Global Journal of Computer Science and Technology
Neural & Artificial Intelligence
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Intelligent Intrusion Detection in Computer Networks Using Fuzzy Systems

By Amin Einipour
Islamic Azad University, Andimeshk, Iran

Abstract - The Internet and computer networks are exposed to an increasing number of security threats. With new types of attacks appearing continually, developing flexible and adaptive security oriented approaches is a severe challenge. Intrusion detection is a significant focus of research in the security of computer systems and networks. The security of computer networks plays a strategic role in modern computer systems. In order to enforce high protection levels against threats, a number of software tools are currently developed.

In this paper, we have focused on intrusion detection in computer networks by combination of fuzzy systems and Particle Swarm Optimization (PSO) algorithm. Fuzzy rules are desirable because of their interpretability by human experts. PSO algorithm is employed as meta-heuristic algorithm to optimize the obtained set of fuzzy rules. Results on intrusion detection dataset from KDD-Cup99 show that the proposed approach would be capable of classifying instances with high accuracy rate in addition to adequate interpretability of extracted rules.

Keywords : *Intrusion Detection, Fuzzy rule extraction, Particle Swarm Optimization (PSO) algorithm.*

GJCST-E Classification: *C.2.0*

Strictly as per the compliance and regulations of:

Intelligent Intrusion Detection in Computer Networks Using Fuzzy Systems

Amin Einipour

Abstract - The Internet and computer networks are exposed to an increasing number of security threats. With new types of attacks appearing continually, developing flexible and adaptive security oriented approaches is a severe challenge. Intrusion detection is a significant focus of research in the security of computer systems and networks. The security of computer networks plays a strategic role in modern computer systems. In order to enforce high protection levels against threats, a number of software tools are currently developed.

In this paper, we have focused on intrusion detection in computer networks by combination of fuzzy systems and Particle Swarm Optimization (PSO) algorithm. Fuzzy rules are desirable because of their interpretability by human experts. PSO algorithm is employed as meta-heuristic algorithm to optimize the obtained set of fuzzy rules. Results on intrusion detection dataset from KDD-Cup99 show that the proposed approach would be capable of classifying instances with high accuracy rate in addition to adequate interpretability of extracted rules.

Keywords : Intrusion Detection, Fuzzy rule extraction, Particle Swarm Optimization (PSO) algorithm.

I. Introduction

Data mining usually means the methodologies and tools for the efficient new knowledge discovery from databases. It is also a form of knowledge discovery essential for solving problems in a specific domain.

An intrusion is defined as any set of actions that attempt to compromise the integrity, confidentiality, or availability of a resource [1]. An Intrusion Detection System (IDS) monitors and restricts user access to the computer system by applying certain rules. These rules are based on expert knowledge extracted from skilled administrators, who construct attack scenarios and apply them to find system exploits. The system identifies all intrusions by users and takes or recommends necessary action to stop an attack on the database.

Two approaches to intrusion detection are currently used. The first one, called misuse detection, is based on attack signatures, i.e., on a detailed description of the sequence of actions performed by the attacker. This approach allows the detection of intrusions matching perfectly the signatures, so that new attacks performed by slight modification of known attacks cannot be detected. The second approach is based on statistical knowledge about the normal activity of the computer system, i.e., a statistical profile of what constitutes the legitimate traffic in the network. In this case, intrusions correspond to anomalous network activity, i.e. to traffic whose statistical profile deviates significantly from the normal one [2, 3].

Earlier studies, the statistical related techniques were most commonly used data mining approaches to construct classification models. However, as the intrusion detection classification problem is highly nonlinear in nature, it is hard to develop a comprehensive model taking into account all the independent variables using conventional statistical modeling techniques. Furthermore, traditional ad hoc mixtures of statistical techniques and data management tools are no longer adequate for analyzing the vast collection of data. For the needs of improving the prediction accuracy in intrusion detection, more and more researchers have tried to apply artificial intelligence related approaches for intrusion detection in computer networks [4, 5].

A good computerized detection tool should possess two characteristics. First, the tool must attain the highest possible performance. Moreover, it would be highly desirable to be in possession of a so-called degree of confidence: the system not only provides a crisp detection, but also outputs a numeric value that represents the degree to which the system is confident about its response. Second, it would be highly beneficial for such a detection system to be human-friendly, exhibiting so-called interpretability. This means that the experts in computer networks is not faced with a black box that simply spouts answers (albeit correct) with no explanation; rather, we would like for the system to provide some insight as to how it derives its outputs.

Some experimental studies reported that success of artificial neural networks in intrusion detection [6-8], but there is a major drawback in building and using a model in which the user cannot readily comprehend the final rules that neural networks models acquire. In other words, the results of training a neural network are internal weights distributed throughout the network. These weights provide no more insight into why the solution is valid than asking many human experts why a particular decision is the right decision. For example, the weights are not readily understandable although, increasingly, sophisticated techniques for probing into neural networks help provide some explanation. It is also some recently studies used

Author : Department of Computer, Andimeshk Branch, Islamic Azad University, Andimeshk, Iran. E-mail : a.einipour@gmail.com

meta-heuristic and K-NN approaches to intrusion detection which interpretability of these approaches higher than of neural networks [9-12, 21].

In this paper we combine two methodologies—fuzzy systems and meta-heuristic algorithm—so as to automatically produce systems for intrusion detection in computer networks. The major advantage of fuzzy systems is that they favor interpretability; however, finding good fuzzy systems can be quite an arduous task. This is where PSO algorithm step in, enabling the automatic production of fuzzy systems, based on a database of training cases. Our fuzzy-PSO approach produces systems exhibiting two prime characteristics: first, they attain high classification performance, with the possibility of attributing a confidence measure to the output detection; second, the resulting systems involve a few simple rules, and are therefore (human-) interpretable.

