Proactively Detecting Distributed Denial of Service Attacks Using Source IP Address Monitoring

Tao Peng, Christopher Leckie, and Kotagiri Ramamohanarao

ARC Special Research Center for Ultra-Broadband Information Networks
Department of Computer Science and Software Engineering
The University of Melbourne
Victoria 3010, Australia
{tpeng, caleckie, rao}@cs.mu.oz.au

Abstract. In this paper, we propose a simple but robust scheme to detect denial of service attacks (including distributed denial of service attacks) by monitoring the increase of new IP addresses. Unlike previous proposals for bandwidth attack detection schemes which are based on monitoring the traffic volume, our scheme is very effective for highly distributed denial of service attacks. Our scheme exploits an inherent feature of DDoS attacks, which makes it hard for the attacker to counter this detection scheme by changing their attack signature. Our scheme uses a sequential nonparametric change point detection method to improve the detection accuracy without requiring a detailed model of normal and attack traffic. Furthermore, we show that with the combination of monitoring per flow speed, we can detect all types of DDoS attacks. We demonstrate that we can achieve high detection accuracy on a range of different network packet traces.

1 Introduction

A denial-of-service (DoS) attack is a malicious attempt by a single person or a group of people to cripple an online service. The impact of these attacks can vary from minor inconvenience to users of a website, to serious financial losses for companies that rely on their on-line availability to do business. Sophisticated tools to gain root access to other people's machines are freely available on the Internet. These tools are easy to use, even for unskilled users. Once a machine is cracked, it is turned into a "zombie" under the control of one "master". The master is operated by the attacker. The attacker can instruct all its zombies to send bogus data to one particular destination. Simultaneously, the resulting traffic can clog links, and cause routers near the victim or the victim itself to fail under the load. The type of DoS attack that causes problems by overloading the victim with useless traffic is known as a *bandwidth attack*. This paper focuses on curtailing bandwidth attacks.

A key problem to tackle when solving bandwidth attacks is *attack detection*. However, there are two challenges for detecting bandwidth attacks. The first

challenge is how to detect malicious traffic close to its source. This is particularly difficult when the attack is highly distributed, since the attack traffic from each source may be small compared to the normal background traffic. The second challenge is to detect the bandwidth attack as soon as possible without raising a false alarm, so that the victim has more time to take action against the attacker.

Previously proposed approaches rely on monitoring the volume of traffic that is received by the victim [10][16][2]. A major drawback of these approaches is that they do not provide a way to differentiate flash crowds from DDoS attacks. Due to the inherently bursty nature of Internet traffic, a sudden increase of in traffic may be mistaken as an attack. If we delay our response in order to ensure that the traffic increase is not just a transient burst, then we risk allowing the victim to be overwhelmed by a real attack. Moreover, some persistent increases in traffic may not be attacks, but actually "flash crowd" events, where a large number of legitimate users access the same website simultaneously. Clearly, there is a need for a better approach to detecting bandwidth attacks.

A better approach is to monitor the number of new source IP addresses, rather than the local traffic volume. Jung et al. [8] have observed that during bandwidth attacks, most source IP addresses are new to the victim, whereas most source IP addresses in a flash crowd appeared at the victim before. Previously, this observation has been used as the basis for a mechanism to filter out attack traffic at the victim [12]. In this paper, we propose to monitor the number of new IP addresses in a given time period in order to detect bandwidth attacks. We demonstrate that this is a more sensitive variable for detecting bandwidth attacks than monitoring the total volume of incoming traffic. In addition, we present a method for detecting changes in our monitoring variable, based on the non-parametric Cumulative Sum (CUSUM) algorithm [3][15]. The CUSUM algorithm reduces the false positive rate, and has been shown to optimal in terms of detection accuracy and computing overhead for parametric model and have good performance for non-parametric model [3].

Our main contribution in this paper is a novel approach to detecting bandwidth attacks by monitoring the arrival rate of new source IP addresses. We show that this approach is much more effective than earlier schemes, especially when there are multiple attack sources and the attack traffic is highly distributed. We adapt the detection scheme proposed by Wang et al. [15], which is based on an advanced non-parametric change detection scheme, CUSUM, and demonstrate that this approach detects a wide range of simulated attacks quickly and with high accuracy.

