ISSN No. 0976-5697

Volume 6, No. 8, Nov-Dec 2015

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Layered approach for Intrusion Detection using PSO

B. Ben Sujitha
Associate Professor,
Department of CSE,
Ponjesly College of Engineering,
Nagercoil, India

Dr.V.Kavitha
Associate Professor,
Department of CSE,
University College of Engineering,
Kanchipuram India

Abstract: Intrusion Detection Systems (IDS) is a key part of system defense, where it identifies abnormal activities happening in a computer system. Intrusion detection system (IDS) provides a layer that monitors the network traffic for predefined suspicious patterns and inform about the unauthorized activity. In this paper, we address the effective feature selection method and combined with the layered approach. To safeguard the networks from known vulnerabilities and at the same time take steps to detect new and unseen, but possible, system abuses by developing more reliable and efficient intrusion detection systems. Any intrusion detection system has some inherent requirements. Its prime purpose is to detect as many attacks as possible with minimum number of false alarms, i.e., the system must be accurate in detecting attacks. However, an accurate system that cannot handle large amount of network traffic and is slow in decision making will not fulfill the purpose of an intrusion detection system. We desire a system that detects most of the attacks, gives very few false alarms, copes with large amount of data, and is fast enough to make real-time decisions.

Keywords: feature, Particle, Optimization, attack.

1. INTRODUCTION

The information communication infrastructure has highly improved the lives of modern society. However, this infrastructure is always under the threats of intrusion and misuse. In order to prevent such threats the research and industry community have come up with different threat detection and prevention technologies. One of such technologies is Intrusion Detection Systems (IDS). An intrusion detection system monitors and analyzes the events occurring in a computer system or network environment and alerts a human operator to the presence of possible incidents that violate standard security practices [1]. Based on the deployment area intrusion detection technologies could be categorized as Host-based IDS (deployed at individual computers) or Network-based IDS (deployed at network level). According to the methods used for analyzing the collected data, IDS can also be categorized into two broad categories: Misuse based detection and anomaly based detection.

Misuse based (signature based) intrusion detection system tries to detect malicious activities based on patterns or signatures of known attacks. If a pattern match is detected, an alarm is reported to the network administrator. Since misuse based detection system is specifically designed for detecting known attacks, it generates low number of false alarms. However, misuse based intrusion detection systems could not detect novel attacks [2]. Anomaly based intrusion detection refers to identifying events that are anomalous with respect to the normal system behavior. If the incoming network traffic patterns do not follow the normal network traffic behavior, an alarm will be reported and such patterns are called anomalies or outliers [2]. Despite their capability in detecting novel attacks anomaly based intrusion detection systems suffer from high false positive rate.

2 FEATURE SELECTION

Feature selection is the most critical step in building intrusion detection models [3], [4], [5]. During this step, the set of attributes or features deemed to be the most effective attributes is extracted in order to construct suitable detection algorithms (detectors). A key problem that many researchers face is how to choose the optimal set of features, as not all features are relevant to the learning algorithm, and in some cases, irrelevant and redundant features can introduce noisy data that distract the learning algorithm, severely degrading the accuracy of the detector and causing slow training and testing processes. Feature selection was proven to have a significant impact on the performance of the classifiers. So we have observed , particle swarm optimization technique for feature selection is good and provides in the reduction of feature .

