ISSN No. 0976-5697

Volume 5, No. 3, March-April 2014

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

An Efficient Modular Approach of Intrusion Detection System based on MSPSO-DT

Mr.Mangesh R Umak
Research Scholar, Dept. of Computer Science and Engineering, OCT,Bhopal
At-Post-Madhapuri, Ta- Murtizapur, Dist- Akola, Maharashtra-444106, India.

Dr. Rachana Mishra
Head of Deptment Dept. of Computer Science and Engineering, OCT,Bhopal.
Oriental Campus, Thakral Nagar, Raisen Road, Bhopal M. P.-462021 , India.

Abstract: Intrusion detection is the act of detecting unwanted traffic on a network or a device. An IDS can be a piece of installed software or a physical appliance that monitors network traffic in order to detect unwanted activity and events such as illegal and malicious traffic, traffic that violates security policy, and traffic that violates acceptable use policies.

However, Intrusion detection systems face a number of challenges. One of the important challenges is that, the input data to be classified is in a high dimension feature space. In this paper, we are trying to present MSPSO-DT intrusion detection system. Where, Multi Swam Particle Swarm Optimization (MSPSO) is used as a feature selection algorithm to maximize the C4.5 Decision Tree classifier detection accuracy and minimize the timing speed. To evaluate the performance of the presented MSPSO-DT IDS we are trying to use several experiments on NSL-KDD benchmarked network intrusion detection dataset. Based on the MSPSO-DT system is improved intrusion detection accuracy more than 99.43%, speedup testing time is 11.13 sec and detection speed increased as compare to existing system. Moreover if the intrusion detection accuracy is improved, then we defiantly reduce network traffic problem.

Keywords: Network Security; Intrusion Detection System; Feature Selection; Multi Swam Particle Swarm Optimization; Decision Tree.

I. INTRODUCTION

The goal of intrusion detection is to monitor network assets to detect anomalous behaviour and misuse. This concept has been around for nearly twenty years but only recently has it seen a dramatic rise in popularity and incorporation into the overall information security infrastructure. Beginning in 1980, with James Anderson's paper [1], the notion of intrusion detection was born. Since then, several pivotal events in IDS technology have advanced intrusion detection to its current state. Protection from hackers on networks is currently of great importance. Recent examples of victims include the recent repeated hacking of Sony PS3 (both in April, and in May, 2011), which, according to The Telegraph [2], involved about 70 million customer accounts being vulnerable, and the hacking of websites both including US and Canadian government sites. Besides this, it has become easier for novices in hacking to get user-friendly tools and even lessons on how to successfully hack into networks with relatively sophisticated security.

Intrusion Detection System (IDS) becomes an essential component of computer networks security. IDS aim to identify unusual access or attacks to secure internal networks [3], by looking for potential malicious activities in network traffic and raises an alarm whenever a suspicious activity is detected.

There are a multitude of malicious traffic detection techniques, and thus, vulnerabilities in common security components, such as firewalls, are unavoidable. Intrusion detection systems (IDSs) and intrusion prevention systems (IPSs) are commonly used today [4]. They are used to detect different types of malicious traffic, network communications, and computer system usage with the mission of preserving systems from widespread damage; that is because other detection and prevention techniques, such as firewalls, access control, skepticism, and encryption have failed to fully protect networks and computer systems from increasingly sophisticated attacks and malware [5, 6].

IDS can be categorized into two techniques: misuse detection and anomaly detection. Misuse detection uses well defined patterns of attacks (attacks signatures) to identify known intrusion. While, Anomaly detection creates a normal behaviour profile to identify intrusions traffic based on significant deviations from this normal profile. Anomaly detection techniques have the advantage of identifying the unknown attacks [7].

Several pattern classification techniques have been proposed in the literature for the development of IDS; including Fuzzy Logic (FL) [6], [8], Neural Networks (NN) [12], Support Vector Machines (SVM) [15], Contiguous and Discontiguous Systemand [9], A Game-Theoretic Incentive-Based Mechanism [10] and Decision Tree (DT) [11].

One of the important problems for IDS is dealing with data containing high number of features. High dimensional data may leads to decrease the predictive accuracy of the IDS. Therefore, feature selection can serve as a pre-processing tool for high dimensional data before solving the classification problems. The purpose of the feature selection is to reduce the number of irrelevant and redundant features.

