

DiDDeM: A System for Early Detection of TCP SYN Flood Attacks

J. Haggerty, T. Berry, Q. Shi and M. Merabti

School of Computing and Mathematical Sciences,
Liverpool John Moores University, Liverpool, UK, L3 3AF, UK

Corresponding author: J.Haggerty@livjm.ac.uk

Abstract—This paper presents the *Distributed Denial-of-Service Detection Mechanism (DiDDeM)* system for early detection of denial-of-service attacks. The design requirements of the system are posited to demonstrate the requirements for an early detection system. An overview of the system is presented to show how these requirements are met. DiDDeM provides a two-tier detection approach. First, *pre-filters (PFs)* filter traffic for possible attacks. This is achieved through the application of both stateful and stateless signatures utilising routing congestion algorithms. Second, command and control (C^2) servers provide intra- and inter-domain co-operation and response to contain an attack within the routing infrastructure. The results for stateful and stateless signature detection of TCP SYN flood attacks are presented.

I. INTRODUCTION

Denial-of-service attacks remain a significant problem in today's networks. In a recent survey, 42 per cent of respondents indicated that they had been a victim of such an attack, a rise of 15 per cent on the previous year [1]. This is despite the widespread deployment of perimeter security devices, such as firewalls and intrusion detection Systems (IDS). Denial of service targets the heart of today's information economy, connectivity, by preventing access to services by legitimate users. This may be achieved in a number of ways. However, 94 per cent of attacks utilise TCP to achieve their aim [2].

A number of approaches have been proposed to counter the denial of service problem. These mechanisms include payment for network resources [3], strong authentication [4], Pushback [5], and traffic identification [6]. However, issues such as their inability to scale, differentiate malicious from benign traffic with little overhead, requirement for stateful information, or deal with high-volume flows has ensured that these approaches have not achieved widespread deployment. Therefore, a new approach is required.

This paper presents the design of a novel system, the **D**istributed **D**enial-of-Service **D**etection **M**echanism (DiDDeM), and is organised as follows. Section II discusses related work. Section III presents the requirements for early detection and an overview of the DiDDeM system. Section IV presents signatures for detection of TCP SYN flood attacks and the results of their application. Finally, we make our conclusions.

II. RELATED WORK

TCP SYN flooding remains a problem as it ties up resources on the victim host by subverting the underlying

fundamentals of the TCP protocol. Three principal approaches have been proposed to counter the problem; syncache and syncookies, firewalls, and traffic monitoring. The syncache approach is similar to the existing TCP behaviour, but allocates a much smaller state structure to record the initial connection request. Once the connection is completed, all the resources are then allocated to it [7]. The syncookie approach does not store any state on the machine, but keeps all states regarding initial connections in the network, therefore treating connections as an infinitely deep queue [7]. Firewall approaches posit that with some improvements to existing configurations, such the applying relay and gateway mechanisms, a defence against these attacks can be provided [8, 9]. Traffic monitoring approaches such as [5, 6, 10] monitor network behaviour, and in particular traffic patterns on a network, and if an attack is detected, some form of remedial action is taken. For example, a surge in TCP SYNs or TCP RSTs may indicate an attack [11].

These approaches are not without their problems. For example, the first two approaches do not actually prevent the attack, but attempt to mitigate it. The attack has succeeded as it has already reached its intended target. Traffic monitoring approaches often require state information of the system that they are protecting, or the information required to be held by the monitor within the routing infrastructure is too computationally expensive to be effective.

III. OVERVIEW OF DiDDeM

DiDDeM utilises the two types of signature to meet the requirements for early detection of TCP SYN flood attacks. The three key elements of DiDDeM are: the DiDDeM domain, the pre-filter (PF) detection node, and the command and control (C^2) server. A DiDDeM domain comprises a number of PF nodes and a C^2 server. A PF is a key element in the detection of denial-of-service attacks. A PF is located on a router and utilises the congestion algorithm to infer stateful information from stateless information for detection. A C^2

server manages its DiDDeM domain, communicates with C^2 servers in other DiDDeM domains, provides a central station for attack analysis, and co-ordinates a response to an attack.