2.1. Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) was developed byKennedy and Eberhart in 1995 [7]. PSO is an evolutionary computation technique which simulates the social behaviour of organisms, such as bird flocking. Particle swarm optimization has strong global search capability and initialized with a population of particles having a random position (solution). Each particle is associated with velocity. Particles' velocities are adjusted according to historical behaviour of each particle and its neighbours while they fly through search space. Members of swarm communicate each other and adjust their own position and velocity based on thegood position. Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far. This value is called *pgood*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbours of the particle. This location is called *lbest*. When a particle takes all the population as its

topological neighbours, the best value is a global best and is called *ggood*.In this concept ,at each time step changing the velocity of (accelerating) each particle toward its pgood and *lbes t* locations. Consider Swarm of particles is flying through the parameter space and searching for optimum. Each particle is characterized by Position vector (***xi (t))*** *a*nd Velocity vector (***vi (t))***

During the process, each particle will have its individual knowledge *pgood*, i.e.,its own best-so-far in the position and social knowledge *ggood* i.e., *pgood* of its best neighbour. the velocity will be updated using the formula(1).Thus particles have a tendency to fly towards the better and better search area over the course of search process [41]. The calculation of velocity is:

$$V_i\,(t+1) = \alpha V_i + C_1\ x\ rand\ x\left(pbest(t) - x_i(t)\right) + C_2\ x\ rand\ x\left(gbest(t) - x_i(t)\right) \quad (1)$$

where α is the inertia weight that controls the exploration and exploitation of thesearch space *c*1 and *c*2, the cognition and social components respectively are the acceleration constants which changes the velocity of a particle towards the *pgood* and *ggood*. *rand*is a random number between 0 and 1. Usually *c*1 and *c*2 values areset to 2. Performing the position update using eq .(2),

$$X_{id} = X_{id} + V_{id} \quad (2)$$

Even though PSO is an efficient method to do the feature selection it has drawback.

1. The method suffers from the partial optimism, which leads for the less exact at the regulation of its speed and the direction.
2. The method cannot work out the problems of scattering.
3. The method cannot work out the problems of non-coordinate system. To overcome the drawback multi objective PSO algorithm is used.

Three pre-processing stages are

1. Convert Symbolic features to numeric value.
2. Convert Attack names to its category, 0 for *Normal*, 1 for DoS (Denial of service) , 2 for *U*2*R* (user-to-root), 3 for *R*2*L* (remote-to-local) and 4 for *Probe*.
3. Normalize the features values, since the data have significantly varying resolution and ranges. The features values are scaled to the range [0, 1], using the following equation:

$$X_n = \frac{X - X_{min}}{X_{max} - X_{min}} - 1 \quad (3)$$

Where X_{max},X_{min} are the minimum and maximum value of a specific feature . $X_n$is a normalized output.The algorithm is initialized with a random population (swarm) of individuals (particle) , where each particle of the swarm represents a candidate solution in the d- dimensional search space. To find the best solution, each particle changes its searching direction according to: the best previous position of its individual memory (pgood) , represented by $P_i = (P_{i1}, P_{i2}, \ldots P_{id})$; the global best position gained by the swarm (ggood) $G_i = (G_{i1}, G_{i2}, \ldots G_{id})$n

The d- dimensional position for the particle i at iteration t can be represented as:

$$x_i^t = x_{i1}^t, x_{i2}^t, \ldots x_{id}^t \quad (4)$$

While the velocity (the rate of the position change) for the particle *i* at iteration t is given by

$$V_i^t = V_{i1}^t, V_{i2}^t, \ldots V_{id}^t \quad (5)$$

All of the particles have function have fitness values, which are evaluated based on a function as in eq (6):

$$\text{Fitness} = \alpha.\gamma R(D) + \beta\frac{|C|+|R|}{|C|} \quad (6)$$

Where $\gamma R(D)$ is the classification quality of condition attribute set R relative to decision D and $|R|$ is the length of selected feature subset.$|C|$is the total number of features. While , the subset parameters α and β are correspond to the importance of classification quality and subset length. α=[0,1] and β=$[1-\alpha]$.

2.2. IEM Discretization Phase

Discretization is a process of converting the continuousspace of feature into a nominal space [8]. The goal ofdiscretization process is to find a set of cut points, thesecut points partition the range into a small number ofintervals [9].In this model, the 11 features output from the PSOwhere discretised by the Information EntropyMinimization (IEM) discretization method.