Different feature selection methods are proposed to increase the performance of IDS [14], [15] including Genetic Algorithm (GA) [13], [15] Principal Component Analysis (PCA) [17] and Information Gain (IG) [13].

In this paper, we are trying to propose anomaly intrusion detection system using Particle Swarm Optimization (PSO) to implement a feature selection followed by C4.5 decision tree classifier. The effectiveness of the propose PSO-DT IDS will evaluate by conducting several experiments on NSLKDD network intrusion dataset. The results reveal that our proposed PSO feature selection based IDS increases the accuracy and speed up the detection time than other well known feature selection methods compared to.

II. NETWORK INTRUSION DETECTION SYSTEM

A. An Overview of the Open Systems Interconnection Model:

A NIDS is placed on a network to analyze traffic in search of unwanted or malicious events. Network traffic is built on various layers; each layer delivers data from one point to another.

The OSI model and transmission control protocol (TCP)/IP model show how each layer stacks up. "Figure 1:" Within the TCP/IP model, the lowest link layer controls how data flows on the wire, such as controlling voltages and the physical addresses of hardware, like mandatory access control (MAC) addresses. The Internet layer controls address routing and contain the IP stack. The transport layer controls data flow and checks data integrity. It includes the TCP and user datagram protocol (UDP).

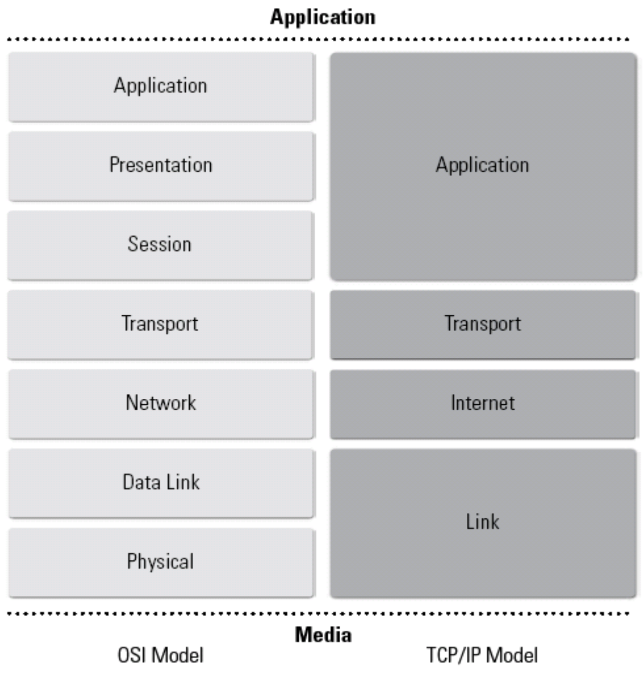

Figure. 1 OSI and TCP/IP models

Lastly, the most complicated but most familiar level is the application layer, which contains the traffic used by programs. Application layer traffic includes the Web (hypertext transfer protocol [HTTP]), file transfer protocol (FTP), email, etc. Most NIDSs detect unwanted traffic at each layer, but concentrate mostly on the application layer.

B. Component Types:

Two main component types comprise a NIDS: appliance and software only. A NIDS appliance is a piece of dedicated hardware: its only function is to be IDS. The operating system (OS), software, and the network interface cards (NIC) are included in the appliance. The second component type, software only, contains all the IDS software and sometimes the OS; however, the user provides the hardware. Software-only NIDSs are often less expensive than appliance-based NIDS because they do not provide the hardware; however, more configuration is required, and hardware compatibility issues may arise. With an IDS, the "system" component is vital to efficiency. Often a NIDS is not comprised of one device but of several physically separated components. Even in a less complicated NIDS, all components may be present but may be contained in one device. The NIDS is usually made of components identified, but more specifically, the physical components usually include the sensor, management sever, database server, and console.

a. Sensor:

The sensor or agent is the NIDS component that sees network traffic and can make decisions regarding whether the traffic is malicious. Multiple sensors are usually placed at specific points around a network, and the location of the sensors is important. Connections to the network could be at firewalls, switches, routers, or other places at which the network divides.

b. Management server:

