International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 12 (2015) pp. 30953-30960

http://www.ripublication.com

PSO_SYN Strategy for Defending SYN Flooding Attacks

[1]Thelukutla Lokesh and [2]K Munivara Prasad

Sree vidyanikethan engineering college
Sree Sainath Nagar, A. Rangampet, Tirupati, Andhra Pradesh 517102
India
Email :{ thelukutlalokesh, prasadkmv27}@gmail.com

ABSTRACT

SYN flooding attack uses the 3-way handshaking protocol running in the TCP connection establishment phase. In a SYN flooding attack, attacker sends a large number of SYN packets to the server. Each of these request has to be handled like a connection request by the server, so the server need to answer with a SYN–ACK and must allocate a memory space to this half-open connection. Attacker tries to exhaust the memory space allotted to the TCP protocol. PSO strategy to defend against SYN-flooding DoS attacks is existing. To Improve the Detection Rate, training time and testing time for Known & unknown Attacks using the SYN-PSO algorithm is proposed.

Keywords: Binary Particle Swarm Optimization, Distributed Denial of Service Attack, Feature Selection, KDD Cup 99 dataset.

1. Introduction

There are several types of important attacks, such as the Trojan horse, worm, virus, and especially Denial of Service, eachof which causes crucial problems to usual business operations. In spite of extensive efforts to provide robustness for thesystems against DoS attack, this attack is yet a serious problem on the Internet. Traditionally, DoS attacks aim at degradingthe availability and quality of services, by consuming the service resources to make it unavailable. In doing this, DoS attacksmay send to the victim a high-rate traffic that exhausts service resources. Statistical evaluations show that DoS ranks at thefourth place in the list of the most venomous attackclasses against information systems [1]. Recently, many efforts havebeen made, in parallel with the evolution of DoS attacks, in the field of prevention and detection in networking security.In terms of prevention, some of the approaches that have been

proposed include egress [2] or ingress filtering [3], disablingunused services [4], and honey pots [5]. In other works acongestion pricing approach [6] and a router-based technique [7]have been employed to neutralize DoS attacks.

The TCP SYN flooding is the most commonly-used attack. A TCP connection is established in what is known as a 3-way handshake protocol. When a client efforts to start a TCP connection to a server, first, the client requests a connection by sending a SYN packet to the server.Then, the server returns a SYN–ACK, to the client. Finally, the client acknowledges the SYN–ACK with an ACK, at whichPoint the connection is established and data transfer commences. In a SYN flooding attack, attackers use this protocol to theirbenefit. The attacker sends a large number of SYN packets to the server. Each of these packets has to be handled like a connectionrequest by the server, so the server must answer with a SYN–ACK. The attacker then has two options. One is simplynot to answer to the SYN–ACK, which will cause the server to have a half-open connection. This would allow the server toblock any further packets from the attacker's IP address, ending the attack prematurely. Then again, the attacker spoofs the IPaddress of some unsuspecting client. The server logically answers to this IP address, but the legitimate client actually residingat this IP address will decline this SYN–ACK as it did not initiate the connection. The result is that the server is left waitingfor a reply from a large number of connections. Since resource of any system is limited, then, there are a limited number ofconnections a server can handle. Once all of these connections are active, waiting for replies that will never come, no newconnections can be established whether valid or not. Note that SYN flooding attacks aim to exhaust TCP buffer space and donot affect the parameters such as link bandwidth and processing resources.

In many fields such as classification, a large numberof features may be contained in the datasets, but notall of them are useful for classification. Redundant orirrelevant features may even reduce the classificationperformance. Feature selection aims to pick a subsetof relevant features that are sufficient to describe thetarget classes. By eliminating noisy and unnecessary features, feature selection could improve classification performance, make learning and executing processes faster, and simplify the structure of the learned models (Dash & Liu 1997) [8]. A feature selection algorithm explores the search space of different feature combinations to optimize the classification performance. The size of search space for n features is $2n$, so it is impractical to search the whole space exhaustively in most situations (Kohavi & John 1997) [9].