Let T partition set S into subsets *S*1 and *S*2, for k classes*C*1, ...,*Ck*the class entropy of a subset S is given by

$$Ent\ (S) = -\sum_{i=1}^{k} P(C_i, S)\log\left(P(C_i, S)\right) \quad (7)$$

Where $P(C_i, S)$ is the proportion of examples in S that have class C_i. For an attribute A , the class information entropy of the partition induced by partition T is define as

$$E(A,T,S) = \frac{|S_1|}{|S|}Ent(S_1) + \frac{|S_2|}{|S|}Ent(S_2) \quad (8)$$

2.3.F-score

F-score is a simple technique which measures the discrimination of two sets of real numbers. Given training vectors $\boldsymbol{X}_k$, k = 1.2,…,*m*, if the number of positive and negative instances are *n*+ and *n*−, respectively, then the F-score of the *i*th feature is defined as follows[10]:

$$F_i = \frac{\sum_{j=1}^{l}(\bar{x}_i^{(j)} - \bar{x}_i)^2}{\sum_{j=1}^{l}\frac{1}{n_j - 1}\sum_{k=1}^{n_j}(x_{k,1}^{(j)} - x_i^{(j)})^2} \quad (9)$$

Where $\bar{x}_i$,$\bar{x}_i^{(j)}$ are the average of the ith feature of the whole dataset and the jth data set respectively . $x_{k,1}^{(j)}$is the ith feature of the kth instance in the jth dataset . From the observation of the result it is found the feature is discriminative if the F value is larger. F score is used to calculate the score of each attribute in order to get the weights of the features according to equation (10) is responsible for calculating the scores of the feature masks . ifthe ith feature is selected ("1" represents that feature *i* is selected and "0" represents that feature *i* is not selected. FS(i) equals the instance of feature *i*, otherwise FS(i) equals 0.

$$FS(i) = \begin{cases} instance\ i\ , if\ i\ is\ selected \\ 0, if\ i\ is\ not\ selected \end{cases} \quad (10)$$

2.4. Objective function

Objective function is the evaluation criteria for the selected features. To get accuracy rate, we need to train and test the dataset according to the selected features.

$$\text{Fitness}_i = \theta_a \; X \; accuracy_i + \theta_b X \left[\left[\frac{\sum_{j=1}^{N_b} F(FS(i))}{\sum_{k=1}^{N_b} F(k)}\right]\right] \quad (11)$$

$\sum_{k=1}^{N_b} F(k)$ and $\sum_{j=1}^{N_b} F(FS(i))$

the fitness of individual feature was obtained and thus that feature can be decided to be added or removed from the feature subset used.

In Eq. (11), θ_a is the weight for SVM classification accuracy rate, $accuracy_i$ the classification accuracy rate for the selected features, θ_b the weight for the score of selected features, $F(FS(i))$ the function for calculating the score of the current features, and the total score of the selected features and all features.However, most existing feature selection algorithms treat the task as a single objective problem. MSPSO is proposed, which holds a number of swarms scheduled by the multi-swarm schedulingmodule. Each swarm controls its iteration procedure, position updates, velocity updates, and other parameters respectively. The scheduling module monitors all the sub-swarms and gathers the results from the 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.

(1) The swarm request rule: If the current sub-swarm meets the condition according to Eq. (12), it sends the results which correspond *pgood* and *ggood* values to the multi-swarm scheduler. If *Si* = 1, the current swarm sends records which contain the *pgood* and *ggood* values, otherwise the current swarm does not send the results

$$S_i = \begin{cases} 1, if \; d_i < \frac{tit_i - it_i}{tit_i} x \; rand() x \; Fitness \\ 0, \; if \; d_i \geq \frac{tit_i - it_i}{tit_i} x \; rand() x \; Fitness \end{cases} \quad (12)$$

In Eq. (12), d_i represents a threshold, *tit* is the maximal iteration number, *it* is the current iteration number and *rand*( ) is a random number uniformly distributed in U (0, 1).