As the analyzer, a management server is a central location for all sensors to send their results. Management servers often connect to sensors via a management network; for security reasons, they often separate from the remainder of the network. The management server will make decisions based on what the sensor reports. It can also correlate information from several sensors and make decisions based on specific traffic in different locations on the network.

c. Database server:

Database servers are the storage components of the NIDS. From these servers, events from sensors and correlated data from management servers can be logged. Databases are used because of their large storage space and performance qualities.

d. Console:

As the user interface of the NIDS, the console is the portion of the NIDS at which the administrator can log into and configure the NIDS or to monitor its status. The console can be installed as either a local program on the administrator's computer or a secure Web application portal. Traffic between the components must be secure and should travel between each component unchanged and unviewed. Intercepted traffic could allow a hacker to change the way in which a network views an intrusion.

C. NIDS Sensor Placement:

Because a sensor is the portion of the NIDS that views network traffic, its placement is important for detecting proper traffic. "Fig. 2." offers an example of how to place a NIDS sensor and other components. There are several ways to connect a NIDS sensor to the network.

a. Inline.:

An inline NIDS sensor is placed between two network devices, such as a router and a firewall. This means that all traffic between the two devices must travel through the sensor, guaranteeing that the sensor can analyze the traffic. An inline sensor of an IDS can be used to disallow traffic through the sensor that has been deemed malicious. Inline sensors are often placed between the secure side of the firewall and the remainder of the internal network so that it has less traffic to analyze.

b. Passive:

A passive sensor analyzes traffic that has been copied from the network versus traffic that passes through it. The copied traffic can come from numerous places.

c. Spanning port:

Switches often allow all traffic on the switch to be copied to one port, called a spanning port. During times of low network load, this is an easy way to view all traffic on a switch; however, as the load increases, the switch may not be able to copy all traffic. Also, if the switch deems the traffic malformed, it may not copy the traffic at all; the malformed traffic that may be the type the NIDS sensor must analyze.

d. Network tap:

A network taps copies traffic at the physical layer. Network taps are commonly used in fibre-optic cables in which the network tap is inline and copies the signal without lowering the amount of light to an unusable level. Because network taps connect directly to the media, problems with a network tap can disable an entire connection.

D. Types of Events:

A NIDS can detect many types of events, from benign to malicious. Reconnaissance events alone are not dangerous, but can lead to dangerous attacks. Reconnaissance events can originate at the TCP layer, such as a port scan. Running services have open ports to allow legitimate connections. During a port scan, an attacker tries to open connections on every port of a server to determine which services are running. Reconnaissance attacks also include opening connections of known applications, such as Web servers, to gather information about the server's OS and version. NIDS can also detect attacks at the network, transport, or application layers. These attacks include malicious code that could be used for denial of service (DoS) attacks and for theft of information. Lastly, NIDS can be used to detected less dangerous but nonetheless unwanted traffic, such as unexpected services (i.e., backdoors) and policy violations.

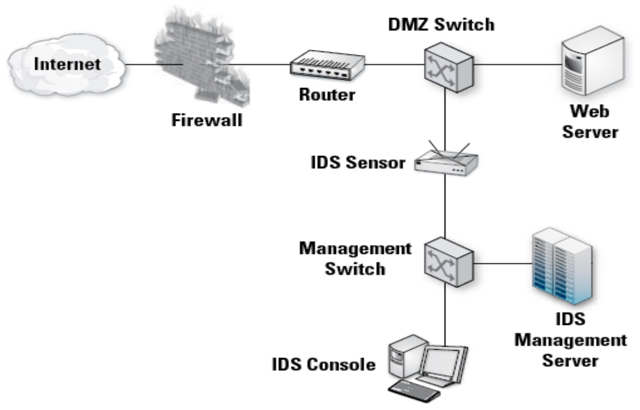

Figure. 2. NIDS Placement

III. AN OVERVIEW OF THE USED METHODS

A. Feature Selection:

Feature selection is one of the most important factors which can influence the classification accuracy rate [14]. If the dataset contains a number of features, the dimension of the space will be large and non-clean, degrading the classification accuracy rate. An efficient and robust feature selection method can eliminate noisy, irrelevant and redundant data [16].

