hence, it builds models for normal behavior and detects anomaly in observed data by noticing deviations from these models. There are two types of anomaly detection [2]. The first is static anomaly detection, which assumes that the behavior of monitored targets never changes, such as system call sequences of an Apache service. The second type is dynamic anomaly detection. It extracts patterns from behavioral habits of end users, or usage history of networks hosts. Sometimes these patterns are called profiles. Clearly, anomaly detection has the capability of detecting new types of intrusions, and only requires normal data when building profiles. However, its major difficulty lies in discovering boundaries between normal and abnormal behaviors, due to the deficiency of abnormal samples in the training phase. Another difficulty is to adapt to constantly changing normal behavior, especially for dynamic anomaly detection [2]. Q. Wang and V. Megalooikonomou [4] present a clustering algorithm which uses fuzzy connectedness as the similarity metric for intrusion detection. This algorithm starts with a single or a few seed points in each cluster, all the data points are dynamically assigned to the cluster that has the highest fuzzy connectedness value. The value of the fuzzy connectedness is calculated using both the Euclidean distance and the statistical properties of clusters. Experimental results showed the stability of efficiency and accuracy of the algorithm. K. Makkithaya et al. [5] present intrusion detection system based on C-fuzzy decision tree. The tree grows gradually by using fuzzy C-means clustering (FCM) algorithm to split the patterns in a selected node with the maximum heterogeneity into C corresponding children nodes. Also they used a modified fuzzy C-means algorithm with controllable membership ratio through an extended distance measure to include an additional higher order term. The results of the algorithm enhanced the performance of IDSs and emphasized the importance of modified tree in developing improved IDSs. M. M. T. Jawhar and M. Mehrotra [6] present an intrusion detection model based on hybrid fuzzy logic and neural network The first stage of this model is applying FCM to classify the data into two clusters normal and attack. The second stage uses Multi-layer feed forward networks (MLP) for classification of attacks. The number of hidden layers, and the number of nodes in the hidden layers, was determined based on the process of trial and error. This model has the ability to recognize an attack, to differentiate one attack from another and to detect new attacks with high detection rate and low false negative. K. bharti et al. [7] have used fuzzy k mean clustering algorithm and Random Tree classification techniques for finally assigning a cluster to a particular class. From experimental results for a two class datasets it is observed that filtered fuzzy random forest dataset gives the better results.

Fuzzy systems have several important characteristics that suit intrusion detection very well namely [8]:

- Fuzzy systems can readily combine inputs from widely varying sources.

- Many types of intrusions cannot be crisply defined (e.g. at what threshold should an alarm be set?)
- The degree of alert that can occur with intrusions is often fuzzy.

Accordingly, this paper proposes an algorithm for intrusion detection that combines both FCM and FCM clustering for symbolic features in one algorithm to distinguish between normal and attack behaviors. The rest of the paper is organized as follows: Section 2 gives some background on FCM clustering algorithm and FCM for symbolic data. Section 3 describes the proposed intrusion detection algorithm. Section 4 presents the description of Knowledge Discovery and Data Mining Cup 1999 (KDD cup 99) on which the proposed algorithm experiments are conducted. Section 5 illustrates the results obtained from implementing and testing the proposed algorithm. Finally, conclusions are presented in section 6.

2. Fuzzy Clustering Algorithm

This section presents a brief background on FCM and FCM clustering for symbolic features.

Cluster analysis groups data objects only on the bases of information found in the data that describes the objects and their relationships. The goal is that the objects within a group should be similar (or related) to one another and different from (or unrelated to) the objects in other groups. The greater the similarity (or homogeneity) within a group and the greater the difference between groups, the better or more distinct the clustering is [9].