A. DiDDeM Design Goals

The DiDDeM design goals provide the requirements of the system. The two principal goals are as follows:

- *Scalability.* Scalability of the system can be measured in two ways [12]. First, in terms of the ability to be deployed within the routing infrastructure, the system must meet the demands of both large LANs and the Internet itself. Second, scalability is measured in terms of the volume of network traffic processed by the system.
- *High-speed, large-volume monitoring.* A large amount of traffic must be considered within very tight temporal constraints within the routing infrastructure. The filtering of a large volume of traffic is achieved by identifying patterns in the TCP/IP headers, which greatly reduces the processing load of monitoring [13, 14].

Other requirements met by DiDDeM include: acceptable performance degradation, inference of stateful information in a stateless environment, and real-time notification and response. These requirements form the focus of the DiDDeM system and are addressed in the remainder of this section.

B. DiDDeM System

If denial-of-service attacks are allowed to reach their intended target, the attack is able to succeed in its objective of denying resources to legitimate users. The objective of the DiDDeM system is to provide early detection and response to denial-of-service attacks before they denigrate the services and resources of the target.

The DiDDeM system integrates co-operative DiDDeM domains. The domain is comprised of two types of element, a C^2 and a number of *PFs*. The C^2 acts as a server to a number of *PFs* located within the domain. The services that the C^2 provides are attack analysis through collation of *PF* reported events, management of the domain, attack response, and authentication of *PFs* within a domain. It is also responsible for intra- and inter-DiDDeM domain communications. A *PF* is responsible for attack detection through stateful and stateless signatures and is located within the routing infrastructure.

The domain allows a two stage detection and analysis process. The first stage of detection is via traffic analysis. First, *PFs* detect surges in network traffic that indicate that a possible denial-of-service attack is under way. A selection of packets are inspected to determine whether the packets form part of an attack. If so, a message is sent to the C^2 server with details of the attack. Second, the C^2 compares this message with its possessed information and that received from C^2 's in

other relevant domains, and then issues a response as required.

The first stage is the detection of possible attacks by the *PFs*. The objective of a *PF* is to identify the stateful signatures of denial-of-service attacks in stateless way, which indicate the possibility that an attack is under way against a particular host, domain, or network. Thus, they detect rises in traffic passing through the router heading in a particular direction, i.e. the target of an attack. If an attack is detected, the alert is then confirmed or discarded by applying stateless signatures to the packets in question.

The stateful detection of attack traffic flows is achieved by interacting with the congestion algorithm used by the router. This algorithm already detects and responds to upsurges in traffic. During periods of heavy traffic, traffic is first queued by the router to control the traffic load on the network. By queuing the traffic, the router is able to implement a volume threshold. Once the threshold is surpassed, it drops packets rather than relay them across an already congested link. Rather than merely drop packets when a queue threshold is surpassed, a *PF* located on a domain ingress router picks packets to be dropped and inspects a statistical sample of those dropped packets to ascertain the direction of traffic flow. If the destination address of all the packets in the statistical sample match, it is likely that they are part of a large flow of traffic towards a destination, providing a stateful signature of a denial-of-service attack. In this way, we can infer unusual rises in traffic against a particular host, domain, or network without holding any state information on those networks.

Once a stateful signature of a denial-of-service attack is inferred, stateless signatures are then applied to a sample of packets to confirm the attack. If an attack is confirmed through the stateless signature, an alert is generated and forwarded to the controlling C^2 . This alert contains details of the reporting *PF*, the destination of the traffic, and the attack signature matched.