The rest of the paper is organized as follows. Section 2 gives a detailed explanation of our solution to this problem. Section 3 explains CUSUM algorithm and the model we proposed for the bandwidth attack detection. Section 4 presents the simulation results of our detection mechanism. Section 5 analyzes possible attacks against our detection mechanism.

2 Our Solution: Source IP Address Monitoring

We propose a scheme called Source IP address Monitoring (SIM) to detect the Highly Distributed Denial of Service (HDDoS) attacks. This detection scheme uses an intrinsic feature of HDDoS attacks, namely the huge number of new IP addresses in the attack traffic to the victim. This novel approach has the advantage that it can detect attacks close to their sources in the early stages of the attack.

2.1 Overview of Source IP Address Monitoring

SIM contains two parts: *off-line training*, and *detection and learning*. The first part is the *off-line training*, where a learning engine adds legitimate IP addresses into an IP Address Database (IAD) and keeps the IAD updated by adding new legitimate IP addresses and deleting expired IP addresses. This is done off-line to make sure the traffic data used for training does not contain any bandwidth attacks. A simple rule can be used to decide whether a new IP address is legitimate or not. For example, a TCP connection with less than 3 packets is considered to be an abnormal IP flow. How to build an efficient IAD is discussed in detail in [12].

The second part is *detection and learning*. During this period, we collect several statistics of incoming traffic for the current time interval Δ_n. By comparing the IP addresses during Δ_n with the IAD, we can calculate how many new IP addresses have appeared in this time slot. If the rate per IP address is larger than a certain threshold, an alarm is set to indicate a bandwidth attack. This is used to detect some unsophisticated attacks that use a small number of source IP addresses. More importantly, by analyzing the number of new IP addresses, we can detect whether a HDDoS attack is occurring. If an attack is detected, the on-line-learning is suspended. Otherwise, on-line-learning proceeds in the same matter as off-line training.

2.2 The choice of a detection feature

The key aspect of our detection scheme is that we choose a completely new detection feature compared to earlier detection proposals. We collect the IP addresses during each time slot Δ_n (*n=1, 2, 3, ...*), which determines the detection resolution. We assume $\Delta_1 = \Delta_2 = ... = \Delta_n$, which means the time slots are of equal length. The choice of Δ_n is a compromise between making Δ_n small so that the detection engine can quickly detect an attack, and making Δ_n large so that the detection engine has less computation load because it checks the traffic less often.

Let $\mathcal{T}_n$ represent the set of unique IP addresses and $\mathcal{D}_n$ represent the items of IP Address Database (IAD) at the end of the time interval Δ_n $(n = 1, 2, 3, ...)$. As we discussed before, $|\mathcal{T}_n - \mathcal{T}_n \cap \mathcal{D}_n|$,which represents the number of new IP addresses in Δ_n, can be used to detect the DDoS attack. However, $|\mathcal{T}_n - \mathcal{T}_n \cap \mathcal{D}_n|$ varies according to the position of the network traffic monitoring point (NTMP)

and different Δ_n. We can normalize this value by defining $X_n = \frac{|\mathcal{T}_n - \mathcal{T}_n \cap \mathcal{D}_n|}{\mathcal{T}_n}$, which will not be affected by the NTMP and Δ_n. Consequently, we use X_n for our detection mechanism.