Feature subset selection algorithms can be categorized into two types: filter algorithms and wrapper algorithms [15].

Filter algorithms select the feature subset before the application of any classification algorithm, and remove the less important features from the subset. Wrapper methods define the learning algorithm, the performance criteria and the search strategy. The learning algorithm searches for the subset using the training data and the performance of the current subset.

B. Multi-swarm scheduling module:

MSPSO is proposed, which holds a number of swarms scheduled by the multi-swarm scheduling module. Each swarm controls its iteration procedure, position updates, velocity updates, and other parameters respectively. Each swarm selects different occasions from current computing environment, then, sends the current results to the multi-swarm scheduling module to decide whether it affects other swarms. The scheduling module monitors all the sub-swarms, and gathers the results from the sub-swarms. Fig. 1 [15] shows the structure of multi-swarm scheduling model, which consists of a multi-swarm scheduler and some sub-swarms. Each sub-swarm contains a number of particles. The multi-swarm scheduler can send commands or data to sub-swarms, and vice versa.

a. The swarm request rule:

If the current sub-swarm meets the condition according to Eq. (a), it sends the results which correspond pbest and gbest values to the multi-swarm scheduler. If Si = 1, the current swarm sends records which contain the pbest and gbest values, otherwise the current swarm does not send the results [15].

$$S_i = \begin{cases} 1, \text{ if } d_i < \dfrac{tit_i - it_i}{tit_i} \times raed\,() \text{ x Fitness} \\ 0, \text{ if } d_i < \dfrac{tit_i - it_i}{tit_i} \times raed\,() \text{ x Fitness} \end{cases} \quad (1)$$

In "(1)", represents a threshold, tit the maximal iteration number, it the current iteration number. rand ( ) is a random number uniformly distributed in U (0, 1).

b. The multi-swarm scheduler request rule:

The multi-swarm scheduler monitors each sub swarm, and sends a request in order to obtain a result form current sub-swarm when the current sub-swarm is valuable. If sub-swarm has sent the swarm request rules more than k × n times, where k = 3, n = 1, 2, 3, ..., 100, the multi-swarm scheduler will send the rule. The multi-swarm scheduler request rule is touched off according to evaluating the activity level of the current sub-swarm.

The more active the sub-swarm is, the more valuable it is, since the best result may be in it.

c. The multi-swarm collection rule:

The multi-swarm scheduler collects results from the alive sub-swarm and updates pbest and gbest from storage table.

d. The multi-swarm destroying rule:

a) If the swarm sends the swarm request rule k times and *k* < *fi* according to Eq. (b), then the multi-swarm scheduler destroys the current sub-swarm.

b) If the swarm does not change the gbest in pn iterations, then the multi-swarm scheduler destroys the current sub-swarm. We set pn in the initialization of PSO.

$$f_i = \sum_{k=1}^{n} \frac{ite(k) \times m}{pk} \tag{2}$$

In "(2)", ite( ) is the function for calculating how many times the sub-swarm sends swarm request rule, m a threshold, pk the alive sub-swarm size.

C. *Data Mining Technology:*

Data mining is the extraction of hidden predictive information from large databases. It is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems [21]. Data mining tools can answer business questions that traditionally were too time consuming to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations.

D. *Decision Tree (DT):*

Decision tree (DT) introduced by Quinlan [18] is a powerful data mining algorithm for decision-making and classification problems. DT classifiers can be build from large volume of dataset with many attributes, because the tree size is independent of the dataset size.

A DT consists of three main components: nodes, leaves, and edges. Each node specifies a feature in the dataset by which the data is to be partitioned. Each node has a number of edges, which are labeled according to possible values of the feature in the parent node. An edge connects either two nodes or a node and a leaf [19]. The process of instructing a decision tree is basically a divide-and conquers process [18]. DT starts from the root node and follow the edges down until a leaf node representing the class is reached, where it divides the dataset into subsets. This process terminates when all the data in the current subset belong to the same class. C4.5 algorithm [18] uses Gain Ratio measure to choose the best attribute for each decision node during the building of the decision tree. Where at each dividing step, C4.5 choose an attribute which provides the maximum information gain while reducing the bias in favor of tests with many outcomes by normalization. Given probabilities p1; p2; :::; ps for different classes in a dataset the entropy is calculated by:

$$H(p1, p2, p3, \ldots, pn) = \sum_{n=1}^{n} \left(p_n \log\left(\frac{1}{p_n}\right) \right) \tag{3}$$