We believe the development of PSO algorithm for data mining is a promising research area, due to the following rationale.

This paper is organized as follows. In Section 2 we describe the Intrusion detection problem and KDD99 dataset, which is the focus of our interest in this paper. The third section describes the fuzzy systems based classification. Section 4 describes our particular PSO-Fuzzy approach to the intrusion detection problem. The fifth section reports on computational results evaluating the performance of the proposed system. Finally, the sixth section concludes the paper.

II. The Intrusion Detection Problem and KDD Cup 99 Dataset

The first major work in the area of intrusion detection was discussed by J.P Anderson in [13]. Anderson introduced the concept that certain types of threats to the security of computer systems could be identified through a review of information contained in the system‘s audit trail. Many types of operating systems, particularly the various ―flavors‖ of UNIX, automatically create a report which details the activity occurring on the system. Anderson identified three threats which could be identified from a concentrated review of the audit data:

1. External Penetrations - Unauthorized users of the system.
2. Internal Penetrations - Authorized system users who utilize the system in an unauthorized manner.
3. Misfeasors - Authorized user who mislead their access privileges.

Anderson indicated that a particular class of external attackers, known as clandestine users, were particularly dangerous to the system resources. Clandestine users are those who evade both system access controls and auditing mechanisms through the manipulation of system privileges or by operating at a level that is lower than what is regularly monitored by the audit trail. Anderson suggested that clandestine users could be detected by lowering the level which is monitored by the audit trail, monitoring the functions that turn off the audit systems, or through a comparison of defined ―normal‖ usage patterns of system resource usage with those levels which are currently observed.

While the concept of manually reviewing operating system audit records for indications of intrusions was recognized as an extremely inefficient method of securing a computer system, Anderson‘s article served to initiate research into the area of intrusion detection. Subsequent research involved the development of automated techniques for the review of audit record data. Until recently, most intrusion detection mechanisms were based on an automated approach to Anderson's concepts. However, the recent development of new intrusion detection approaches and, more significantly, the necessary application of intrusion detection technologies to networked environments, is changing the focus of intrusion detection research.

Dr. Dorothy Denning proposed an intrusion detection model in 1987 which became a landmark in the research in this area [14]. The model which she proposed forms the fundamental core of most intrusion detection methodologies in use today. Because of the applicability of these concepts to most accepted intrusion detection systems, an overview of the primary concepts of the model are presented here to provide a basis of understanding the core technology.

Any statistical intrusion detection methodology requires the use of a set of definable metrics. These indices are the elements upon which all of the tool's statistical analysis is based. These metrics characterize the utilization of a variety of system resources. The resources which would be used in the definition of the metrics are required to be system characteristics which can be statistically based, (i.e., CPU usage, number of files accessed, number of login attempts).

These metrics are usually one of three different types. Event counters identify the occurrences of a specific action over a period of time. These metrics may include the number of login attempts, the number of times that a file has been accessed, or a measure of the number of incorrect passwords that are entered.

The second metric, time intervals, identify the time interval between two related events. Each time interval compares the delay in occurrence of the same or similar event An example of a time interval metric is the periods of time between a user‘s logins.

Finally, resource measurement is the concept of quantifying the amount of resources used by the system over a given period of time. Resource measurement incorporates individual event counters and time interval metrics to quantify the system. Examples of resource measurements include the expenditure of CPU time,

number of records written to a database, or the number of files transmitted over the network.

While not normally considered with the "traditional" intrusion detection metrics, keystroke dynamics is another method of quantifying a user's activities which offers an effective measure of user identification. The concept involves the development of an electronic signature of a user based on their individual typing characteristics. These characteristics usually include typing speed, intervals in typing, number of errors, and the user's typing rhythm. These characteristics may be verified on login and/or monitored throughout a session. Complete intrusion detection mechanisms have been developed exclusively around the use of keystroke dynamics techniques. [15]

In 1998, DARPA in concert with Lincoln Laboratory at MIT launched the DARPA 1998 dataset for evaluating IDS [16]. The DARPA 1998 dataset contains seven weeks of training and also two weeks of testing data. In total, there are 38 attacks in training data as well as in testing data. The refined version of DARPA dataset which contains only network data (i.e. Tcpdump data) is termed as KDD dataset [17]. The Third International Knowledge Discovery and Data Mining Tools Competition were held in colligation with KDD-99, the Fifth International Conference on Knowledge Discovery and Data Mining. KDD dataset is a dataset employed for this Third International Knowledge Discovery and Data Mining Tools Competition. KDD training dataset consists of relatively 4,900,000 single connection vectors where each single connection vectors consists of 41 features and is marked as either normal or an attack, with exactly one particular attack type [18]. These features had all forms of continuous and symbolic with extensively varying ranges falling in four categories:

- In a connection, the first category consists of the intrinsic features which comprises of the fundamental features of each individual TCP connections. Some of the features for each individual TCP connections are duration of the connection, the type of the protocol (TCP, UDP, etc.) and network service (http, telnet, etc.).
- The content features suggested by domain knowledge are used to assess the payload of the original TCP packets, such as the number of failed login attempts.
- Within a connection, the same host features observe the recognized connections that have the same destination host as present connection in past two seconds and the statistics related to the protocol behavior, service, etc are estimated.
- The similar same service features scrutinize the connections that have the same service as the current connection in past two seconds.

A variety of attacks incorporated in the dataset fall into following four major categories:

Denial of Service Attacks (DOS): A denial of service attack is an attack where the attacker constructs some computing or memory resource fully occupied or unavailable to manage legitimate requirements, or reject legitimate users right to use a machine.