The most known method of fuzzy clustering is the FCM. FCM is a method of clustering which allows one piece of data to belong to two or more clusters. This method was proposed by Dunn in 1973 [10] and then generalized by Bezdek in 1981[11]. It is based on the minimization of the objective function formulated in equation (1) [11].

$$J_m = \sum_{k=1}^{n}\sum_{i=1}^{c} \mu_{ik}{}^{m} d^2(x_k, v_i) \qquad (1)$$

Subject to constraints

$$\sum_{i=1}^{c} \mu_{ik} > 0, \qquad \text{for all } k = 1,2,...,n \qquad (2)$$

$$\sum_{k=1}^{n} \mu_{ik} = 1, \qquad \text{for all } i = 1,2,...,c \qquad (3)$$

Where

n is the number of patterns in the data set.

c is the number of clusters.

m is any real number in the range $1 \le m \prec \infty$.

μ_{ik} is the degree of membership of the pattern x_k in cluster i.

x_k is the k^{th} pattern.

v_i is the center (prototype) of the i^{th} cluster.

and

$d^2(x_k, v_i)$ is dissimilarity measure between the pattern x_k and the center v_i of specific cluster i. Fuzzy partitioning is carried out through an iterative optimization of the objective in equation 1, with the update of membership μ_{ik} and the cluster centers v_i by equations 4 and 5 respectively [11].

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{d^2(x_k, v_i)}{d^2(x_k, v_j)}\right)^{\frac{2}{m-1}}} \qquad (4)$$

$$v_i = \frac{\sum_{k=1}^{n} \mu_{ik}{}^{m} x_k}{\sum_{k=1}^{n} \mu_{ik}{}^{m}} \qquad (5)$$

The iterations will stop when $\left|\mu_{ik}{}^{t+1} - \mu_{ik}{}^{t}\right| \le \varepsilon$, where ε is a termination criterion between 0 and 1 and t is the iteration step. This procedure converges to a local minimum or a saddle point of J_m.

In 1998, El-Sonbaty and Ismail [12] suggested a fuzzy symbolic c-means algorithm. A cluster center is assumed to be formed as a group of features and each feature is composed of several events. Let $v_{jp|i}$ be the pth event of feature j in cluster i and let $e_{jp|i}$ be the membership degree of association of the p^{th} event $v_{jp|i}$ to the feature j in cluster i. Thus, the jth feature of the i^{th} cluster center v_{ij} can be presented as equation (6) [13].

$$v_{ij} = [(v_{j1|i}, e_{j1|i}), \ldots, (v_{jp|i}, e_{jp|i})] \qquad (6)$$

Where

$$e_{jp|i} = \frac{\sum_{k=1}^{n} (\mu_{ik})^m \theta}{\sum_{k=1}^{n} (\mu_{ik})^m} \qquad (7)$$

Where $\theta \in \{0,1\}$ and $\theta = 1$ if the jth feature of the kth pattern x_k consists of the pth event, otherwise $\theta = 0$. μ is the membership degree μ_{ik} of each pattern *k* in cluster *i* and *the* member degree is updated according to equation (4).

3. The Proposed Algorithm

Symbolic features appear commonly in the network traffic data stream.

In order to manipulate with network traffic data stream that contains symbolic features in addition to the numeric features, the proposed algorithm combines both the conventional FCM algorithm that partitions only the numeric features patterns and FCM that partitions symbolic features in one algorithm.

The objective function of the FCM with mixed features is defined as equation (1) with adjusting the distance according to the proposed equation (8).

$$d^2(x_k, v_i) = \sum_{s=1}^{\mathrm{sf}} d_s(x_k, v_i) + d_N(x_k, v_i) \qquad (8)$$

Where

sf is the number of symbolic features.

$d_s(x_k, v_i)$ represents the similarity metric for symbolic features in the network traffic data stream and it is computed with formula (9).

$$d_s(x_k, v_i) = \sum_{p=1}^{en} (difference_{ik} \times e_i^p) \qquad (9)$$

Where

en is the number of events for feature s.

$$difference_{ik} \begin{cases} 0 & if\ x_k = v_i^p \\ 1 & otherwise \end{cases} \qquad (10)$$

$d_N(x_k, v_i)$ is the Euclidean distance that is used to measure the similarity among numeric features in the network traffic data stream.