H(D)finds the amount of entropy in class based subsets of the data set. That subset is split into s new subsets S = D1;D2; ::::;Ds using some attribute, where a subset of data set does not need any further split if all examples in it belong to the same class. ID3 algorithm calculates the information gain of a split by and chooses that split which provides maximum information gain. C4.5 algorithm improves ID3 algorithm by using highest Gain Ratio that ensures a larger than average information

$$Gain(D, S) = H(D) - \sum_{i=1}^{n} p(D_i) H(D_i) \tag{4}$$

C4.5 algorithm improves ID3 algorithm by using highest Gain Ratio that ensures a larger than average information gain for the splitting purpose [20].

$$GainRatio(D, S) = \frac{Gain(D,S)}{H\left(\frac{|D_1|}{|D|}, \ldots, \frac{|D_n|}{|D|}\right)} \tag{5}$$

E. *Intrusion Detection Dataset:*

We use NSL-KDD dataset, developed by Tavallaee et al.[22], an enhanced version of KDDCup 1999 benchmark intrusion detection dataset because of the inherent problems. The first important limitation in the KDDCup 1999 [11] dataset is the huge number of redundant records in the sense that almost 78% training and 75% testing records are duplicated, as shown in Table 1 and Table 2; which cause the learning algorithm to be biased towards the most frequent records, thus prevent it from recognizing rare attack records that fall under U2R and R2L categories. At the same time, it causes the evaluation results to be biased by the methods which have better detection rates on the frequent records. It is also stated that though the NSL-KDD dataset still suffers from some of the problems discussed and may not be a perfect representative of existing real networks, it can be applied an effective benchmark dataset to detect network intrusions. In this NSL-KDD dataset, the simulated attacks can fall in any one of the following four categories [17].

a. *Probing Attack:*

This is a type of attack which collect information of target system prior to initiating an attack. Some of the examples are Satan, ipsweep, nmap attacks.

b. *DoS Attack:*

Denial of Service (DoS) attack results by preventing legitimate requests to a network resource by consuming the bandwidth or by overloading computational resources. Examples of this are Smurf, Neptune, Teardrop attacks.

c. *User to Root (U2R): Attack:*

In this case, an attacker starts out with access to a normal user account on the system and is able to exploit the system vulnerabilities to gain root access to the system. Examples are eject, load module and Perl attacks.

d. *Root to Local (R2L): Attack:*

In this, an attacker who doesn't have an account on a remote machine sends packet to that machine over a network and exploits some vulnerabilities to gain local access as a user of that machine. Some examples are ftp_write, guess password and imap attacks.

Table I. Redundant Records in KDD 1999 Training Dataset

	Original Records	**Distinct Records**	**Reduction Rate**
Attacks	3,925,650	262,178	93.32%
Normal	972,781	812,814	16.44%
Total	4,898,431	1,074,992	78.05%

Table 2. Redundant Records in KDD 1999 Testing Dataset

	Original Records	**Distinct Records**	**Reduction Rate**
Attacks	250,436	29,378	88.26%
Normal	60,591	47,911	20.92%
Total	311,027	77,289	75.15%

As there are still some critiques of attack taxonomies and performance measures, we concentrate on anomaly based intrusion detection systems with 2-class classifications, i.e., anomalous and normal, rather than identifying the detailed information of the attacks.

IV. PROPOSED ANOMOLY NETWORK INTRUSIONDETECTION SYSTEM: MSPSO-DT IDS

The proposed anomaly intrusion detection system is using the advantages of MSPSO feature selection in conjunction with C4.5 DT classifier to detect and classify the network intrusions into five outcomes: normal and four categories of intrusions. It consists of the following three fundamental building phases: (1) Preprocessing (2) Feature selection based PSO and (3) Classification using C4.5 DT.