User to Root Attacks(U2R): User to Root exploits are a category of exploits where the attacker initiate by accessing a normal user account on the system (possibly achieved by tracking down the passwords, a dictionary attack, or social engineering) and take advantage of some susceptibility to achieve root access to the system.

Remote to User Attacks (R2L): A Remote to User attack takes place when an attacker who has the capability to send packets to a machine over a network but does not have an account on that machine, makes use of some vulnerability to achieve local access as a user of that machine.

Probes (PRB): Probing is a category of attacks where an attacker examines a network to collect information or discover well-known vulnerabilities. These network investigations are reasonably valuable for an attacker who is staging an attack in future. An attacker who has a record, of which machines and services are accessible on a given network, can make use of this information to look for fragile points.

Table1 illustrates a number of attacks falling into four major categories and table 2 presents a complete listing of a set of features characterized for the connection records.

Table 1 : Various types of attacks described in four major categories

Denial of Service Attacks(DOS)	Back, land, neptune, pod, smurf, teardrop
User to Root Attacks(U2R)	Buffer _overflow, loadmodule, perl, rootkit,
Remote to Local Attacks(R2L)	Ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster
Probes(PRB)	Satan, ipsweep, nmap, portsweep

Table 2 : A complete list of features given in KDD cup 99 dataset

Feature index	feature name	description	Type
1	duration	length (number of seconds) of the connection	Continuous
2	Protocol_type	type of the protocol, e.g. tcp, udp, etc.	Symbolic
3	service	network service on the destination, e.g., http, telnet,etc.	Symbolic
4	flag	normal or error status of the connection	Symbolic
5	Src_Bytes	number of data bytes from destination to source	Continuous
6	Dst_Bytes	number of data bytes from destination to source	Continuous
7	Land	1 if connection is from/to the same host/port; 0 otherwise	Symbolic
8	wrong_fragment	number of ``wrong'' fragments	Continuous
9	Urgent	number of urgent packets	Continuous
10	hot	number of ``hot'' indicators	Continuous
11	num_failed_logins	number of failed login attempts	Continuous
12	logged_in	1 if successfully logged in; 0 otherwise	Symbolic
13	num_compromised	number of ``compromised'' conditions	Continuous
14	root_shell	1 if root shell is obtained; 0 otherwise	Continuous
15	su_attempted	1 if ``su root'' command attempted; 0 otherwise	Continuous
16	num_root	number of ``root'' accesses	Continuous
17	num_file_creations	number of file creation operations	Continuous
18	num_shells	number of shell prompts	Continuous
19	num_access_files	number of operations on access control files	Continuous
20	num_outbound_cmds	number of outbound commands in an ftp session	Continuous
21	is_hot_login	1 if the login belongs to the ``hot'' list; 0 otherwise	Symbolic
22	is_guest_login	1 if the login is a ``guest'' login; 0 otherwise	Symbolic
23	Count	number of connections to the same host as the current connection in the past two seconds	Continuous
24	Srv_count	number of connections to the same service as the current connection in the past two seconds	Continuous
25	serror_rate	% of connections that have ``SYN'' errors	Continuous
26	srv_serror_rate	% of connections that have ``SYN'' errors	Continuous
27	rerror_rate	% of connections that have ``REJ'' errors	Continuous
28	srv_rerror_rate	% of connections that have ``REJ'' errors	Continuous
29	same_srv_rate	% of connections to the same service	Continuous
30	diff_srv_rate	% of connections to different services	Continuous
31	srv_diff_host_rate	% of connections to different hosts	Continuous
32	dst_host_count	count for destination host	Continuous
33	dst_host_srv_count	srv_count for destination host	Continuous
34	dst_host_same_srv_rate	same_srv_rate for destination host	Continuous
35	dst_host_diff_srv_rate	diff_srv_rate for destination host	Continuous
36	dst_host_same_src_port_rate	same_src_port_rate for destination host	Continuous
37	dst_host_srv_diff_host_rate	diff_host_rate for destination host	Continuous
38	dst_host_serror_rate	serror_rate for destination host	Continuous
39	dst_host_srv_serror_rate	srv_serror_rate for destination host	Continuous
40	dst_host_rerror_rate	rerror_rate for destination host	Continuous
41	dst_host_srv_rerror_rate	srv_serror_rate for destination host	Continuous

III. Fuzzy System Based Classification

Let us assume that our pattern classification problem is a c -class problem in the $n-$dimensional pattern space with continuous attributes. We also assume that m real vectors $x_p = (x_{p1}, x_{p2}, ..., x_{pn}), p = 1, 2, ..., m,$ are given as training patterns from the c classes ($c \ll m$).

Because the pattern space is $[0,1]^n$, attribute values of each pattern are $x_{pi} \in [0,1]$ for $p = 1, 2, ..., m$ and $i = 1, 2, ..., n$. In computer simulations of this paper, we normalize all attribute values of each data set into the unit interval [0, 1].

In the presented fuzzy classifier system, we use fuzzy if-then rules of the following form:

Rule R_j : If x_1 is A_{j1} and … and x_n is A_{jn} , then Class C_j with $CF = CF_j$.

where R_j is the label of the *jth* fuzzy if–then rule, $A_{j1}, ..., A_{jn}$ are antecedent fuzzy sets on the unit interval [0,1] , C_j is the consequent class (i.e., one of the given c classes), and CF_j is the grade of certainty of the fuzzy if–then rule R_j . In computer simulations, we use a typical set of linguistic values in Fig. 1 as antecedent fuzzy sets. The membership function of each linguistic value in Fig. 1 is specified by homogeneously partitioning the domain of each attribute into symmetric triangular fuzzy sets. We use such a simple specification in computer simulations to show the high performance of our fuzzy classifier system, even if the membership function of each antecedent fuzzy set is not tailored. However, we can use any tailored membership functions in our fuzzy classifier system for a particular pattern classification problem.