The following steps describe the proposed algorithm in details.

Setp1: Initialization

Number of clusters c=2.
Number of patterns in the dataset (*n*).
Number of features (*nf*).
Fuzziness control (m), with m>1
Iteration number t=0
Stopping criterion $\varepsilon = 0.001$
and Initialize fuzzy membership μ_{ik} which satisfies equations (2) and (3).

Step2: Calculation

For $1 \le k \le n$
For $1 \le i \le c$

- Compute the cluster center $v_i^{(t)}$.
  If the feature is numeric then compute the *i*th cluster center using equation (5).
  Else compute the center of cluster *i* using equation (6)
- Computer the distance $d^2(x_k, v_i)$ between the *k*th pattern and the *i*th cluster center using equation (8).
- Update the fuzzy membership degree $\mu_{ik}^{(t)}$ using equation (4).

Step3: Stopping criterion

If $\max_{ik} \left\{ \left| \mu_{ik}^{t+1} - \mu_{ik}^{t} \right| \right\} \prec \varepsilon$ then STOP
Else
Increment iteration number $t = t + 1$ and GO TO step 2

4. Dataset Description

The experiments were carried out on a real data stream called intrusion detection dataset which has been used in the KDD cup 99 [14].

KDD cup 99 dataset was derived in 1999 from the DARPA98 network traffic dataset by assembling individual TCP packets into TCP connections. It was the benchmark dataset used in the International KDD tools competition, and also the most popular dataset that has ever been used in the intrusion detection field [14]. The KDD cup 99 dataset includes a set of 41 features derived for each connection and a label which specifies the status of connection records as either normal or specific attack type. Table 1 clarifies the features data types of KDD Cup 99 intrusion detection dataset.

Table 1: List of Features and data types

No.	Feature	Type
1	duration	Continuous
2	protocol_type	Symbolic
3	Service	Symbolic
4	Flag	Symbolic
5	src_bytes	Continuous
6	dst_bytes	Continuous
7	Land	Symbolic
8	wrong_fragment	Continuous
9	Urgent	continuous
10	Hot	Continuous
11	num_failed_logins	Continuous
12	logged_in	Symbolic
13	num_compromised	Continuous
14	root_shell	Continuous
15	su_attempted	Continuous
16	num_root	Continuous
17	num_file_creations	Continuous
18	num_shells	Continuous
19	num_access_files	Continuous
20	num_outbound_cmds	Continuous
21	is_host_login	Symbolic
22	is_guest_login	Symbolic
23	Count	Continuous
24	srv_count	Continuous
25	serror_rate	Continuous
26	srv_serror_rate	Continuous
27	rerror_rate	Continuous
28	srv_rerror_rate	Continuous
29	same_srv_rate	Continuous
30	diff_srv_rate	Continuous
31	srv_diff_host_rate	Continuous
32	dst_host_count	Continuous
33	dst_host_srv_count	Continuous
34	dst_host_same_srv_rate	Continuous
35	dst_host_diff_srv_rate	Continuous
36	dst_host_same_src_port_rate	Continuous
37	dst_host_srv_diff_host_rate	Continuous
38	dst_host_serror_rate	Continuous
39	dst_host_srv_serror_rate	Continuous
40	dst_host_rerror_rate	Continuous
41	dst_host_srv_rerror_rate	Continuous

The attacks fall in one of the following four categories:

1. Denial of Service (DoS): Attacker tries to prevent legitimate users from using a service.
2. Remote to Local (R2L): Attacker does not have an account on the victim machine, hence tries to gain access.
3. User to Root (U2R): Attacker has local access to the victim machine and tries to gain super user privileges.
4. Probe: Attacker tries to gain information about the target host.