A. *Preprocessing phase:*

The following three pre- processing stages have been done on the NSL-KDD dataset:

a. Symbolic features are converted to numeric value.
b. Each Attack name is converted to its category, 0 for Normal, 1 for DoS (Denial of service), 2 for U2R(user-to-root), 3 for R2L (remote-to-local), and 4 for Probe
c. Normalization is implemented since the data have significantly varying resolution and ranges. The features values are scaled to be within the range [0,1], using the following equation:

$$X_n = \frac{X - X_{min}}{X_{max} - X_{min}} - 1 \qquad (6)$$

Where, Xmin , Xmax are the minimum and maximum value of a specific feature. Xn is the normalized output.

B. *MSPSO Feature Selection Phase:*

In this project, MSPSO algorithm [21] has been used as a feature selection method to reduce the dimensionality of the NSL-KDD dataset. MSPSO efficiently reduces the NSL-KDD dataset from 41 features to 11 features, which reduces 73:1% of the feature dimension space. At every iteration of the PSO algorithm, each particle Xi is updated by the two best values pbest and gbest. Where, pbest denotes the best solution the particle Xi has achieved so far, and gbest denotes the global best position so far.

a. *MSPSO algorithm:*

Step 1: Load the dataset from the text file and convert the dataset from stream format to object format. Store the formatted memory data to temporary table for the initialization of PSO. Initialize the size of swarms randomly, and assign different memory to each swarm. Initialize all particle positions xij and velocities vij of each swarm with random values, then calculate objective function. Update pbest (local best) and gbest (global best) of each swarm from the table. Go to Step 2.

Step 2: Specify the parameters of each swarm including the lower and upper bounds of the velocity, the size of particles, the number of iterations, c1(the cognition learning factor), c2(social learning factor), di (in Eq. (1)), m(in the multi-swarm destroying rule) and pn(in Eq.(2)). Set iteration number = 0, current particle number = 1, titi = size of particles, and iti = current particle number. Go to Step 3.

Step 3: In each swarm, if current iteration number < iteration number or gbest keeps no changes less than 45 iterations, go to Step 4, otherwise destroy the swarm, and go to Step 10. The main scheduling module updates the pbest, and compares the gbest of current swarm with the previous one in the module, then judge whether to update gbest using multi-swarm scheduler request rule or not. If gbest or pbest is changed, execute multi-swarm collection rule.

Step 4: In each swarm, if current particle number < particle size, go to Step 5, otherwise, go to Step 9.

Step 5: In each swarm, get gbest and pbest from the table and each particle updates its position and velocity. Go to Step 6.

Step 6: Restrict position and velocity of each individual. Go to Step 7.

Step 7: Each particle calculates its fitness and updates pbest and gbest. Execute swarm request rule, and go to Step 8. If the current swarm needs to be destroyed according to multi-swarm destroying rule, dispose the current swarm, and exit.

Step 8: current particle number = current particle number + 1. Go to Step 4.

Step 9: current iteration number = current iteration number + 1. Go to Step 3.

Step 10: Execute multi-swarm collection rule, and exit.

C. *C4.5 DT classification Phase:*

A decision tree classifier is built using the C4.5 algorithm [31].Then the reduced 11 features output from the MSPSO where passed to the C4.5 decision tree classifier to be classified to one of the five categories: Normal, Dos, U2R, R2L and prob.

The proposed MSPSO-DT intrusion detection system is evaluated using the NSL- KDD dataset, where 59586 records are randomly taken. All experiments have been performed using Intel Core i3 2.13 GHz processor with 2 GB of RAM. The experiments have been implemented using Dot Net environment with Microsoft Visual Studio 10.0.

V. PERFORMANCE EVALUATION

The detection effectiveness of the proposed MSPSO-DT IDS are measured in term of TP Rate, FP Rate and F-measure; which are calculated based on the confusion matrix. The confusion matrix is square matrix where columns correspond to the predicted class, while rows correspond to the actual classes. Table 4 gives the confusion matrix, which shows the four possible prediction outcomes [32].