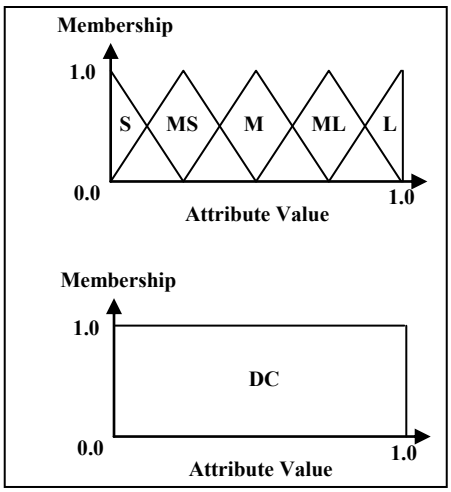

Fig. 1 : The used antecedent fuzzy sets in this paper. 1: Small, 2: medium small, 3: medium, 4: medium large, 5: large, and 6: don't care

The total number of fuzzy if–then rules is in the case of the -dimensional pattern classification problem. It is impossible to use all the fuzzy if– then rules in a single fuzzy rule base when the number of attributes (i.e.) is large (e.g., intrusion detection problem which n = 41). 6n n 6n n

Our fuzzy classifier system searches for a relatively small number of fuzzy if–then rules with high classification ability. Since the consequent class and the certainty grade of each fuzzy if–then rule can be determined from training patterns by a simple heuristic procedure [19], the task of our fuzzy classifier system is to generate combinations of antecedent fuzzy sets for a set of fuzzy if–then rules. While this task seems to be simple at first glance, in fact it is very difficult for high-dimensional pattern classification problems, since the search space involves combinations. 6n

In our fuzzy classifier system, the consequent Class and the grade of certainty of each fuzzy if– then rule are determined by a modified version of the heuristic procedure which is discussed in [18][19]. j C j CF

IV. Proposed Pso-Based Fuzzy System Approach

Swarm intelligence describes the collective behavior of decentralized, self organized natural or artificial systems. Swam intelligence model were employed in artificial intelligence. The expression was introduced in the year 1989 by Jing wang and Gerardo Beni in cellular robotic systems. Swarm Intelligence (SI) was a innovative pattern for solving optimizing problems. SI systems are typically made up of populations of simple agents interacting locally with one another and with their environment. The agent follows simple rules and the interactions between agents lead to the emergence of *"intelligent"* global behavior, unknown to the individual agents. Examples of SI include ant colonies, bird flocking, animal herding, bacterial growth and fish schooling.

The example algorithms of Swarm Intelligence are i) Ant Colony Optimization ii) Particle Swarm Optimization iii) Gravitational Search Algorithm iv) Stochastic diffusion search. Particle Swarm Optimization belongs to the class of swarm intelligence techniques that are used to resolve the optimization problems.

Particle Swarm Optimization (PSO) works with a population-based heuristic inspired by the social behavior of bird flocking aiming to find food [20]. In Particle Swarm Optimization the system initializes with a set of solutions and searches for optima by updating generations. The set of possible solutions is a set of particles, called swarm, which moves in the search space, in a cooperative search procedure. These moves are performed by an operator called velocity of a particle and moves it through an n-dimensional space based on the best positions of their leader (social component) and on their own best position (local component).

The main strength of PSO is its fast convergence, which compares with many global optimization algorithms like Genetic algorithms, Simulated Annealing and other global optimization algorithms. Particle Swarm Optimization shares many similarities with evolutionary computation techniques such as Genetic Algorithms. The system is initialized with a population of random solutions and searches for optima by updating generations. PSO has no evolution operators such as cross over and mutation. In PSO, the potential solutions called particles fly through the problem space by following the current optimum particles. PSO is a global optimization algorithm for dealing with problems in which a best solution can be represented as a point or surface search in n-dimensional space.

Hypotheses are plotted in this space and seeded with an initial velocity as well as a communication channel between the particles. Particles then move through the solution space and are evaluated according to some fitness criterion following each time step. The particles were accelerated in the direction of communication grouping which have better fitness values. The main advantage of such approach great global minimization strategies such as simulated annealing is that the large number of members that make up the particle swarm formulate the technique impressively flexible to the problem of local minima.

Each particle keeps track of its coordinates in the problem space which are associated with the best solution it has achieved so far. This value is called pbest. Another best value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called lbest. When a particle takes all the population

as its topological neighbors, the best value is a global best and is called gbest.

The basic idea of combining PSO with data mining is simple. To extract this knowledge, a database may be considered as a large search space and a mining algorithm as a search strategy. PSO makes use of particles moving in an n-dimensional space to search for solutions for an n-variable function optimization problem. The datasets are the sample space to search and each attribute is a dimension for the PSO-miner.

The pseudo-code of the PSO-Based Search algorithm is presented in figure 2. Note that the input of this algorithm is a fuzzy rule and the output is an improved version of that fuzzy rule.

```
For each particle
Initialize particle
End For
Do
For each particle
Calculate fitness value of the particle fp
/*updating particle's best fitness value so far)*/
If fp is better than pBest
set current value as the new pBest
End For
/*updating population's best fitness value so far)*/
Set gBest to the best fitness value of all particles
For each particle
Calculate particle velocity according equation (1)
Update particle position according equation (2)
End For
While maximum iterations OR minimum error criteria
is not attained
```

Fig. 2 : Pseudo-code for the PSO-Based Search algorithm

a) Proposed Fuzzy-PSO Approach for Intrusion Detection in Computer Networks

Outline of the proposed approach for intrusion detection follows in figure 3.