(2) The multi-swarm scheduler request rule: The multi-swarm scheduler monitors each subswarm, and sends a request in order to obtain a result from current sub-swarm when the current sub-swarm is valuable. If sub-swarm has sent the swarm request rules more than $k \times n$ times, where $k = 3$, $n = 1, 2, 3, \ldots, 100$, the multi-swarm scheduler will send the r1ule. The multi-swarm scheduler request rule is touched off according to evaluating the activity level of the current sub-swarm.

(3) The multi-swarm collection rule :The multi-swarm scheduler collects results from the alive sub-swarm and updates *pgood* and *ggood* from storage table.

(4) The multi-swarm destroying rule:

a. If the swarm sends the swarm request rule *k* times and $k < fi$ according to Eq. (13), then the multi-swarm scheduler destroys the current sub-swarm.

b. If the swarm does not change the *ggood* in p_n iterations, then the multi-swarm scheduler destroys the current sub-swarm. We set p_n in the initialization of PSO.

$$f_i = \frac{\sum_{l=1}^{n} ite(l) X \; m}{pl} \quad (13)$$

In Eq. (13), *ite*( ) is the function for calculating how many times the sub-swarm sends swarm request rule, *m* a threshold, *pl* the alive sub-swarm size.

3. LAYERED APPROACH FOR INTRUSION DETECTION

The reason to use the layered approach is to reduce computation and the overall time required to detect anomalous events. This can be achieved by making the layers autonomous and self-sufficient to block an attack without the need of a central decision-maker. Every layer in the LIDS[10] framework is trained separately and then deployed sequentially. We define four layers that correspond to the four attack groups mentioned in the data set as given in table 1 .They are Probe layer, DoS layer, R2L layer, and U2R layer. Each layer is then separately trained with a small set of relevant features. Feature selection is significant for Layered Approach and discussed in the previous section. The layers essentially act as filters that block any anomalous connection, thereby eliminating the need of further processing at subsequent layers enabling quick response to intrusion. The effect of such a sequence of layers is that the anomalous events are identified and blocked as soon as they are detected. Our second goal is to improve the speed of operation of the system. Hence, we implement the LIDS and select a small set of features for every layer rather than using all the41 features. This results in significant performance improvement during both the training and the testing of the system.

3.1. INTEGRATING LAYERED APPROACH WITH PSO

In Section 1, we discussed two main requirements for an intrusion detection system; accuracy of detection and efficiency in operation. As discussed in Sections 2 , the PSO can be effective in improving the attack detection accuracy by reducing the number of false alarms, while the Layered Approach can be implemented to improve the overall system efficiency. Hence, a natural choice is to integrate them to build a single system that is accurate in detecting attacks and efficient in operation. Given the data, we first select four layers corresponding to the four attack groups (Probe, DoS, R2L, and U2R) and perform feature selection for each layer, which is described next.

3.1.1 Feature Selection

Ideally, we would like to perform feature selection automatically. In this section, we describe our approach for selecting features for every layer and why some features were chosen over others. In our system, every layer is separately trained to detect a single type of attack category. We observe that the attack groups are different in their impact, and hence, it becomes necessary to treat them differently. Hence, we select features for each layer based upon the type of attacks that the layer is trained to detect.

3.1.2 Probe Layer

The probe attacks are aimed at acquiring information aboutthe target network from a source that is often external to the network. Hence, basic connection level features such as the“duration of connection” and “source bytes” are significant while features like “number of files creations” and

"numberof files accessed" are not expected to provide information for detecting probes.

3.1.3 DoS Layer

The DoS attacks are meant to force the target to stop the service(s) that is (are) provided by flooding it with illegitimate requests. Hence, for the DoS layer, traffic features such as the "percentage of connections having same destination host and same service" and packet level features such as the "source bytes" and "percentage of packets with errors" are significant. To detect DoS attacks, it may not be important to know whether a user is "logged in or not."