Employing a *PF* approach enables the reduction of computational overheads in three ways. First, the *PF* utilises the already existing congestion algorithm for detection. The stateful signature can be inferred by the upsurge in traffic volumes that cause the congestion algorithm to drop packets. Second, only a small number of packets, i.e. those that are dropped, need be inspected by the *PF*. The number of packets during a denial-of-service attack may be high, and to perform signature analysis on each and every packet adds computational overheads. Therefore, the packets are inspected to ascertain direction of flow. Finally, a message reporting the alert is sent to the C^2 rather than redirecting the packets themselves. In traditional IDSS, each packet on the network is compared to a database of signatures, and as many as 300 signatures are applied to each and every packet traversing the network [15]. Every time a signature is

matched with that in the IDS database, an alert is generated. By using a statistical sample of packets, the ratio of packets to alerts can be further enhanced to reduce computational load.

The second stage of detection occurs at the C^2 server overseeing the DiDDeM domain. The C^2 receives the alert message from a subordinate PF. If the attack is confirmed, a response directive, such as blocking all traffic matching a particular signature, is passed to the PF from the C^2 .

To provide holistic security, the C^2 passes information about the attack to other DiDDeM domains. The C^2 identifies adjacent domains joined via ingress filter PFs. Once an alert is received from the PF, the server analyses the information. If an attack is confirmed, the C^2 passes an alert to the C^2 's of adjacent domains. Each such adjacent C^2 is identified by the ingress link of the reporting PFs.

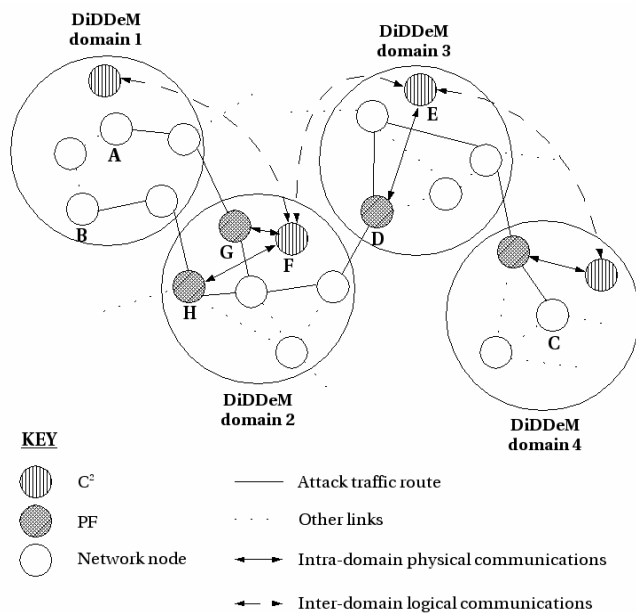


Fig. 1. DiDDeM domain co-operation.

Figure 1 demonstrates the co-operative process among domains during a denial-of-service attack.

1. Nodes *A* and *B* in DiDDeM domain 1 launch a TCP SYN flood as part of a distributed denial-of-service attack against node *C* in DiDDeM domain 4. Due to the network topology, traffic passes through DiDDeM domain 2 undetected as congestion does not occur.
2. As traffic passes into DiDDeM domain 3, a PF, *D*, detects the upsurge in network traffic against one destination.
3. The PF issues an alert to the domain C^2 , *E*. *E* determines the attack and issues a message alerting

adjacent DiDDeM domains 2 and 4. A response is also passed to the reporting PF, *D*.

4. The message is received by the C^2 in DiDDeM domain 2, *F*. *F* compares this information with that received from its own PFs. If an attack is detected within the domain, *F* issues a response to PFs, *G* and *H*, and alerts its adjacent domains. If an attack is not detected, the alert times out after a set time period.

Thus, the attack is traced back to the originating domain of the attack, DiDDeM domain 1, and the attack is contained. It should be noted that trace back is not the aim of DiDDeM, however, it remains a benefit of the inter-domain co-operation.