2. Background

2.1 Particle Swarm Optimization Algorithm

In a particle swarm optimization algorithm [10], particles is initiated randomly using random function with particles and evaluated to compute fitness with finding the particle best (pbest) and global best (gbest). Initially, each individualwith its dimensions and fitness value is assigned to its pbest. The best individual among particle best population, with its dimension and fitness value is, on the other hand, assigned to the gbest. Then a do while loop starts to converge to a minimum solution.

In the loop, particle and global bests are used to update the velocity. Then the current position of each particle is updated with the current velocity. Evaluation is again performed to compute the fitness of the particles in the swarm. This loop is terminated with a stopping criterion predetermined in advance.

Binary PSO algorithm
Initialize parameter values
Initialize particles
Evaluate
Do {
Find particle best
Find global best
Update velocity using particle and gbest and pbest
Update position
Evaluate
} While (Termination criteria met)

2.2 KDD Cup 99 Dataset

This is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition [11], which was held in conjunction with KDD-99 the Fifth International Conference on Knowledge Discovery and Data Mining. The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. A connection is a sequence of TCP packets starting and ending at some well-defined times, between which data flows to and from a source IP to a target IP under some well-defined protocol.

Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes.

Attacks fall into four main categories:

1. DOS: denial-of-service, e.g. syn flood;
2. R2L: unauthorized access from a remote machine, e.g. guessing password;
3. U2R: unauthorized access to local super user (root) privileges, e.g., various "buffer overflow" attacks;
4. Probing: surveillance and other probing, e.g., port scanning.

The datasets contain a total of 24 attack types, with an additional 14 types in the test data set only.

3. BPSO Based Feature Minimization

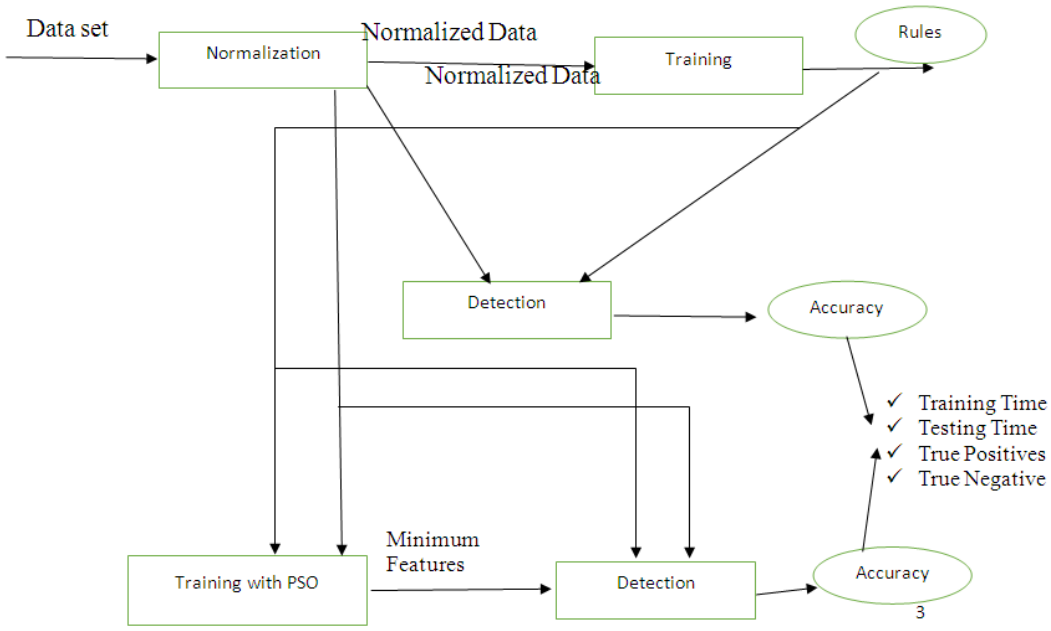

Figure 1: The process of the parameter minimization

3.1 Normalization

A problem with typical data is that different features are on different scales. This cause bias toward some features over other features. To solve this problem, we convert the data instances to a standard form based on the training dataset's Distribution. That is, we make the assumption that the training dataset accurately reflects the range and deviation of feature values of the entire distribution. Normalization also converts the data in the range of 0 and 1. If the attribute contains value then '1' otherwise '0'.

