International Journal of the Physical Sciences Vol. 6(19), pp. 4558-4563, 16 September, 2011
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS11.860
ISSN 1992 - 1950 ©2011 Academic Journals

Full Length Research Paper

Self-managing defense against SYN-flooding attacks

Gholam Shaker[1]* and Shahram Jamali[2]

[1]Department of Computer Engineering, Zanjan Branch, Islamic Azad University, Zanjan, Iran.
[2]Department of Computer Engineering, University of Mohaghegh Ardabili, Ardabil, Iran.

Accepted 25 July, 2011

SYN-flooding attack uses the weakness available in TCP's three-way handshake process to keep it from handling legitimate requests. This attack causes the victim host to populate its backlog queue with forged TCP connections. In other words it increases PSA (probability of success of attack) and decreases BUE (buffer utilization efficiency) in the victim host and results to decreased performance of the host. This paper proposes a self-managing approach, in which the host defends against SYN-flooding attack by dynamically tuning off its own two parameters, that is, *m* (maximum number of half-open connections) and *h* (hold time for each half-open connection). In this way, it formulates the defense problem, an optimization problem and then employs the particle swarm optimization (PSO) algorithm to solve it. The simulation results show that the proposed defense strategy improves performance of the under attack system in terms of BUE and PSA.

Key words: SYN-flooding, PSO, DoS, TCP, queuing model.

INTRODUCTION

Security has become necessary in a world where more services are relying on internet technology. For this reason, it has attracted a lot of attention in various areas of communication networks (Alam et al., 2011; Al-Bakri et al., 2011; Nejati and Khoshbin, 2010). One of the security breaches is denial-of-service (DoS) attack. A DoS attack can be considered as an attempt of attackers to prevent legal users from gaining a normal network service (Siris, 2006; Wang, 2007; Bicakci, 2009). Recent evaluations (Gordon, 2005; Hamdi, 2007) show that DoS attacks ranks at the fourth place in the list of the most important attack classes for information systems. More than 90% of distributed denial-of-service (DDoS) attacks exploit a system's transmission control protocol (TCP) (Wang et al., 2002a). A well known DoS attack is SYN-flooding attack. A TCP connection is established in what is known as a 3-way handshake. When a client efforts to start a TCP connection to a server, firstly, 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 which point the connection is established and data transfer commences (Safa, 2008; Xiao, 2008). In a SYN flooding attack, attackers use this protocol to their benefit. The attacker sends a large number of SYN packets to the server. Each of these packets has to be handled like a connection request by the server, so the server must answer with a SYN-ACK. The attacker does not answer to the SYN-ACK, which will cause the server to have a half-open connection. The result is that the server is left waiting for a reply from a large quantity of connections. There are a limited number of connections a server can handle. Once all of these are in use, waiting for connections that will never come, no new connections can be made whether valid or not. There are some proposed defenses for this attack. Zuquete (2002) proposes SYN cookies to defend against SYN-flooding attacks. A SYN-flood detection approach was proposed in (Wang et al., 2002b). This approach monitors the difference between the number of SYN segments and the number of FIN or RST segments since, under normal TCP behavior, each SYN will correspond to a FIN or RST. Therefore, a sharp rise in difference between the number of SYNs and FINs/RSTs, within a certain time frame, is indicative

*Corresponding author. E-mail: gholamshaker@gmail.com. Tel: 0989360239587.

of a SYN flooding attack. Chang (2002) mentioned a simple queuing model for the SYN-flooding attack. Long (2005) proposed two queuing models for the DoS attacks in order to obtain the packet delay jitter and the loss probability. Peng (2003) compiled an IP address database of previous successful connections. When a network was suffering from traffic congestion, an IP address that did not appear in the database was construed as more suspicious. As another work (Xiao, 2008) proposes an autonomous approach in which the victim host defends against SYN-flooding attack by itself and does not involves ISP, router and other network devices. Ling (2009) proposed a defense procedure that uses the edge routers that connect end hosts to the Internet to store and detect whether the outgoing SYN, ACK or incoming SYN/ACK segment is valid. This is accomplished by maintaining a mapping table of the outgoing SYN segments and incoming SYN/ACK segments, and creating the destination and source IP address database. In Ming (2009) a probabilistic drop scheme is given for implementation in a host server to mitigate SYN-flooding attacks. It proposes an analysis for this scheme, and a general principle for evaluation of the probability of successful connection establishment is presented.