A fuzzy inference system is a rule-based system that uses fuzzy logic, rather than Boolean logic, to reason about data [23]. Its basic structure includes four main components, as depicted in Fig. 3: (1) a fuzzifier, which translates crisp (real-valued) inputs into fuzzy values; (2) an inference engine that applies a fuzzy reasoning mechanism to obtain a fuzzy output; (3) a defuzzifier, which translates this latter output into a crisp value; and (4) a knowledge base, which contains both an ensemble of fuzzy rules, known as the rule base, and an ensemble of membership functions, known as the database.

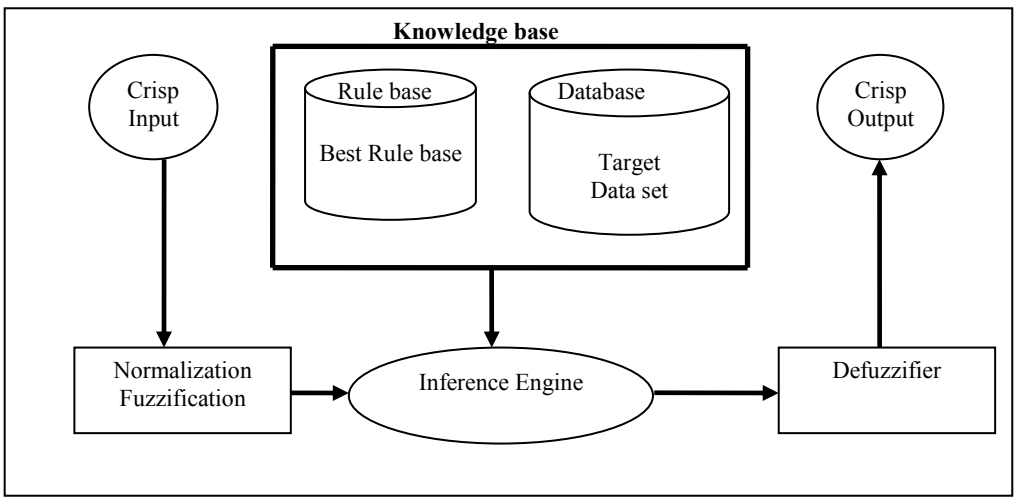

Fig. 3 : Basic structure of a fuzzy inference system for intrusion detection

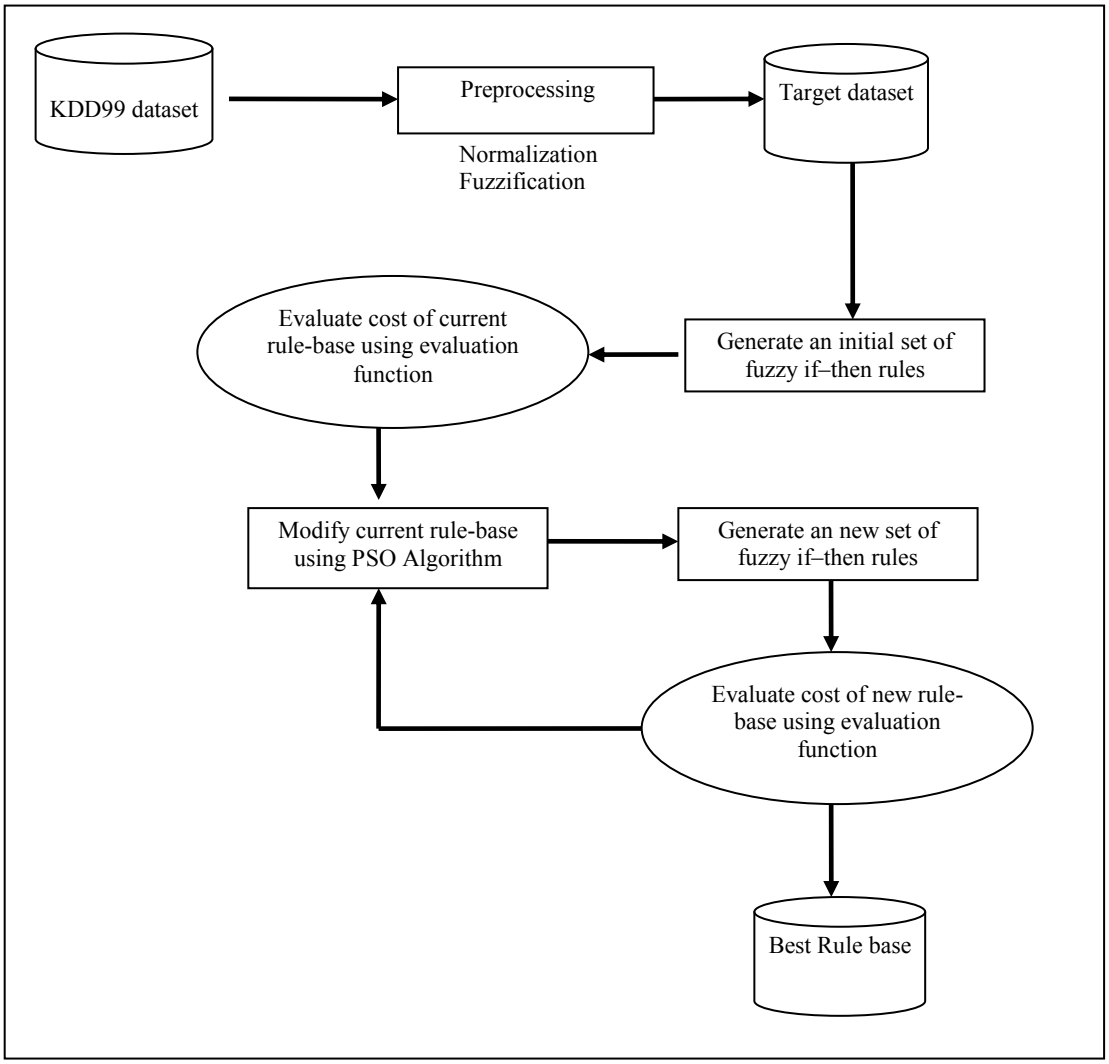

Fig. 4 : Generate Best Knowledge base Using PSO

The decision-making process is performed by the inference engine using the rules contained in the rule base. These fuzzy rules define the connection between input and output fuzzy variables.

Fuzzy modeling is the task of identifying the parameters of a fuzzy inference system so that a desired behavior is attained [24]. With the direct approach a fuzzy model is constructed using knowledge from a human expert. This task becomes difficult when the available knowledge is incompleteor when the problem space is very large, thus motivating the use of automatic approaches to fuzzy modeling.

There are several approaches to fuzzy modeling, based on neural networks [25, 26, 27], genetic algorithms [28, 29], and hybrid methods [9].

In this paper we use Particle Swarm Optimization (PSO) as automatic approach to produce knowledge base which indicated in figure 4.

In the next subsection we describe some steps of the proposed approach which presented in figure 3, 4.

i. *Generate initial rule-base*

There are different method for generate initial rule-base. One of the methods is that we assigned one of the symbols in figure 1 randomly. One method is that we increase probability of don't care fuzzy term. One innovative method is that we generate fuzzy if-then rules directly using training set. First, compatible fuzzy if-then rule is created; then antecedent part of some rules is replaced with don't care term.

The proposed evolutionary fuzzy system (EFS) is considered for each of the classes of the classification problem separately. One of the important benefits of this separation is that the learning system can focus on each of the classes of the classification problem. According to this fact, the mentioned random pattern is extracted according to the patterns of the training dataset, which their consequent class is the same as the class that the algorithm works on. Next, for this random pattern, we determine the most compatible combination of antecedent fuzzy sets using only the five linguistic

values (Figure. 1). The compatibility of antecedent fuzzy rules with the random pattern is calculated by equation (1).