The DiDDeM design has a number of advantages. First, PFs filter out a large number of packets to be inspected compared to traditional IDS that apply numerous signatures to each packet traversing the network. Second, by using the filtering process, signature analysis and stateful information are achieved statelessly, thus reducing the computational overheads placed on the system, particularly during a denial-of-service attack. Any stateful information required is held on a dedicated server, the C^2 . Third, by utilising congestion algorithms currently employed by the router, packet inspection only occurs during periods of high traffic volume, which is a signature of an attack. Fourth, only one message per signature application is issued by a PF. Due to the statistical sample used, this alert covers a number of attack packets rather than re-directing each and every attack packet inspected to the C^2 , ensuring efficiency of network resources. Fifth, domains co-operate to provide a holistic approach to ensure trace back and containment of the attack source.

IV. SIGNATURE MATCHING

In order to achieve early detection, both stateful and stateless signatures must be utilised. Stateful signatures alert us to unusual traffic loads towards a target host or network, whereas stateless signatures verify that an attack is indeed taking place. Stateless signatures also reduce the number of false positives reported by the DiDDeM system. In traditional IDS, there are no combinations of the two types of signature. Systems are either one type or the other, i.e. misuse-based or anomaly-based IDS. These systems have knowledge of the network under protection: the perimeter is well-defined, security policies are applied, legitimate operations are determined, audit logs are maintained, and there is user accountability. Within the routing infrastructure, we have no such demarcations. Therefore, DiDDeM proposes a novel approach to counter these issues to provide a defence against denial-of-service attacks.

The two main elements of the PF are prototyped. First, stateful signature detection is prototyped using the network simulation tool, *ns2*. This simulation enables an investigation

into the behaviour and effectiveness of a PF within the network environment. Second, once an attack is detected, packets to be dropped are searched for elements of interest to confirm the attack.

A. Stateful Signatures

Congestion occurs within networks. Thus, routers employ congestion algorithms, such as RED [16] or ChOKE [17], to ensure that they do not fail when faced with high levels of traffic. These algorithms may be as simple as employing a *first in, first out* (FIFO) queue. Once the queue maximum limit is reached, packets are dropped in accordance with the congestion algorithm to ensure queue space for further incoming packets. In this way, a level of traffic throughput is maintained.

Within the *stateful signature detection* module, congestion algorithms are adapted for use in the detection of denial-of-service attacks. During these attacks, large volumes of traffic are observed. As illustrated in figures 2 and 3, the control traffic fits the normal probability plot well, whilst the upsurge in traffic caused by a TCP SYN flood is apparent. Rather than purely dropping packets when the router threshold is met, packets that are to be dropped from the queue are inspected by the PF. This enables inference of stateful information about traffic flows and whether these unusual flows are attributable to a particular destination. It is the random inspection that allows the state inference. If two (or more) sampled dropped packets are heading to one destination, they are passed for stateless signature inspection as it indicates the possibility of a large flow of traffic in one direction.

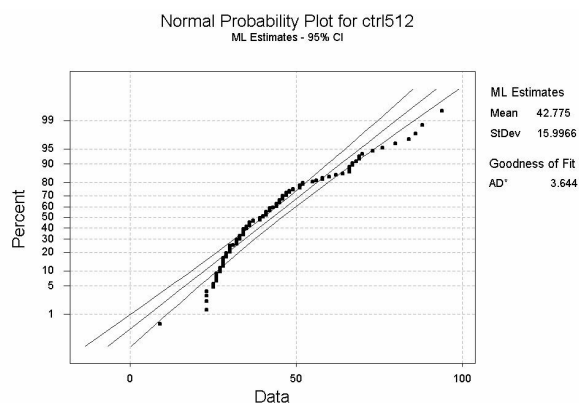


Fig. 2. Probability plot for control traffic on a network.