3.2 Rule Set Creation

Creationof the rule set is very crucial for detection. After normalization of data set 'S' we get the normalized data set. In this dataset we treat each object (tcp connection) as a pattern S_i. Initially the rule set is empty first simply copy the first object to the rule set then compare that object to all other objects if the new pattern arrives we copy it in rule set.

3.3 Training with PSO

The result of this section we will get the minimum number of features.by using the binary particle swarm optimization. And this section gives the detailed description abut implementation of binary particle swarm optimization. Let's look at the algorithm.

Algorithm:

For each particle
initialize particle
END
Do
For each particle
Calculate fitness value and ratio.
If (ratio >pratio) then
Pratio = ratio;
Pfitness = fitness;
Pbest = current particle;
Else if (pratio == 100 and fitness >Pfitness)
Pratio = ratio;
Pfitness = fitness;
Pbest = current particle;
If (ratio >gratio) then
Gratio = ratio;
Gfitness = fitness;
Gbest = current particle;
Else if (gratio == 100 and fitness >Gfitness)
Gratio = ratio;
Gfitness = fitness;
Gbest = current particle;
End
For each particle
Calculate particle velocity
Update particle position
End
While maximum iterations or minimum error criteria is not attained
(Ratio < 99 and fitness < total weight -1)

Updating Particle Velocity and Position:

We need to use two useful functions for generating new solutions, namely a sigmoid function to force the real values between 0 and 1, and linear function to force velocity values to be inside the maximum and minimum allowable values. So whenever a velocity value is computed, the following piece-wise function, whose range is closed interval [*V*min, *Vmax]* is used to restrict them to minimum and maximum value.

After applying the linear function, the following sigmoid function is used to scale the velocities between 0 and 1, which is then used for converting them to the binary values. That is

- Calculation of particle velocity:

v[] = v[] + c1 * rand() * (pbest - position[]) + c2 * rand() * (gbest - position[])

- Update particle position:

position[] = position[] + v[]

Particle velocity update:

- $\Delta V_{12}^{0} = 0.5 * 0.5 (Pb_{12} – X_{12}) + 0.5 * 0.5(gb_{12} – X_{12})$
- $\Delta V_{12}^{0} = 0.5 * 0.5 (1-1) + 0.5 * 0.5 (0 -1)$
- $\Delta V_{12}^{0} = -0.25$
- $V_{12}^{1} = V_{12} + \Delta V_{12}^{0}$
- $V_{12}^{1} = 1.6 + (-0.25) = 1.35$

Particle Position update:

Update position by using sigmoid function,

- Sigmoid function = $1/1+e^{-t}$
- U (0, 1) = 0.99 < sigmoid (V_{12} = 1.35) = 0.79
- $X_{12}^{1} = 0.$

Weight calculation:

Weight of attribute w_i = total number of non-zero values / total number of object

In the implementation, if the attribute value is other than zero then increment the attribute count. Total count by total number of objects in the data set. Similarly calculate the weight for all attributes.

Fitness calculation:

Fitness of particle p_i = total weight of that particle

If the particle attribute value is one then add that attribute weight to fitness. Initially the fitness is zero.

Ratio calculation:

Ratio of particle p_i= Fitness of particle / number of selected attributes in particle

Number of selected attributes is number of ones in the particle and the fitness of the particle is calculated using the above formula.