$$\mu_j(x_p) = \mu_{j1}(x_{p1}) \times \ldots \times \mu_{jn}(x_{pn}) \qquad p = 1,2,\ldots,m \qquad (1)$$

Where $\mu_{A_{ji}}(.)$ is the membership function of Aji.

After generating each fuzzy if-then rule, the consequent class of this rule is determined according to (2).

$$\beta_{Class\ h}(R_j) = \sum_{x_p \in Class\ h} \mu_{R_j}(x_p) \qquad ,h = 1,2,\ldots,c \qquad (2)$$

Where,

$$\beta_{Class\ C_j}(R_j) = \max\{\beta_{Class\ 1}(R_j),\ldots,\beta_{Class\ c}(R_j)\}. \qquad (3)$$

Where β Class h (R j) is the sum of the compatibility grades of the training patterns in Class h with the fuzzy if–then rule Rj and N Class h is the number of training patterns which their corresponding class is Class h . Each of the fuzzy rules in the final classification has a certainty grade, which denotes the strength of that fuzzy rule. This number is calculated according to (4).

$$CF_j = \frac{\beta_{Class\ C_j}(R_j) - \overline{\beta}}{\sum_{h=1}^{c} \beta_{Class\ h}(R_j)} \qquad (4)$$

Where,

$$\overline{\beta} = \frac{\sum_{h \neq C_j} \beta_{Class\ h}(R_j)}{c-1} \qquad (5)$$

ii. *Evaluate rule-base*

The generation of each fuzzy rule is accepted only if its consequent class is the same as its corresponding random pattern class. Otherwise, the generated fuzzy rule is rejected and the rule generation process is repeated. After generation of N pop fuzzy if-then rules, the fitness value of each rule is evaluated by classifying all the given training patterns using the set of fuzzy if–then rules in the current population. The fitness value of the fuzzy if–then rule is evaluated by the following fitness function:

$$fitness(R_j) = NCP(R_j) \qquad (6)$$

Where $NCP(R_j)$ denotes the number of correctly classified training patterns by rule R_j .

iii. *PSO-based rule set update and optimization*

As we have mentioned in the previous subsections, initial rule-base is generated randomly, so that accuracy of the initial rule-base is low. For optimize initial rule-base, we use an PSO algorithm.

Input of this algorithm is a fuzzy rule and the output is an improved version of that fuzzy rule. The improvement is accomplished by some modifications (local search) to the current (input) fuzzy rule. The algorithm is capable of searching for the best modification according to the lifetime of the current fuzzy rule. in each step the algorithm performs one changes to the current (input) fuzzy rule. For each one value, a complete PSO process is done.

In PSO there are many fitness functions. By exploring Pareto dominance concepts, it is possible to obtain results with specific properties. Based on this concept each particle of the swarm could have different leaders, but only one may be selected to update the velocity. This set of leaders is stored in a repository, which contains the best non-dominated solutions found. The PSO components are defined as follows [22].