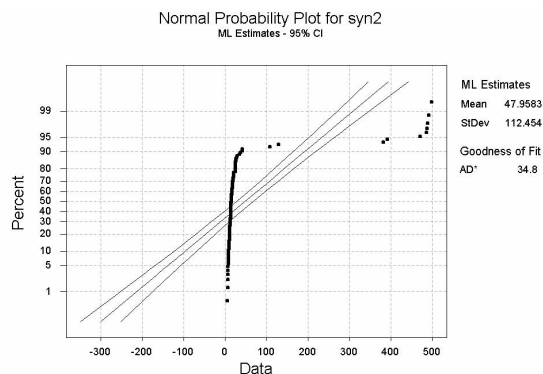


Fig. 3. Probability plot for TCP SYN flood attack traffic on a network.

To demonstrate the way in which this is achieved in the DiDDeM architecture, a *first-in, first-out* (FIFO) queue is used. The available space within the DiDDeM FIFO queue is divided into two sub-queues to allow comparison of packets whilst in the queue. An incoming packet to the router is placed in a queue, if due to bandwidth restrictions the packet cannot be immediately forwarded to the next router. These packets are placed in either the first or second sub-queue at the router based on a first-come, first-served basis.

Packets placed in the queue, and its sub-queues, are dequeued and forwarded to their destination. If the threshold of the total queue limit is exceeded the router begins to drop packets to ensure that packets already in the queue are forwarded and that new incoming packets can be placed in the queue. In this way, no stateful information is held about the queue apart from whether the queue limit has been exceeded, thereby reducing the computational overhead placed on the router.

At periods where congestion occurs, packets are dropped. By meeting the threshold of the particular router which invokes packet dropping, an upsurge in traffic can be inferred. However, this may or may not be due to large amounts of traffic, such as would occur during a denial-of-service attack, directed at a particular victim host or domain. Therefore, prior to packets being dropped, the IP header is accessed and the destination address obtained. This IP destination address is compared to the previous packet's IP destination address. If they are the same, then the IP destination address is stored for comparison with the next packet and the packet is passed to for stateless signature analysis. If the destination addresses are not the same, the destination IP address is still stored for comparison with the next packet, but the packet is dropped. The DiDDeM prototype algorithm is illustrated in figure 4.

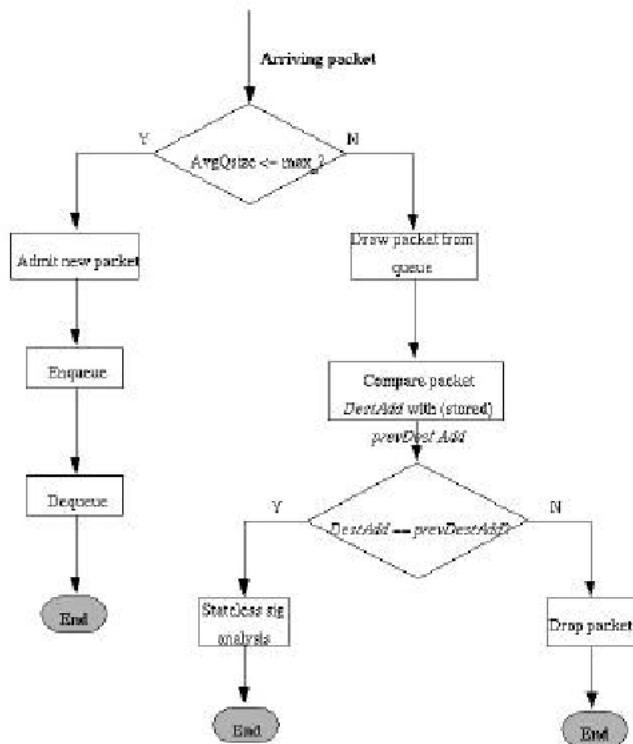


Figure 4. PF algorithm using FIFO queue.