There are multiple attack types for each category as shown in table 2 [14].

Table 2: Attack types

Category	Type
DoS	smurf, neptune, back, teardrop, pod, land
Probe	satan, ipsweep, portsweep, nmap
R2L	warezclient, guess_passwd, warezmaster, ftp_write, multihop, phf, spy, imap
U2R	buffer_overflow,rootkit, loadmodule, perl

The KDD dataset consists of three components: "Whole KDD", "Corrected KDD" and "10% KDD" as illustrated in table 3 [14].

Table 3: Number of Patterns in KDD Cup 99 Datasets

KDD dataset	DoS	Probe	R2L	U2R	Normal
Whole	3883370	41102	1126	52	972780
Corrected	229853	4166	16347	70	60593
10%	391458	4107	1126	52	97277

5. Experimental Results

The network traffic data from the KDD Cup 99 dataset was used to evaluate the proposed algorithm. The following subsections will illustrate the KDD Cup 99 dataset preprocessing and features selection, the performance evaluation criteria and finally, the results of the proposed algorithm.

5.1 KDD Cup 99 Preparation and Features Selection

Features in the KDD cup 99 datasets cover different forms including continuous and symbolic [14]. Preprocessing is required for continuous features because theses features have different scales and this causes bias over some features over the other features. The continuous features with very large scale namely src_bytes [0, 1.3billion] and dst_bytes [0, 1.3billion] are scaled logarithmically (base 10) to the ranges [0, 9.11] and all other continuous features are scaled linearly to the range [0, 1] [15]

As mentioned in section 4, there are 41 features in KDD cup 99 dataset. The proposed algorithm uses 33 features and eliminates the features that have no relevance including 20, 21 in intrusion detection [16], or little significant in intrusion detection namely 13, 15, 17, 22, 40, and 41 [17] as shown in the table 1 (the shaded texts represent the excluded features). Elimination of these features will increase the speed of the proposed algorithm without affecting accuracy

5.2 Performance Evaluation

The effectiveness of an ID is evaluated by its capability to make accurate predictions. According to the real nature of a given event compared to the prediction from the ID, four possible outcomes are shown in table 4, known as the confusion matrix [2].

Table 4: Confusion Matrix

Actual Class	**Predicted Class**	
	Negative class (Normal)	**Positive class (Attack)**
Normal	True Negative (TN)	False positive (FP)
Attack	False Negative (FN)	True Positive (TP)

Where True negatives (TN) as well as true positives (TP) correspond to a correct operation of the IDS that is, events are successfully labeled as normal and attacks, respectively.

False positives (FP) refer to normal events being predicted as attacks; false negatives (FN) are attack events incorrectly predicted as normal events [2]. A high FP rate will seriously affect the performance of the system being detected. A high FN rate will leave the system vulnerable to intrusions. So, both FP and FN rates should be minimized, together with maximizing TP and TN rates.

In view of that, to evaluate the performance of the proposed algorithm for intrusion detection, detection rate (i.e. the proportion of true positives which are correctly identified as such), false alarm rate (i.e. the proportion of all negative substances that are incorrectly identified as positive), and accuracy (i.e. the proportion of true results in the population) based on table 4 are calculated separately as formulated in equation 11, 12, and 13 respectively [2].

$$DR = \frac{TP}{TP + FN} \qquad (11)$$

$$FAR = \frac{FP}{TN + FP} \qquad (12)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \qquad (13)$$

5.3 Results

A subset of the “10% KDD cup 99” datasets is used for training and testing the proposed algorithm.

First of all, the proposed algorithm was passed through training phase using randomly selected normal and attacks patterns from KDD cup 99 training dataset. Additionally, one normal pattern was selected randomly to initialize the center of the normal cluster and nine attack patterns were selected randomly to initialize the center of the attack cluster.

Then, the testing data of KDD cup 99 passed through the trained model to detect the intrusions and find the DR, FAR, and accuracy of the proposed algorithm.