Each particle pi, at a time step t, has a position x (t) ϵ R^n, that represents a possible solution. The position of the particle, at time t +1, is obtained by adding its velocity, v (t) ϵ R^n, to x (t):

$$x(t+1) = x(t) + v(t+1) \qquad (9)$$

The velocity of a particle pi is based on the best position already fetched by the particle, pbest(t), and the best position already fetched by the set of neighbors of pi, Rh(t), that is a leader from the repository. The velocity update function, in time step t +1 is defined as follows:

$$v(t+1) = \varpi * v(t) + (c_1 * \phi_1) * (p_{best}(t) - x(t)) + (c_2 * \phi_2) * (R_h(t) - x(t)) \qquad (10)$$

The variables ϕ1and ϕ2, in Equation 10, are coefficients that determine the influence of the particle's positions. The constants c_1 and c_2 indicates how much each component influences on the velocity. The coefficient ϖ is the particle inertia and controls how much the previous velocity affects the current one R_h is a particle from the repository, chosen as a guide of p_i. There are many ways to make this choice. At the end of the algorithm, the solutions in the repository are the final output.

V. Experimental Result

Experiments were carried out on a subset of the database created by DARPA in the framework of the 1998 Intrusion Detection Evaluation Program [16]. We used the subset that was pre-processed by the Columbia University and distributed as part of the UCI KDD Archive [17]**Error! Reference source not found.** The available database is made up of a large number of network connections related to normal and malicious traffic. Each connection is represented with a 41-dimensional feature vector which presented in table 2. Connections are also labeled as belonging to one out of

five classes. One of these classes is the normal class and the rest indicates four different intrusion classes: PRB, DOS, U2R, and R2L which presented in table 1. These intrusion classes are a classification of 22 different types of attacks in a computer network.

This approach is implemented by using C++ programming language. We use 10-CV technique for evaluate proposed approach. In this technique, KDD99 dataset divided to 10 parties, nine parties for train set and one party for test set.

Table 3 is the confusion matrix of Proposed approach. The top-left entry of Table 3 shows that 3194 instances of the actual PRB test set were detected to be PRB; the last column indicates that 76.66% of the actual PRB samples were detected correctly. In the same way, for R2L, 1971 instances of the actual _attack' test set were correctly detected. The last column indicates that 12.17% of the actual R2L samples were detected correctly. The bottom row shows that 83.48% of the test set classified, as R2L indeed belongs to R2L. The bottom-right entry of the table 3 shows that 93.70% of all patterns in the test set are correctly classified.

Table 4 represents the cost matrix that defines the cost for each type of misclassification.

We aim at minimizing that cost function. Given the confusion and cost matrixes, we calculated the cost of our simulated annealing based fuzzy intrusion detection system as shown in table 5. The bottom-right entry of the table 5 shows that the classification cost of our algorithm is 0.1872.

Proposed approach is compared with some algorithms, such as C4.5, *k*-NN, Naïve Bayes, SVM, MLP. Result of comparison is indexed in table 6.

Table 3 : Confusion Matrix for the Proposed Approach

Real Class	Detected Class				
	PRB	DOS	U2R	R2L	%
PRB	3194	306	0	0	76.66
DOS	576	226388	10	0	98.49
U2R	116	0	37	11	16.22
R2L	77	3447	8	1971	12.17
%	76.66	98.33	46.83	83.48	**93.70**

Table 4 : Cost Matrix Used To Evaluate the Confusion of Proposed Approach

Real Class	Detected Class			
	PRB	DOS	U2R	R2L
PRB	0	2	2	2
DOS	1	0	2	2
U2R	2	2	0	2
R2L	2	2	2	0

Table 5 : Cost-Based Scoring of the Proposed Approach

Real Class	Detected Class				
	PRB	DOS	U2R	R2L	
PRB	0	612	0	0	
DOS	576	0	20	0	
U2R	232	0	0	22	
R2L	154	6894	16	0	
					0.1872

Table 6 : Recall, Precision, and F-measure For Different Classifier. The Best Values are Bold

Class	Algorithm	C4.5	5-NN Error! Reference source not found.	SVM Error! Reference source not found.	Winner Entry	Proposed Approach
PRB	Recall	81.88	81.61	**86.27**	83.30	76.66
	Precision	52.20	55. 6	77.72	64.81	**76.66**
	F-measure	63.76	66.05	81.77	72.90	**86.2**
DOS	Recall	96.99	97.00	97.65	97.10	**99.20**
	Precision	99.69	99.42	99.86	**99.88**	98.39
	F-measure	98.32	98.19	98.70	98.47	**98.79**
U2R	Recall	14.47	14.91	10.09	13.20	**16.23**
	Precision	9.35	5.47	53.49	**71.43**	32.46
	F-measure	11.36	8.00	16.97	**22.28**	21.64
R2L	Recall	1.45	6.90	3.55	8.40	**12.65**
	Precision	30.32	66.97	62.39	**98.84**	90.26
	F-measure	2.77	12.51	6.71	15.48	**22.19**

VI. Conclusions

In this paper, we focused on intrusion detection in computer networks by combination of fuzzy systems and PSO algorithm. The proposed method performs the classification task and extracts required knowledge using fuzzy rule based systems which consists of fuzzy if-then rules. Particle Swarm Optimization algorithm is employed to optimize the obtained set of fuzzy rules. The proposed system has two main features of data mining techniques which are high reliability and adequate interpretability, and is comparable with several well-known algorithms. Results on *intrusion detection* data set from KDD cup-99 repository show that the proposed approach would be capable of classifying intrusion instances with high accuracy rate in addition to adequate interpretability of extracted rules.