2.3 Implementation of Our Source IP Address Monitoring (SIM) Scheme

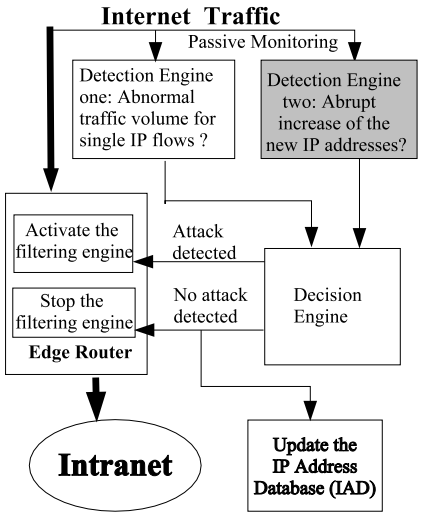


Fig. 1. DDoS attack detection and history-based IP source address filtering

Figure 1 provides an overview of our SIM scheme. The SIM scheme consists of three parts: *detection engine*, *decision engine*, and *filtering engine*. The *detection engine* analyzes the incoming traffic pattern to detect any abnormalities. The *decision engine* summarizes the results from the detection engine and decides whether an attack is occurring. The *filtering engine* filters the attack traffic according to the identified attack traffic pattern. Note that there are two detection engines. The first detection engine is used to detect non-distributed attacks from a single source, while the second detection engine is used to detect highly distributed denial of service attacks.

There are two steps in the detection engines. First, the detection engine sorts the incoming IP flows according to source IP addresses, and identifies whether there is an IP flow with an unusually large number of packets. If there is, we activate the *filtering engine* to block this abnormal IP flow. This step is very effective for defending against some naive DoS attacks launched from a single or small number of sources. The second step is the core technology of our SIM scheme, which is shown in the shadow part of Figure 1. This step is designed to defend against sophisticated DDoS attacks and is described in detail in the

following sections. As we can see from Figure 1, the detection engine monitors the traffic through a passive (read-only) interface which is pre-configured with a non-routable IP address. This implementation feature can make the detection engine immune to the attacks since it is invisible to the attacker. When no attack is detected in the detection engine, a control signal is sent to the edge router [1] to stop the *filtering engine*.

3 Abrupt Change Detection

In order to detect a DDoS attack, we need to be able to detect changes in our detection feature over time. However, our detection feature is a random variable due to to the stochastic nature of Internet traffic. Consequently, before describing the proposed flooding detection mechanism, we discuss the details of the theoretical background of our detection algorithm.

3.1 Change Detection Modelling

Internet traffic can be viewed as a complex stochastic model and any traffic abnormalities, for example, a HDDoS attack, can lead the abrupt change of the model. Our goal is to detect the change in the number of new IP addresses. There are two approaches to detect this change. One is *fixed-size batch detection*, which monitors the change of mean value every fixed time period. Another is *sequential change-point detection*, which monitors the variables successively. The latter is designed to detect a change in the model as soon as possible after its occurrence, which meets the key design requirement for our detection engine. Thus, we can model our task as a sequential change point detection problem. Consider the illustrative example in Figure 3. For the random sequence $\{X_n\}$, there is a step change of the mean value at m from α to $\alpha + h$. We require an algorithm to detect changes of at least step size h and estimate m in a sequential manner so that the detection delay and false positive rate are both minimized. The random sequence $\{X_n\}$ can be formalized as follows:

$$X_n = \alpha + \xi_n I(n < m) + (h + \eta_n) I(n \geq m), \tag{1}$$

where $\xi = \{\xi_n\}_{n=1}^{\infty}$, $\eta = \{\eta_n\}_{n=1}^{\infty}$ are random sequences such that $E(\xi_n) = E(\eta_n) \equiv 0, h \neq 0$. $I(\mathcal{H})$ is the indicator function, it equals "1" when the condition $\mathcal{H}$ is satisfied and "0" otherwise.

3.2 The CUSUM Algorithm

The CUSUM (Cumulative Sum) algorithm is a commonly used algorithm in statistical process control, which can detect the change of mean value of a statistical

[1] We use the term *edge router* to refer to the router that provides access to the Internet for the victim's subnetwork that we are defending.

process. CUSUM relies on the fact that if a change occurs, the probability distribution of the random sequence will also change. Generally, CUSUM requires a parametric model for the random sequence so that the probability density function can be applied to monitor the sequence. Unfortunately, the Internet is a very dynamic and complicated entity, and the theoretical construction of Internet traffic models is a complex open problem, which is beyond the scope of this paper. Thus, a key challenge is how to model $\{X_n\}$. Since non-parametric methods are not model-specific, they are more suitable for analyzing the Internet. In our experiment, we applied the non-parametric CUSUM (Cumulative Sum) method [3] in our detection algorithm. This general approach is based on the model presented in Wang et al. [15] for attack detection using CUSUM. The main idea behind the non-parametric CUSUM algorithm is that we accumulate values of X_n that are significantly higher than the mean level under normal operation. One of the advantages of this algorithm is that it monitors the input random variables in a sequential manner so that real-time detection is achieved.