Three experiments were carried out to test the proposed algorithm with 41 and 33 features. The results of the proposed algorithm was compared with conventional FCM that is appropriate for numeric features and FCM for symbolic features as shown in table 5.

In the first experiment (Exp1), 2776 patterns are selected randomly from “10%KDD” dataset that contains normal and the most common attacks patterns namely smurf, ipsweep, neptune and back, most attacks can be distinguished from the normal activities and the average DR is as high as 99%. At the same time, approximately 1.6% normal activities are erroneously labeled as attacks. Among the patterns labeled as attacks, about 99% of them are classified to the correct attack types.

The second experiment (Exp2) was carried out to test the ability of the proposed algorithm to detect new variations of attacks. The tested dataset contains 2776 data randomly selected from

"10%KDD" dataset, and contains different attacks which were not present in the training dataset. The DR is 99.4%, which is even better than before. At the same time, the FAR is 1.1%.

In third experiment (Exp3), the tested dataset contains normal and different attack types selected randomly but with respect to their portion in the "10%KDD" dataset. The DR is 98.6% and FAR is 1.4%.

Table 5: Experimental Results of the Proposed Algorithm

Algorithm	Metric	Exp1	Exp2	Exp3
FCM for Numeric Features	DR	48.8	71.7	51.1
	FAR	97.9	98.6	98.5
	Accuracy	29.5	58.2	41.3
FCM for Symbolic Features	DR	52.3	28.7	50
	FAR	99.7	99.8	99.5
	Accuracy	30.8	23.2	40.3
Proposed Algorithm with 41 Features	DR	98.2	99.1	98.5
	FAR	1.6	1.1	1.4
	Accuracy	98.3	99	98.2
Proposed Algorithm with 33 Features	DR	99	99.4	98.6
	FAR	1.6	1.1	1.4
	Accuracy	98.5	99.3	98.6

The software tool sipina which is a publicly available pattern classification software package [18] was used to evaluate the performance of the proposed algorithm with classifier algorithms including C4.5 and ID3 algorithm.

The C4.5 algorithm generates decision trees using an information theoretic methodology. The goal is to construct a decision tree with minimum number of nodes that gives least number of misclassifications on training data. The C4.5 algorithm uses divide and conquer strategy [15].

ID3 constructs decision tree by employing a top-down, greedy search through the given sets of training data to test each attribute at every node. It uses statistical property call information gain to select which attribute to test at each node in the tree. Information gain measures how well a given attribute separates the training examples according to their target classification. Table ٦ demonstrates the compassion results.

Table ٦: Performance Comparison of the Proposed Algorithm with C4.5 and ID3 Algorithms

Algorithm	Metric	Exp1	Exp2	Exp 3
The Proposed Algorithm with 33 Features	DR	99	99.4	98.6
	FAR	1.6	1.1	1.4
	Accuracy	98.5	99.3	98.5
C4.5	DR	99.8	98.5	96.6
	FAR	0.1	0.1	0.1
	Accuracy	99.8	98.7	97.2
ID3	DR	99.4	99.2	77.8
	FAR	0.1	0.1	0.1
	Accuracy	99.6	99.3	82.5

6. Conclusions

In this paper, a FCM with mixed features clustering algorithm was proposed for intrusion detection. This algorithm was combined both the conventional FCM and FCM for symbolic feature in one algorithm. The proposed algorithm was tested on the KDD cup 99 benchmark intrusion detection dataset.

The proposed algorithm uses 33 features from KDD cup 99 instead of 41 features. Also, the proposed algorithm gives better result than conventional FCM and FCM for symbolic features. The average DR of the proposed algorithm was 99 which outperforms the average DR of algorithms C5.4 and ID3 that have average DR is 98.3 and 92.1 respectively.