During the simulation, approximately 19,500 attack packets were directed at the victim node by two attacking nodes. This represented an attack consisting of approximately 1,000 packets per second. Once the congestion algorithm was invoked by the router, 798 attack and legitimate packets were to be dropped. Of this number, 742 packets were actual attack packets whilst the remainder were legitimate traffic. Therefore, out of a total of 19,500 attack packets, only 4 per cent of this volume was inspected. DiDDeM detected 697 packets using the algorithm in figure 4. The 697 packets detected out of the 742 inspected by the DiDDeM PF ensures a 94 per cent detection rate. In addition, only two legitimate packets were falsely detected as attack packets, showing a false positive rate of 0.3 per cent.

One key measure for the PF stateful signature matching is its performance within the network environment. In particular, the DiDDeM algorithm should not have an adverse affect on the network, thereby requiring a tradeoff between usability and effectiveness. In order to measure the performance of the DiDDeM PF in the network it is compared to an existing routing algorithm, DropTail, in the network simulator ns2. As illustrated in figures 5 and 6, stateful signature matching has a comparable performance with DropTail and does not impede network performance.

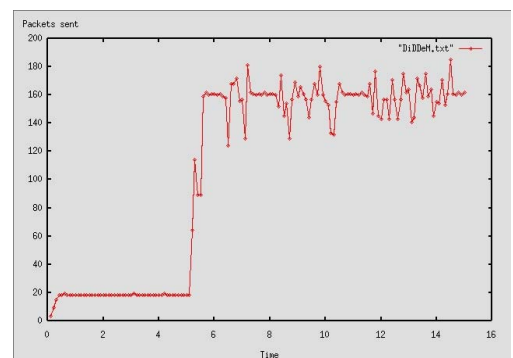


Fig.5. Network packets sent by router enabled with DiDDeM PF.

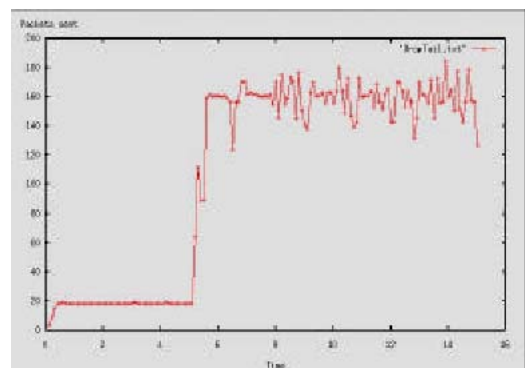


Fig.6. Network packets sent by router enabled with DropTail.

The *stateless signature detection* module receives a sample of packets from the stateful signature detection module.

String-matching techniques are used to match the attack signatures. Not all the information in the header is actually relevant for the detection of a denial-of-service attack; only certain fields are of interest. To search all elements of the stream adds overheads with the search of redundant information. Therefore, the signature detection jumps from one element of interest to the next within the stream to ensure that only the relevant parts of a packet header are read and compared.

For TCP SYN flood detection, the stateless signature detection module searches for instances of SYN flags within the TCP header, which are indicative of an attack. If a flag is found, it is compared with other packets in the stream to ascertain whether its neighbouring packets also have the same flag set. For example, the stream is searched for SYN flags within 3 packets of each other. In experiments, this criteria was applied to 432 packets of which 50 were known

to be SYN flood packets. The 50 attack and 8 benign packets were identified giving a 100 per cent detection rate.

This approach is efficient to meet the demands of high-volume detection. In the example above, the signature search was completed in 0.65 seconds. This process includes the time to read the text into the program, the processing of the stream, and print the results on a Pentium II 400MHz test host. Of this time, the host took 0.64 seconds to read the text into the program, so that the signature match itself actually took 0.01 second.

V. CONCLUSIONS AND FURTHER WORK

Denial-of-service attacks remain a significant problem despite the widespread deployment of perimeter security devices such as firewalls and IDS. This paper presents the DiDDeM system and the signatures required for early detection of TCP SYN flood denial-of-service attacks. This system utilises congestion algorithms already used by routers to filter the volume of traffic inspected and infer stateful information about direction of traffic flows. Stateless signatures are then applied. The results of experiments with DiDDeM against TCP SYN floods are presented and demonstrate the applicability of the system in high-speed, high-volume network environments. Future work concentrates on the application of the system to other denial-of-service attacks and other attacks that require a high volume of traffic such as worms.