Let us begin by defining our notation before we give a formal definition of our algorithm. As we mentioned in Sec 2.2, X_n represents the fraction of new IP addresses in the measurement interval Δ_n. The top graph in Figure 3 shows an illustrative example of $\{X_n\}$. In normal operation, this fraction will be close to 0, i.e. $E(X_n) = \alpha \ll 1$, since there is only a small proportion of IP addresses that are new to the network under normal conditions [8] [12]. However, one of the assumptions for the nonparametric CUSUM algorithm [3] is that mean value of the random sequence is negative during normal conditions, and becomes positive when a change occurs. Thus, without loss of any statistical feature, $\{X_n\}$ is transformed into another random sequence $\{Z_n\}$ with negative mean a, i.e. $Z_n = X_n - \beta$, where $a = \alpha - \beta$ (See the middle graph of Figure 3). Parameter β is a constant value for a given network condition, and it helps to produce a random sequence $\{Z_n\}$ with a negative mean so that all the negative values of $\{Z_n\}$ will not accumulate according to time. When an attack happens, Z_n will suddenly become large and positive, i.e. $h + a > 0$, where h can be viewed as a lower bound of the increase in Z_n during an attack. Hence, Z_n with a positive value ($h + a > 0$) is accumulated to indicate whether an attack happens or not (See the bottom graph of Figure 3). One thing worth noting is that h is defined as the minimum increase of the mean value during an attack and it is not the threshold for the bandwidth attack detection. The attack detection threshold N is used for the y_n, accumulated positive values of Z_n, which is illustrated in Figure 3. Our change detection is based on the observation of $h \gg \beta$. Now our detection problem is to find the abrupt change in the random sequence $\{Z_n\}$ which is described as follows:

$$Z_n = a + \xi_n I(n < m) + (h + \eta_n) I(n \geq m), \tag{2}$$

where $a < 0$, $-a < h < 1$, and other conditions are the same as Eq. 1.

The formal definition of the non-parametric CUSUM algorithm is illustrated as follows:

$$y_n = S_n - \min_{1 \leq k \leq n} S_k, \tag{3}$$

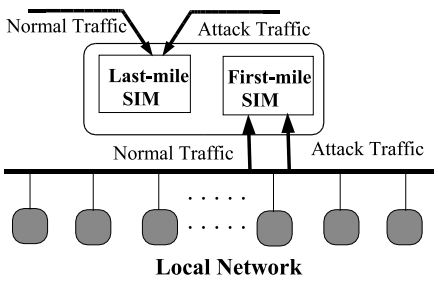


Fig. 2. The trace-driven simulation experi-t ment

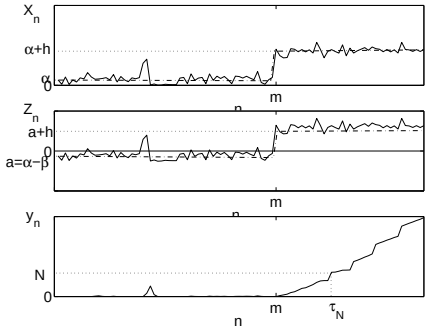


Fig. 3. The CUSUM algorithm

where $S_k = \sum_{i=1}^{k} Z_i$, with $S_0 = 0$ at the beginning, and y_n is our test statistic. In order to reduce the overhead for online implementation, we use the recursive version of non-parametric CUSUM algorithm [1][3][2][15] which is shown as follows:

$$y_n = (y_{n-1} + Z_n)^+, \qquad y_0 = 0, \tag{4}$$

where x^+ is equal to x if $x > 0$ and 0 otherwise. A large y_n is a strong indication of an attack.