References Références Referencias

1. Heady, G. Luger, A. Maccabe, and M. Servilla, -*The architecture of network level intrusion detection system*‖, echnical Report, Department of Computer Science, University of New Mexico, 1990.
2. J. McHugh, A. Christie, and J. Allen, —Defending Yourself: The Role of Intrusion Detection Systems‖, IEEE Software, Sept./Oct. 2000, pp. 42-51
3. P.E. Proctor, The Practical Intrusion Detection Handbook, Prentice Hall, 2001
4. Xu L., A. Krzyzak and C.Y. Suen, "*Methods for combining multiple classifiers and their applications to handwriting recognition*", *IEEE Trans. Systems, Man and Cybernetics* 22, 1992, pp. 418-435.
5. Anderson D, Lunt TF, Javitz H, Tamaru A, Valdes A. "Detecting unusual program behavior using the statistical component of the next-generation intrusion detection expert system (NIDES)", Menlo Park, CA, USA: Computer Science Laboratory, SRI International; 1995. SRIO-CSL-95-06.
6. A.K. Ghosh and A. Schwartzbard, —*A Study in Using Neural Networks for Anomaly and Misuse Detection*", Proc. of the USENIX Security Symposium, August 23-26, 1999, Washington, USA.
7. J. Cannady, —*An adaptive neural network approach to intrusion detection and response*‖, PhD Thesis, School of Comp. and Inf. Sci., Nova Southeastern University, 2000.
8. J.M. Bonifacio et al. —*Neural Networks applied in intrusion detection systems*‖, Proc. of the IEEE World congress on Comp. Intell. (WCCI '98), 1998.
9. Mohammad saniee abadeh, Jafar Habibi, " *A Hybridization of Evolutionary Fuzzy Systems and Ant Colony Optimization for Intrusion Detection* ", INTERNATIONAL JOURNAL OF INFORMATION SECURITY, (2010), Vol. 2, No. 1, pp. 33-45.
10. Mohammad saniee abadeh, hamid mohammadi, Jafar Habibi, "*Design and analysis of genetic fuzzy systems for intrusion detection in computer networks* ", EXPERT SYSTEMS WITH APPLICATIONS, (2011), Vol. 38, No. 6, pp. 7067-7075.
11. M Saniee, Jafar Habibi, C Lucas, "*Intrusion Detection Using a Fuzzy Genetics-Based learning algorithm*", JOURNAL OF NETWORK AND COMPUTER APPLICATIONS, (2007), No. 0, pp. 414-428.

12. Mohammad Saniee Abadeh, Jafar Habibi, Zeynab Barzegar, Muna Sergi, "*A Parallel Genetic Local Search Algorithm for Intrusion Detection in Computer Networks*", ENGINEERING APPLICATIONS OF ARTIFICIAL INTELLIGENCE, (2007), Vol. 20, No. 20, pp. 1058-1069.
13. Anderson, J.P., "*Computer Security Threat Monitoring and Surveillance*" , Technical Report, J.P. Anderson Company, Fort Washington, Pennsylvania, April, 1980.
14. Denning, Dorothy, "An Intrusion-Detection Model" , IEEE Transactions on Software Engineering, February, 1987 , Vol. SE-13, No. 2.
15. Lunt, T.F., "*Automated Audit Trail Analysis and Intrusion Detection: A Survey*", Proceedings of the 11th National Computer Security Conference, 1990.
16. http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1998data.html
17. http://www.sigkdd.org/kddcup/index.php?section=1999&method=data
18. Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu and Ali A. Ghorbani, "*A detailed analysis of the KDD CUP 99 data set*", in Proceedings of the Second IEEE international conference on Computational intelligence for security and defense applications, pp. 53-58, Ottawa, Ontario, Canada, 2009.
19. H. Ishibuchi, K. Nozaki, and H. Tanaka, *—Distributed representation of fuzzy rules and its application to pattern classification*‖ , Fuzzy Sets and Systems, 52(1), 1992, pp. 21-32.
20. Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In IEEE International Conference on Neural Networks, pages 1942–1948. IEEE Press.
21. Amin Einipour *—A Fuzzy-ACO Method for Detect Breast Cancer*‖ Global Journal of Health Science October 2011.
22. Andre B. de Carvalho, Taylor Savegnago, Aurora Pozo, *"Swarm Intelligence for Rule Discovery in Data Mining"*, pp 314-319, In ICEIS 2010, Volume 2, AIDSS, Funchal, Madeira, Portugal, June 8 - 12, 2010
23. Yager RR, Zadeh LA. Fuzzy Sets, Neural Networks, and Soft Computing. New York: Van Nostrand Reinhold, 1994.
24. Yager RR, Filev DP. Essentials of Fuzzy Modeling and Control. Wiley, 1994.
25. Jang J-S R, Sun C-T. , "*Neuro-fuzzy modeling and control. Proceedings*", IEEE. 1995 83(3):378–406.
26. Lee K-M, Kwak D-H, Lee-Kwang H. , "*Fuzzy inference neural network for fuzzy model tuning*" , IEEE Trans Syst Man Cybern 1996;26(4):637–45.
27. Lin C-T, Lee CSG. , "Reinforcement structure: parameter learning for neural-network-based fuzzy logic control systems ", IEEE Trans Fuzzy Syst 1994;2(1):46–63.
28. Heider H, Drabe T., "*Fuzzy system design with a cascaded genetic algorithm*" , Proceedings of 1997 IEEE International Conference on Evolutionary Computation. IEEE and IEEE Neural Network Council and Evolutionary Programming Society, 1997:585–588.
29. Nawa NE, Hashiyama T, Furuhashi T, Uchikawa Y. , " *A study on fuzzy rules discovery using pseudo-bacterial genetic algorithm with adaptive operator*" , Proceedings of 1997 IEEE International Conference on Evolutionary Computation. IEEE and IEEE Neural Network Council and Evolutionary Programming Society, 1997.