Table 3 CONFUSION MATRIX

	Predicted Class	
Actual Class	Normal	Attack
Normal	TN	FP
Attack	FN	TP

where,

a. ***True negatives (TN):*** indicates the number of normal events is successfully labeled as normal.
b. ***False positives (FP):*** refer to the number of normal events being predicted as attacks.
c. ***False negatives (FN):*** The numbers of attack events are incorrectly predicted as normal.
d. ***True positives (TP):*** The numbers of attack events are correctly predicted as attack.

$$TP\ Rate = \frac{TP}{TP + FN}$$

$$FN\ Rate = \frac{FP}{FP + TN}$$

$$F - measure\ = \frac{2 * TP}{(2 * TP) + FP + FN}$$

VI. EXPERIMENTS AND ANALYSIS

The classification performance measurements are shown in Table 5 and 6. Table 5 shows the accuracy measurements achieved for C4.5 classifier using the full dimension data (41 features). While, Table 6 gives the accuracy measurements for the proposed anomaly MSPSO-DT network intrusion detection system with reduced dimension feature (11 features).

Table 4. C4.5 Dt Detection Measurements (41-Dimension Feature)

Class Name	TP Rate	FP Rate	F-Measure
Normal	0.990	0.01	0.988
DoS	0.999	0.003	0.997
U2R	0.985	0.001	0.985
R2L	0.917	0.001	0.943
Probe	0.991	0.001	0.992

Table 5. Mspso-Dt Detection Measurements (11-Dimension Feature)

Class Name	TP Rate	FP Rate	F-Measure
Normal	0.989	0.006	0.991
DoS	0.999	0.002	0.998
U2R	0.990	0	0.992
R2L	0.963	0.003	0.954
Probe	0.993	0.001	0.994

From table 5 and 6, it is clear that the classification accuracy achieved using MSPSO as feature selection method with C4.5 classifier is improved than using C4.5 as standalone classifier. We compared the MSPSO feature selection method with a well known feature selection method genetic algorithm (GA).

Table 7 shows the classification accuracy of applying GA feature selection algorithm with C4.5 classifier.

Table:6 Ga-Dt Detection Measurements (12-Dimension Feature)

Class Name	TP Rate	FP Rate	F-Measure
Normal	0.982	0.012	0.983
DoS	0.998	0.002	0.997
U2R	0.967	0.003	0.958
R2L	0.932	0.003	0.935
Probe	0.983	0.002	0.985

Table 8 compare the detection accuracy, feature numbers and timing speed of C4.5, GA-DT and proposed MSPSO-DT intrusion detection systems. Table 8 illustrate that the proposed PSO-DT IDS gives better detection performance (99.43%) than the C4.5 and GA-DT IDS. Also the proposed MSPSO-DT IDS reduced the feature space from 41 to 11 features and enhance the timing speed to 11.13 sec which is important for real time network applications.

Table 7. Testing accuracy, features number and timing comparison

System	Test accuracy	Features number	Model building Time
C4.5 DT	98.45%	41	64.71 sec.
GA-DT	98.92%	12	12.26 sec.
Proposed MSPSO-DT	99.43%	11	11.13 sec.

VII. CONCLUSIONS

Intrusion Detection Systems provide the fundamental detection techniques to secure the systems present in the networks that are directly or indirectly connected to the Internet and effectively analysis the problems available in the existing intrusion detection techniques. In this paper we are providing solution on the existing intrusion detection techniques through speedup and accurate anomaly network intrusion detection system (MSPSO-DT). Where, MSPSO algorithm is used as a feature selection method and then classifies the reduced data by C4.5 decision tree classifier. The NSL-KDD network intrusion benchmark is used for conducting several experiments for testing the effectiveness of the propose MSPSO-DT network intrusion detection system. Also, a comparative study with apply existing feature selection intrusion detection techniques with C4.5 decision tree classifier is accomplish. Based on the observations this system can improve intrusion detection accuracy more than 99.50%, reduce testing time and detection speed is increased as compare to existing system. Moreover if the intrusion detection accuracy is improved, then we can defiantly reduce network traffic problem.

VIII. REFERENCES

[1] J.P. Anderson,"Computer security threat monitoring and surveillance", Technical Report, James P. Anderson Co., Fort Washington, PA, April 1980.

[2] http://www.telegraph.co.uk/technology/news/8475728/Millions-of-internet-users-hit-by-massive-Sony-PlayStationdata-theft.html. [Retrieved Sept 24, 2011]

[3] C. Tsai, Y. Hsu, C. Lin and W. Lin, "Intrusion detection by machine learning: A review", Expert Systems with Applications, vol. 36, pp.11994-12000, 2009.