PARTICLE SWARM OPTIMIZATION ALGORITHM

PSO (Kennedy, 1995; Zahiri, 2007) technique finds the optimal solution using a population of particles. Each particle represents a candidate solution to the problem. PSO is basically developed through simulation of bird flocking in two dimensional spaces. Some of the attractive features of the PSO include ease of implementation and the fact that no gradient information is required. It can be used to solve a wide array of different optimization problems; some example applications include neural network training and function minimization. The PSO is a population based optimization technique, where the population is called a swarm. A simple explanation of the PSO's operation is as follows. Each particle represents a possible solution to the optimization task. During each iteration, each particle accelerates in the direction of its own personal best solution found so far, as well as in the direction of the global best position discovered so far by any of the particles in the swarm.

It lies somewhere between evolutionary programming and the genetic algorithms. As in evolutionary computation paradigms, the concept of fitness is employed and candidate solutions to the problem are termed particles, each of which adjusts its flying based on the flying experiences of both itself and its companion. The position and velocity of particle i at iteration k can be, respectively expressed as:

$$X_i(k) = [X_{i1}(k), X_{i2}(k), \dots, X_{iN}(k)] \tag{1}$$

$$V_i(k) = [V_{i1}(k), V_{i2}(k), \dots, V_{iN}(k)] \tag{2}$$

Particle i keeps track of its coordinates in the solution space which are associated with the best solution that has been achieved so far by that particle. This value is called local best, Lbesti. Another best value that is tracked by the PSO is the best value obtained so far by any particle in the neighborhood of that particle. This value is called global best, Gbest. The basic concept of PSO lies in accelerating each particle toward its local best and the global best locations. The velocity and position of particle *i* at iteration k+1 can be calculated according the following equations:

$$\begin{aligned} V_i(k+1) = {} & wV_i(k) + c_1 r_1 \big(Lbest_i(k) - X_i(k)\big) \\ & + c_2 r_2 \big(Gbest(k) - X_i(k)\big) \end{aligned} \tag{3}$$

$$X_i(k+1) = X_i(k) + V_i(k+1) \tag{4}$$

where *w* is the inertia weight, *c1* and *c2* are constants which determine the influence of the local best position $Lbest_i(k)$ and the global best position Gbest(k). Parameters *r1* and *r2* are random numbers uniformly distributed within (0, 1).

QUEUING MODEL

Since queues provide the most intuitive language for explaining traffic and its dependence structure, in the network environment (Rolls, 2005), in this work we use queuing theory to draw a defense map against SYN-flooding attacks. Although a computer system includes several resources, for simplicity, we consider only one resource that is, memory and corresponding backlog buffer. In this model, all connection requests share the same backlog buffer. When a request arrives at the system, the system instantly receives a buffer space of the backlog queue upon finding an inactive buffer space and is blocked otherwise. Now, consider a server under the SYN-flooding attacks. Assume that in this computer each half-open connection is held for at most a period of time *h*, and at most *m* concurrent half-open connections are allowed. We assume that a half-open connection for a regular request packet is held for a chance time which is exponentially distributed with parameter *μ*. The arrivals of the regular request packets and the attack packets are both Poisson processes with rates *λ1* and *λ2*, respectively. The two arrival processes are independent of each other and of the holding times for half-open connections. Obviously, when the system is under attack, then number of pending connections increases and in a point in which there is no more room for pending connection to be saved the arriving packets will be blocked. This leads to increased number of lost connections. On the other hand, when a server is under SYN-flooding attacks, half-open connections can quickly consume all the memory allocated for the pending connections and prevent the victim from further accepting new requests, leading to the well-known buffer overflow problem. In this case, less percentage of buffer space is occupied by legal requests, and major part of this space will be allocated to the attacker requests.

PROPOSED APPROACH

Since values of *h* and *m* can affect success of SYN-flooding attack remarkably and hence can be considered as important parameters in our defense design. We use the PSO algorithm to design a defense procedure in which *h* and *m* are set dynamically based on the system condition. Since, SYN-flooding attack tries to maximize number of attack half-open connections and run over buffer space, then we can redefine the defense against SYN-flooding attack as an optimization problem that tries to minimize PSA and also maximize BUE, where the parameters are defined as follows:

1. BUE is buffer utilization efficiency and refers to mean ratio of the number of regular half-open connections to total number of half-open connections (regular and attack connections).
2. PSA is probability of success of attack and can be defined as ratio of the number of attack half-open connections to total number of half-open connections.