4. Results and production Runs:

For evaluating the performance of our proposed technique, we had conducted various experiments on KDD Cup 99 dataset.

$$\text{Detection Rate} = TP/ (TP+ FP)$$

Where

TP is True Positive

FP is False Positive

The detection rate is the number of attacks detected by the system divided by the number of attacks in the data set. The false positive rate is the number of normal connections that are misclassified as attacks divided by the number of normal connections in the data set.

We evaluate the Metrics Namely, Accuracy, True Positives, False Positives, Training Time, and Testing Time.

For smurf Attack with 3212 Training /30113 Testing records		
	With 41 Parameters	With Minimum Parameters
Detection Accuracy	92.9866	93.04942
Training time	270974 ms	58871 ms
Testing Time	545.1 ms	46.9 ms
True Positives	28002	28021
False Positives	2112	2093

For smurf Attack with 4477 Training /41093Testing records		
	With 41 Parameters	With Minimum Parameters
Detection Accuracy	94.51488	94.56112
Training time	181011 ms	21469 ms
Testing Time	682 ms	59.0 ms
True Positives	38839	38858
False Positives	2254	2235

For smurf Attack with 5762 Training /51538 Testing records		
	With 41 Parameters	With Minimum Parameters
Detection Accuracy	95.0193	95.05617
Training time	123600 ms	14305 ms
Testing Time	791 ms	70.4 ms
True Positives	48972	48991
False Positives	2567	2548

For smurf Attack with 6456 Training /58357 Testing records		
	With 41 Parameters	With Minimum Parameters
Detection Accuracy	94.6177	95.65036
Training time	151932ms	67353ms
Testing Time	905ms	328ms
True Positives	55217	55236
False Positives	3141	3122

Similarly we can do it for all attacks in the KDD Cup 99 Dataset. Here the results shows the Detection Accuracy with minimum features is greater than the Detection Accuracy with 41 features. And Training time and Testing time with minimum features is better than the 41 features. The True Positives and False Positives with minimum features is better than with 41 features.

5. Conclusion

The goal of this paper is to select a minimum feature subset for detecting all Distributed denial of service attacks. This goal was successfully achieved by using binary particle swarm optimization based parameter minimization. The future work is by using data mining techniques we can tune fitness function more effectively then the performance of binary particle swarm optimization is increased. If the binary particle swarm optimization performance increases, automatically the features subset length will be minimized.

1. References

[1] Gordon LA, Loeb MP, Lucyshyn W, Richardson R. CSI/FBI computer crime and security survey. ComputSecurInst 2005.

[2] Ehlert S, Geneiatakis D, Magedanz T. Survey of network security systems to counter SIP-based denial-of-service attacks. ComputSecur 2010;29(2):225–43.

[3] Yu CF, Gligor VD. A formal specification and verification method for the prevention of denial of service. In: IEEE symposium on security and privacy proceedings; 1988. p. 187–02.

[4] Warrender BPC, Forrest S. Detecting intrusions using system calls: altemative data models. In: IEEE symposium on security and privacy; 1999. p. 133–45.

[5] Hussain A, Heidemann J, Papadopoulos C. A framework for classifying denial of service attacks. USC Information Sciences Institute; 2003. p. 99–110.

[6] Xu Y, Guérin R. On the robustness of router-based denial-of-service (DoS) defense systems. SIGCOMM ComputCommun Rev 2005:47–60.

[7] Vulimiri A, Agha GA, Godfrey PhB, Lakshminarayanan K. How well can congestion pricing neutralize denial of service attacks? SIGMETRICS PerformEval Rev 2012:137–50.

[8] Dash, M. & Liu, H. (1997), 'Feature selection for classification', Intelligent Data Analysis 1, 131–156.

[9] Kohavi, R. & John, G.H. (1997), 'Wrappers for feature subset selection', Articial Intelligence 97, 315–333.

[10] M. FatihTaşgetiren& Yun-Chia Liang, 'A Binary Particle Swarm Optimization Algorithm for Lot Sizing Problem' Journal of Economic and Social Research 5 (2), 1-20

[11] http://www.sigkdd.org/kddcup/index.php?section=1999&method=info