References

1. Bace, R. and Mell P., **2001**, "Intrusion Detection Systems", NIST Special Publications SP 800-31.
2. Wu, S. X. and Banzhaf W., **2010**, "The Use of Computational Intelligence in Intrusion Detection Systems: A Review", *Elsevier, Applied Soft Computing*, pp. 1-35.
3. Denning, D.E., **1987**, "An Intrusion Detection Model", *IEEE Transactions on Software Engineering* 13 (**2**), pp. 222–232.
4. Wang, Q.and Megalooikonomou V., **2005**, "A Clustering Algorithm for Intrusion Detection," In Proceedings of the SPIE Conference on Data Mining, Intrusion Detection, Information

Assurance, and Data Networks Security, Orlando, Florida, USA, March 28 - April 1, Vol. **5812**, pp. 31-38.

5. Makkithaya K., N.V. S. Reddy, and U. D. Acharya, **2008**, "Intrusion Detection System Using Modified C-Fuzzy Decision Tree Classifier", *International Journal of Computer Science and Network Security*, Vol. **8** No. 11, pp 29-35.
6. Jawhar ,M. M. T. and M. Mehrotra, **2010**, "Design Network Intrusion Detection System Using Hybrid Fuzzy-Neural Network", *International Journal of Computer Science and Security*, Vol. **4**: Issue (3), pp. 285.
7. bharti K.; Jain S., and Shukla S., **2010**, "Fuzzy K-mean Clustering via Random Forest for Intrusion Detection System", *International Journal on Computer Science and Engineering* Vol. **02**, No. 06, pp. 2197-2200.
8. Dickerson, J;E; Juslin, J; Koukousoula O.and Dickerson, J.A. **2001**, "Fuzzy Intrusion Detection," IFSA World Congress and 20th North American Fuzzy Information Processing Society (NAFIPS) International Conference, Vancouver, British Columbia, Vol. **3**, pp. 1506-1510, July.
9. Zheng, C.and Chen L., **2003**, "FCBI-an efficient User-Friendly Classifier Using Fuzzy Implication Table", In L. Kalinichenko, R. Manthey, B. Thalheim, U. Wloka (Eds.), Advances in Databases and Information Systems, Vol. **2798** of Lecture Notes in Computer Science, Springer, pp. 266–277.
10. Dunn, J. **1973**, "A Fuzzy Relative of The Isodata Process and Its Use in Detecting Compact Well–Separated Clusters", Journal of Cybernetics, Vol**. 3,** pp. 32–57.
11. Bezdek, J. C., **1981**, "*Pattern Recognition with Fuzzy Objective Function Algorithms*", New York: Plenum Press.
12. El-Sonbaty,Y. and Ismail,M.A. **1998**, "Fuzzy Clustering for Symbolic Data", IEEE Trans. Fuzzy Systems 6 (**2**), pp.195–204.
13. Yang,M; Hwang, P. and Chen,D. **2004**, "Fuzzy Clustering Algorithms for Mixed Feature Variables", Elsevier, Fuzzy Sets and Systems 141, pp.301–317.
14. [KDD-CUP 1999 Data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
15. Sabhnani, M. and Serpen,G. **2003**, "Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context", In Proceedings of the International Conference on Machine Learning: Models, Technologies, and Applications, pp. 209-215.
16. Olusola, A.; Oladele, A. S. and Abosede,D. O. **2010**, "Analysis of KDD '99 Intrusion Detection Dataset for Selection of Relevance features", Proceedings of the World Congress on Engineering and Computer Science, Vol. **I**, San Francisco, USA.
17. Farid,D.M.; Darmont, J; Harbi, N; Nguyen,H.H. and Rahman,M.Z. **2009**, "Adaptive Network Intrusion Detection Learning: Attribute Selection and Classification", International Conference on Computer Systems Engineering (ICCSE 09), Bangkok, Thailand, December.
18. Sipina Review, http://eric.univ-lyon2.fr/~ricco/sipina.html.