To involve our design goals that is maximized BUE and minimized PSA, we provide the following objective function to link the design requirements with the optimization algorithm:

$$\text{Objective Function:} \quad \text{Maximize } \frac{BUE}{PSA} \tag{5}$$

This defense scheme employs PSO algorithm to find those values of h and m parameters that optimizes the objective function of Equation 5. In this approach, *h* and *m* will be calculated by PSO algorithm according to the following equations:

$$v_h^{k+1} = wv_h^k + c_1 r_1\left(Lbest_h^k - h^k\right) + c_2 r_2\left(Gbest^k - h^k\right) \tag{6}$$

$$h^{k+1} = h^k + v_h^{k+1} \tag{7}$$

$$v_m^{k+1} = wv_m^k + c_1 r_1\left(Lbest_m^k - m^k\right) + c_2 r_2\left(Gbest^k - m^k\right) \tag{8}$$

$$m^{k+1} = m^k + v_m^{k+1} \tag{9}$$

where ($Pbest_h$, $Pbest_m$) is the local best position and ($Gbest_h$, $Gbest_m$), is the global best position. These best positions are selected according to the objective function of Equation 5.

The parameters *c1* and *c2* determine the relative pull of Lbest and Gbest and the parameters *r1* and *r2* lead to stochastically varying these pulls. These parameters should be selected sensitively for efficient performance of proposed approach. The constants *c1* and *c2* represent the weighting of the stochastic acceleration terms that pull each particle toward Lbest and Gbest positions. Low values allow particles to roam far from the target regions before being tugged back. On the other hand, high values result in abrupt movement toward, or past, target regions. Hence, the acceleration constants are set to be 0.5. Suitable selection of inertia weight, *w*, provides a balance between global and local explorations, thus requiring less iteration on average to find a sufficiently optimal solution. We set *w* to be 0.9.

IMPLEMENTATION AND SIMULATION RESULTS

Here, we study the proposed defense scheme by using two essential security metrics namely, the probability of success of attack and the buffer utilization efficiency. These two metrics represent how severe the SYN-flooding attacks affect the system performance. For this purpose, we follow the manner of Wang (2007) and give some numerical examples to exhibit how to quantify these security metrics. Let $\lambda 1$=10 as the parameter for the Poisson arrival process of the regular request packets and $\lambda 2 = k\lambda 1$, as the Poisson arrival process of the attack request packets, in which, *k* represent the ratio between arrival rates of the attack packets and the regular request packets. We use the exponential distribution with the parameter μ=100 /s as the service time of regular request packets, and it could represent the strictness of congestions in the network. In order to study proposed approach performance in wide range of attack intensity, we change *k* from 0.1 to 2. Total number of connection request is considered, 50000 requests, some of them are legal requests and others are attack connection requests. As a reference point we compare our approach with Linux in which *m*=128 and *h*=75s statically (Philip, 2008).

Figure 1 shows some simulation results to study about BUE of the proposed defense. It presents BUE for a wide range of attack intensity from *k*=0.1 to k=2 in which BUE decreases as attack intensity grows up. But, it can be seen in Figure 1a that when *h* and *m* is tuned dynamically by PSO, BUE is remarkably lower than the cases in which *h* and *m* are set statically to 75 and 128, respectively. Figure 1b shows these dynamic values of *h* and *m* generated by PSO algorithm in our defense scheme.

Figure 2 shows how the proposed defense scheme improves performance of the under attack server in