3.1.4 R2L Layer

The R2L attacks are one of the most difficult to detect as they involve the network level and the host level features. We therefore selected both the network level features such as the "duration of connection" and "service requested" and the host level features such as the "number of failed login attempts" among others for detecting R2L attacks.

3.1.5 U2R Layer

The U2R attacks involve the semantic details that are very difficult to capture at an early stage. Such attacks are often content based and target an application. Hence, for U2Rattacks, we selected features such as "number of file creations" and "number of shell prompts invoked," while we ignored features such as "protocol" and "source bytes."We used domain knowledge together with the practical significance and the feasibility of each feature before selecting it for a particular layer. Thus, from the total 41 features, we selected only 5 features for Probe layer,9 features for DoS layer, 14 features for R2L layer, and 8 features for U2R layer.

We now give the algorithm for integrating PSO with the Layered Approach.

Algorithm

Training

Step 1: Select the number of layers, n, for the complete system.

Step 2: Separately perform features selection for each layer.

Step 3: Train a separate model with PSO for each layer using the features selected from Step 2.

Step 4: Plug in the trained models sequentially such that only the connections labeled as normal are passed
to the next layer.

Testing

Step 5: For each (next) test instance perform Steps 6through 9.

Step 6: Test the instance and label it either as attack or normal.

Step 7: If the instance is labeled as attack, block it and identify it as an attack represented by the layer
name at which it is detected and go to Step 5Else pass the sequence to the next layer.

Step 8: If the current layer is not the last layer in the system, test the instance and go to Step 7. Else go to Step 9.

Step 9: Test the instance and label it either as normal or as an attack. If the instance is labeled as an attack, block it and identify it as an attack corresponding to the layer name.

Our final goal is to improve both the attack detection accuracy and the efficiency of the system. Hence, we integrate the PSO and the Layered Approach to build a single system. We perform detailed experiments and show that our integrated system has dual advantage. First, as expected, the efficiency of the system increases significantly. Second, since we select significant features for each layer, the accuracy of the system further increases. This is because all the 41 features are not required for detecting attacks belonging to a particular attack group. Using more features than required can result in fitting irregularities in the data, which has a negative effect on the attack detection accuracy of the system.

4 EXPERIMENTS

For our experiments, we use the benchmark KDD '99intrusion data set [6]. This data set is a version of the original1998 DARPA intrusion detection evaluation program, which is prepared and managed by the MIT Lincoln Laboratory. The data set contains about five million connection records as the training data and about two million connection records as the test data. In our experiments, we use10 percent of the total training data and 10 percent of the test data (with corrected labels), which are provided separately. This leads to 494,020 training and 311,029 test instances. Each record in the data set represents a connection between two IP addresses, starting and ending at some well defined times with a well-defined protocol. Further, every record is represented by 41 different features. Each record represents a separate connection and is hence considered to be independent of any other record. The training data is either labeled as normal or as one of the 24 different kinds of attack. These 24 attacks can be grouped into four classes; Probing, DoS, R2L, and U2R.Similarly, the test data is also labeled as either normal or as one of the attacks belonging to the four attack groups Table 1gives the number of instances for each group of attack in the data set.

Table 1: Dataset taken for experiment

	Training set	Test set
Normal	97,277	65,500
Probe	4,107	4,500
DoS	391,458	229,813
R2L	1,126	16,391
U2R	52	78
Total	494,020	316282