REFERENCES

- [1] Richardson, R., '2003 CSI/FBI Computer Crime and Security Survey,' *Computer Security Institute/Federal Bureau of Investigation Technical Report*, 2004, available from www.gocsi.com.
- [2] Moore, D., Voelker, G. M., & Savage, S., 'Inferring Internet Denial-of-Service Activity,' *Proceedings of 10th Usenix Security Symposium*, 2001, Washington, DC.
- [3] Mankins, D., Krishnan, R., Boyd, C., Zao, J., Frentz, M., 'Mitigating Distributed Denial of Service with Dynamic Resource Pricing,' *Proceedings of the Annual Computer Security Applications Conference (ACSAC) 2001*, New Orleans, Louisiana, USA, 2001.
- [4] Meadows, C., 'A cost-based framework for analysis of denial of service in networks,' *Journal of Computer Security*, **9**, pp. 143-164, 2001.
- [5] Ioannidis, J., Bellovin, S. M., 'Implementing Pushback: Router-based Defense Against DDoS Attacks,' *Proceedings of the Network and Distributed Systems Security Symposium*, San Diego, CA, USA, 2002.
- [6] Kuzmanovic, A., Knightly, E. W., 'Low-Rate TCP-Targeted Denial of Service Attacks,' *Proceedings of SIGCOMM 03*, Karlsruhe, Germany, 2003.
- [7] Lemon, J., 'Resisting SYN floods DoS attacks with a SYN cache,' *Proceedings of BSDCon 2002*, Berkeley, CA, USA, 2002.
- [8] Schuba, C. L., Krsul, I. V., Kuhn, M. G., Spafford, E. H., Sundaram, A., Zamboni, D., 'Analysis of a Denial of Service Attack on TCP,' *Proceedings of IEEE Symposium on Security and Privacy*, Oakland, CA, USA, 1997.
- [9] Mutaf, P., 'Defending Against a Denial-of-Service Attack on TCP,' *Proceedings of RAID '99*, 1999.
- [10] Huang, Y., Pullen, J. M., 'Countering Denial-of-Service Attacks Using Congestion Triggered Packet Sampling and Filtering,' in *Proceedings of the 10th International Conference on Computer Communication and Networks*, Scottsdale, AZ, USA, 2001.
- [11] Mansfield, G., Ohta, K., Takei, Y., Kato, N., Nemoto, Y., 'Towards Trapping Wily Intruders in the Large,' *Computer Networks*, **34**(4), p. 659-670, 2000.
- [12] Krugel, C., Toth, T., 'Distributed Pattern Detection for Intrusion Detection,' *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, 2002.
- [13] Zhang, Y., Paxson, V., 'Detecting Backdoors' *Proceedings of the 9th USENIX Security Symposium*, Denver, Colorado, USA, 2000.
- [14] Hussain, A., Heidemann, J., Papadopoulos, C., 'A Framework for Classifying Denial of Service Attacks,' *Proceedings of SIGCOMM 03*, Karlsruhe, Germany, 2003.
- [15] Coit, C.J., Staniford, S., McAleney, J., 'Towards Faster String Matching for Intrusion Detection or Exceeding the Speed of Snort,' *Proceedings of DARPA Information Survivability Conference and Exposition (DISCEX 01)*, Anaheim, CA, USA, 2001.
- [16] Floyd, S., Jacobson, V., 'Random Early Detection Gateways for Congestion Avoidance,' *IEEE/ACM Transactions on Networking*, **3**(4), pp. 365-386, 1993.
- [17] Pan, R., Prabhakar, B., Psounis, K., 'CHOCk: A Stateless Active Queue Management Scheme for Approximating Fair Bandwidth Allocation', *Proceedings of IEEE Infocomm*, Tel Aviv, Israel, 2000.