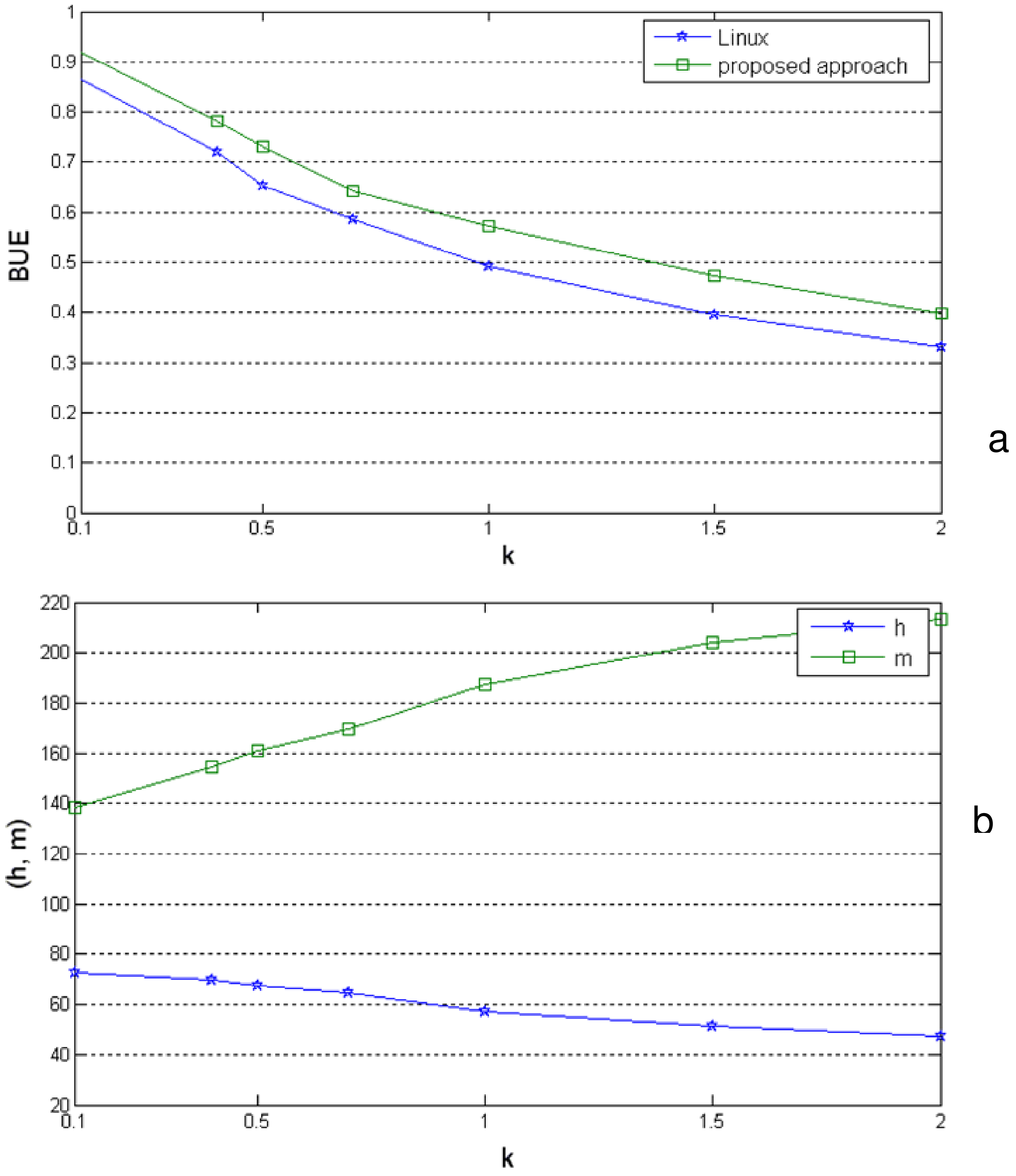

Figure 1. Buffer utilization efficiency analysis: (a) comparison of BUE of Linux and proposed approach, (b) evolution of *h* and *m*.

term of the PSA. We can observe in Figures 2a that the proposed algorithm, keeps PSA in lower level comparing to the case that uses fix values for *h* and *m*. These better results are coming from dynamic and intelligent setting of *h* and *m* shown in Figure 2b. While Linux uses fixed value for *h* this figure shows that when *k* increases and attack intensity goes up, PSO decreases *h*. Hence, long life half-open connections that typically are attack connection are closed and this makes new capacity to accept new legal connections. On the other hand, in contrast with Linux that uses fixed values for *m*, Figure 2b shows that when *k* increases PSO increases *m*. This makes new capacity for coming legal connections and hence decreases rejection probability of legal