We note that our system is very efficient during testing. When we considered all the 41 features, the time taken to test all the 250,436 attacks was 57 seconds, which reduced to15 seconds when we performed feature selection and implemented the Layered Approach. More details will be presented when we give the detailed results for the experiments. We divide the training data into different groups; Normal, Probe, DoS, R2L, and U2R. Similarly, we divide the test data. We perform 5 experiments for each attack class by randomly selecting data corresponding to that attack class and normal data only. For example, to detect Probe attacks, we train and test the system with Probe attacks and normal data only. We do not add the DoS, R2L,and U2R data when detecting Probes. Not including these attacks while training allows the system to better learn the features for Probe attacks and normal events. When such a system is deployed online, other attacks such as DoS can either be seen as normal or as Probes. If DoS attacks are detected as normal, we expect them to be detected as attack at other layers in the

system. However, if the DoS attacks are detected as Probe, it must be considered as an advantage since the attack is detected at an early stage. Similarly, if some Probe attacks are not detected at the Probe layer; they may be detected at subsequent layers. Hence, for four attack classes, we have four independent models, which are trained separately with specific features to detect attacks belonging to that particular group. For our experiments, we report the best, the average, and the worst cases.

Table 2. Detection rate

	DoS	Probe	U2R	R2L
PSO with layered	98.0	89.8	89	50
Multi-SVM	96.8	75	5.3	4.2
PN rule	96.9	73.2	6.6	10.7
Layered CRF	97.4	98.6	86.3	29.6
PSO with SVM	97.9	98.6	68.9	19.5

Table 3. False Alarm rate

	DoS	Probe	U2R	R2L
PSO with layered	0.06	6.05	0.032	0.6
Multi-SVM	0.1	11.7	47.8	35.4
PN rule	0.05	7.5	89.5	12.0
Layered CRF	0.07	0.91	0.05	0.35
PSO with SVM	0.07	3.1	0.05	0.35

5. CONCLUSION

Our integrated system also has the advantage that any method can be used in the layers of the system. This gives flexibility to the user to decide between the time and accuracy trade-off. Furthermore, we can increase or decrease the number of layers in the system depending upon the task requirement. Finally, our system can be used for performing analysis on attacks because the attack category can be inferred from the layer at which the attack is detected. Our proposed has very good improvement in the detection and false alarm rate. The training time is reduced, so the new system has good computation facility.

6. REFERENCES

[1]. Overview of Attack Trends, http://www.cert.org/archive/pdf/attack_trends.pdf, 2002.

[2]. SANS Institute—Intrusion Detection FAQ, http://www.sans.org/resources/idfaq/, 2010.

[3]. J.P. Anderson, Computer Security Threat Monitoring and Surveillance, http://csrc.nist.gov/publications/history/ande80.pdf, 2010.

[4]. R. Bace and P. Mell, Intrusion Detection Systems, Computer Security Division, Information Technology Laboratory, Nat'l Inst. of Standards and Technology, 2001.

[5]. R. Bace and P. Mell, Intrusion Detection Systems, Computer Security Division, Information Technology Laboratory, Nat'l Inst. of Standards and Technology, 2001.

[6]. KDD Cup 1999 Intrusion Detection Data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 2010.

[7]. Yuanning Liu1,2, Gang Wang1,2, Huiling Chen1,2, Hao Dong1,2, Xiaodong Zhu1,2, Sujing Wang1,2 An Improved Particle Swarm Optimization for Feature Selection. Journal of Bionic Engineering 8 (2011)

[8]. E. Tombini, H. Debar, L. Me, and M. Ducasse, "A Serial Combination of Anomaly and Misuse IDSes Applied to HTTP Traffic," Proc. 20th Ann. Computer Security Applications Conf. (ACSAC '04), pp. 428-437, 2004.

[9]. D. Boughaci, H. Drias, A. Bendib, Y. Bouznit, and B. Benhamou, "Distributed Intrusion Detection Framework Based on Mobile Agents," Proc. Int'l Conf. Dependability of Computer Systems (DepCoS-RELCOMEX '06), pp. 248-255, 2006.

[10]. KK Gupta, B Nath, R Kotagiri, Layered Approach using Conditional Random Fields for Intrusion Detection. IEEE Trans. Dependable Secure Comput 7, 1 (2010).