[4] Cheng-Yuan Ho, Yuan-Cheng Lai, I-Wei Chen, Fu-Yu Wang, and Wei-Hsuan Tai, "Statistical Analysis of False Positives and False Negatives from Real Traffic with Intrusion Detection/Prevention Systems", IEEE Communications Magazine , March 2012.

[5] S.-X. Wu and W. Banzhaf, "The Use of Computational Intelligence in Intrusion Detection Systems: A Review," Elsevier Applied Soft Computing, vol. 10, issue 1, Jan.2010, pp. 1–35.

[6] H. T. Elshoush and I. M. Osman, "Reducing False Positives through Fuzzy Alert Correlation in Collaborative Intelligent Intrusion Detection Systems A Review,", IEEE Int'l. Conf. Fuzzy Systems, July 2000, pp. 1–8.

[7] Anna Sperotto, Michel Mandjes, Ramin Sadre, Pieter-Tjerk de Boer, and Aiko Pras "Autonomic Parameter Tuning of Anomaly-Based IDSs: an SSH Case Study", IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, VOL. 9, NO. 2, JUNE 2012.

[8] Shingo Mabu, Ci Chen, Nannan Lu, Kaoru Shimada, and Kotaro Hirasawa, "An Intrusion-Detection Model Based on Fuzzy Class-Association-Rule Mining Using Genetic Network Programming" IEEE Ttransactions on Systems, Man, and Cybernetics—part c: Applications and Reviews, vol. 41, no. 1, january 2011.

[9] Gideon Creech and Jiankun Hu†, "A Semantic Approach to Host-based Intrusion Detection Systems Using Contiguous and Discontiguous System Call Patterns", IEEE Transactions on Computers, 2013.

[10] Quanyan Zhu, Carol Fung, Raouf Boutaba, and Tamer Bas¸ar,"GUIDEX: A Game-Theoretic Incentive-Based Mechanism for Intrusion Detection Networks", IEEE Journal on Selected Areas in Communications, vol. 30, no. 11, December 2012.

[11] K.V.R. Swamy and K.S. Vijaya Lakshmi, "Network Intrusion Detection Using Improved Decision Tree Algorithm", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 10, No. 8, August 2012.

[12] G. Wang, J. Hao, J. Ma and L. Huang, "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering", Expert Systems with Applications, vol. 37, pp.6225-6232, 2010.

[13] Xiaohua Xia "Particle Swarm Optimization Method Based on Chaotic Local Search and Roulette Wheel Mechanism" 2012 International Conference on Applied Physics and Industrial Engineering, Physics Procedia 24 (2012) 269 – 275.

[14] Bing Xue, Mengjie Zhang, and Will N. Browne "Particle Swarm Optimization for Feature Selection in Classification: A Multi-Objective Approach", IEEE Transactions on Cybernetics, 2012.

[15] Yuanning Liu1, Gang Wang, Huiling Chen, Hao Dong, Xiaodong Zhu, Sujing Wang, "An Improved Particle Swarm Optimization for Feature Selection", Journal of Bionic Engineering, (2011) Vol.8 No.2.

[16] Guyon I, Elisseeff A. "An introduction to variable and feature selection." Journal of Machine Learning Research, 2003, 3, 1157–1182.

[17] Mrutyunjaya Pandaa, Ajith Abrahamb, Manas Ranjan Patra, "A Hybrid Intelligent Approach for Network Intrusion Detection" International Conference on Communication Technology and System Design 2011, Procedia Engineering 30 (2012) 1 – 9

[18] J. R. Quinlan, "C4.5 Programs for Machine Learning", Morgan Kaufmann San Mateo Ca, 1993.

[19] Y. Kuo-Ching, L. Shih-Wei, L. Chou-Yuan and L. Zne-Jung, "An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection", Applied Soft Computing, In press, 2012.

[20] D. Farid and M. Rahman, "Anomaly Network Intrusion Detection Based on Improved Self Adaptive Bayesian Algorithm", Journal of Computers, vol.5, pp. 23-31, 2010.

[21] V. Jaiganesh, M. Thenmozhi Dr. P. Sumathi, "A Survey on Building Intrusion Detection System Using Data Mining Framework", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 10, No. 3, March 2012.

[22] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 2009.