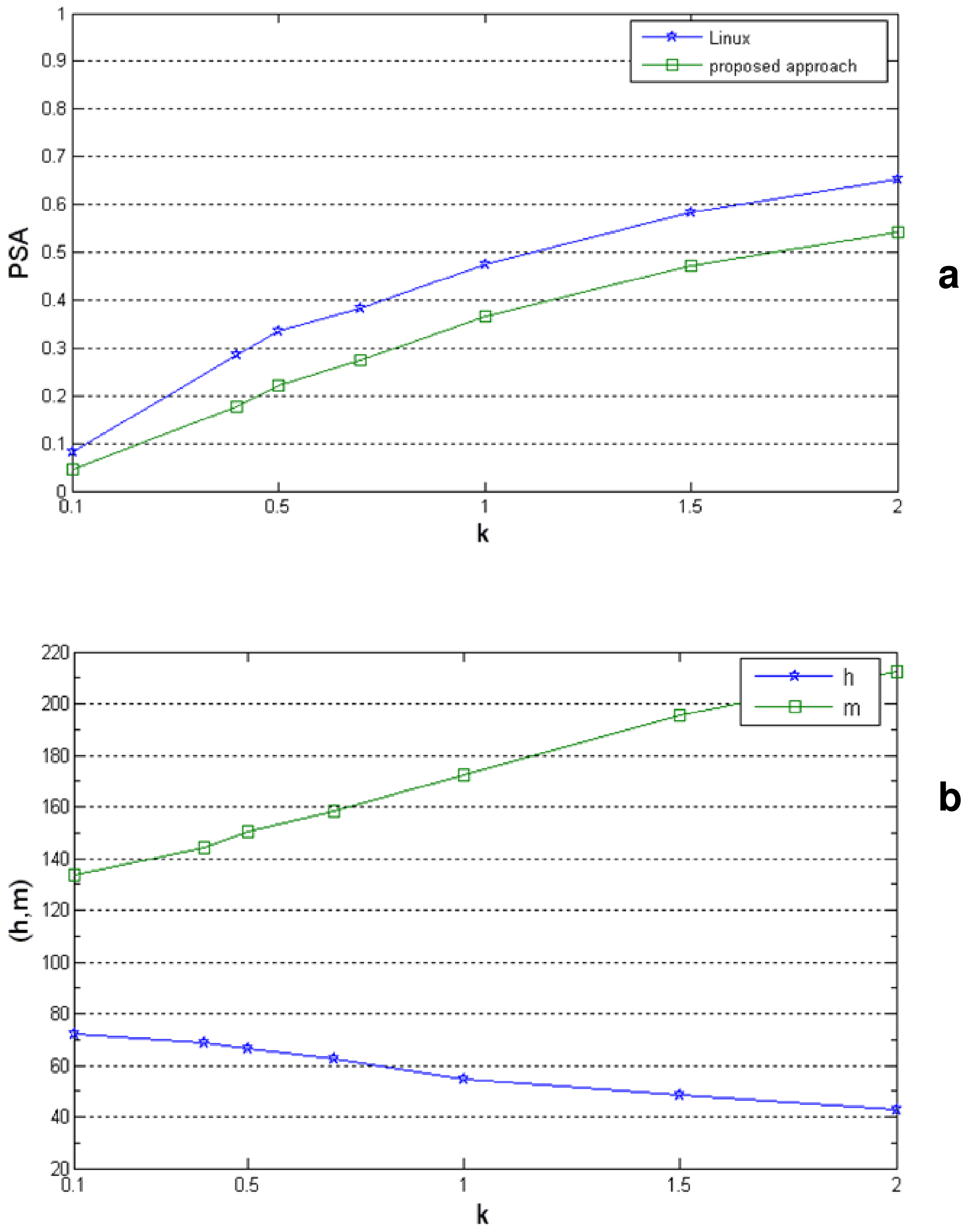

Figure 2. Probability of success of attack analysis: (a) comparison of PSA of Linux and proposed approach, (b) evolution of *h* and *m*.

connections.

Finally Table 1 compares average behaviors of proposed approach with Linux. In this case 50000 requests (attack and regular requests) have been sent to the server. Attack intensity is varying from 0.1 to 2 during the simulation time. According to this table, our proposed approach behaves better than Linux in efficient utilization of buffer and leads to higher BUE comparing to the Linux. On the other hand, Table 1 shows PSA of proposed approach is lower than that of Linux. This means that the proposed approach defends more efficient against SYN-flooding attacks

Table 1. Average BUE and PSA for Linux and the proposed approach.

Algorithm	BUE (%)	PSA (%)
Linux	57	40
Proposed approach	64	35

and decreases number of successful attack request that occupy buffer space.

Future works

This research can be pursued in the following directions. First of all, since in this paper the defense against SYN-flooding attack has been redefined as an optimization problem, hence other optimizer mechanisms such as learning automata, game theory, etc., can be applied to solve this defense problem. As another proposal, this defense can be improved by it merging with other SYN-flooding defense approaches such as attack requests detection and filtering in routers or end systems.

Conclusion

This paper represented a novel approach for defense against SYN-flooding attacks. We used a simple queuing model, to show important metrics of a network under DoS attacks. Then, we mapped the problem of SYN-flooding attack as an optimization problem and then employed PSO technique to solve this problem. We tuned holding time and maximum allowable number of half-open connections parameters, dynamically to achieve high performance for the dynamic conditions of the network by PSO technique. Simulation results confirmed success of the proposed approach.

REFERENCES

Alam L, Ali M, Alam Q, Ali T, Anwar S, Adnan A, Ali M (2011). Mauth: A fine-grained and user-centric permission delegation framework for web services. Int. J. Phys. Sci., pp. 2060-2071.

Al-Bakri SH, Mat Kiah ML, Zaidan AA, Zaidan BB, Alam GM (2011). Securing peer-to-peer mobile communications using public key cryptography: New security strategy. Int. J. Phys. Sci., pp. 930-938.

Bicakci K, Tavli B (2009). Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks. J. Comput. Standards Interf., pp. 931–941.

Chang RKC (2002). Defending against flooding-based distributed denial-of-service attacks: A tutorial. IEEE Commun. Mag., pp. 42-51.

Gordon LA, Loeb MP, Lucyshyn W, Richardson R (2005). 10t**h** annual CSI/FBI computer crime and security survey. Comput. Secur. Inst., pp. 1-26.

Hamdi M, Boudriga N (2007). Detecting Denial-of-Service attacks using the wavelet transform. J. Comput. Commun., pp. 3203–3213.

Kennedy J, Eberhart R (1995). Particle swarm optimization. IEEE Neural Netw., pp. 1942-1948.

Long M, Wu CH, Hung JY (2005). Denial of service attacks on network-based control systems: Impact and mitigation. IEEE Trans. Ind. Inform., pp. 85-96.

Ling Y, Gu Y, Wei G (2009). Detect SYN Flooding Attack in Edge Routers. Int. J. Secur. Appl., pp. 31-46.

Ming Y (2009). A Probabilistic Drop Scheme for Mitigating SYN. Flooding Attacks. International Conference on NSWCTC '09, pp. 734-734.

Nejati F, Khoshbin H (2010). A novel secure and energy-efficient protocol for authentication in wireless sensor networks. Int. J. Phys. Sci., 5(10): 1558–1566.

Peng T, Leckie C, Kotagiri R (2003). Protection from distributed denial of service attack using history-based IP filtering. In Proceedings of the IEEE International Conference on Communications, pp. 482–486.

Philip (2008). Linux TCP/IP parameters reference. ip-sysctl.txt reference for IP networking parameters based on the 2.4 kernel.

Rolls DA, Michailidis G, Hernandez-Campos F (2005). Queuing analysis of network traffic: Methodology and visualization tools. J. Comput. Netw., pp. 447-473.

Safa H, Chouman M, Artail H, Karam M (2008). A collaborative defense mechanism against SYN flooding attacks in IP networks. J. Netw. Comput. Appl., pp. 509-534.

Siris VA, Papagalou F (2006). Application of anomaly detection algorithms for detecting SYN flooding attacks. J. Comput. Commun., pp. 1433-1442.

Wang Y, Lin Ch, Li QL, Fang Y (2007). A queuing analysis for the denial of service (DoS) attacks in computer networks." J. Comput. Netw., pp. 3564–3573.

Wang H, Zhang D, Shin KG (2002a). Detecting SYN flooding attacks. Proceedings of IEEE INFOCOM, pp. 1530–1539.

Wang H, Zhang D, Shin G (2002b). SYN-dog: sniffing SYN flooding sources. In Proceedings of the 22nd International Conference On Distributed Computing Systems (ICDCS'02), pp. 421–428.

Xiao B, Chen W, He Y (2008). An autonomous defense against SYN flooding attacks: Detect and throttle attacks at the victim side independently. J. Parallel Distrib. Computi., pp. 456-470.

Zahiri SH, Seyedin SA (2007). Swarm intelligence based classifiers. Franklin Institute, pp. 362-376.

Zuquete A (2002). Improving the functionality of SYN cookies. In Proceedings of 6th IFIP Communications and Multimedia Security Conference, pp. 57-77.