As we see in the bottom graph of Figure 3, y_n represents the cumulative positive values of Z_n. We consider the change to have occurred at time τ_N if $y_{\tau_N} \geq N$. The decision function can be described as follows:

$$d_N(y_n) = \begin{cases} 0 \text{ if } y_n \leq N; \\ 1 \text{ if } y_n > N. \end{cases}$$

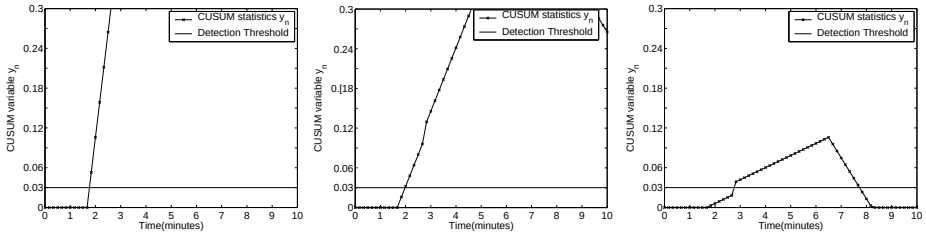
N is the threshold for attack detection and $d_N(y_n)$ represents the decision at time n: '1' if the test statistic y_n is larger than N, which indicates an attack, and '0' otherwise, which indicates the normal operation (no statistical feature change for the random sequence $\{Z_n\}$).

4 Performance Evaluation

The CUSUM algorithm detects changes based on the cumulative effect of the changes made in the random sequence instead of using a single threshold to check every variable. Therefore, with the deployment of the CUSUM algorithm, the performance of our detection scheme will not be affected by whether the attack rate is bursty or constant. To evaluate the efficacy of our detection scheme SIM, we conducted the following simulation experiments. As shown in Figure 2, we created different types of DDoS attack traffic and merged them with the normal

traffic. SIM was then applied to detect the attacks from the merged traffic. The normal traffic traces used in our study are collected at different times from two different sources. The first set was gathered at the University of Auckland [7] with an OC3 (155.52 Mbps) Internet access link [6]. The second data trace was taken on a 9 MBit/sec Internet Connection in Bell Labs [13].

We use the first-mile SIM to monitor the traffic coming into the target network, and last-mile SIM to monitor the traffic going out from the target network. Hence, the outgoing traffic data traces of the University of Auckland can be used evaluate the performance of the first-mile SIM while the incoming traffic data traces can be used evaluate the performance of the last-mile SIM. For the simplicity of the experiment design, we assume the attack traffic rate to be constant. The attack period is set to be 5 minutes, which is a commonly observed attack period in the Internet [11]. The attack traffic rate for all the simulated DDoS attacks is set to be 1 Mbps. Since the network we are defending has the connection capacity of 155.2 Mbps and the average peak connection speed of about 6 Mbps, we define 1 Mbps as the minimum traffic rate to disrupt the network services. We set this conservative attack traffic rate, and aim to test the *detection sensitivity* of the SIM. Attack traffic with higher traffic volume should be easier to detect, and hence is not covered by our performance evaluation.



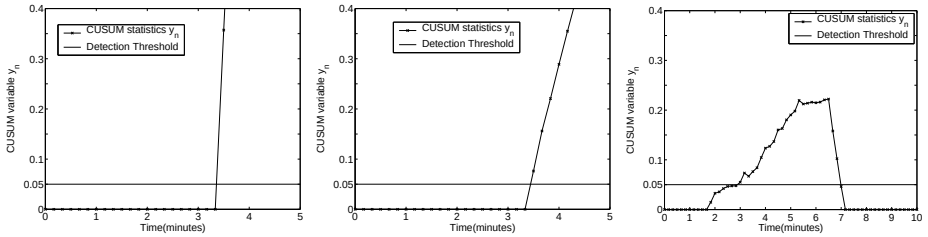
(a)10 new IP addresses (b)4 new IP addresses (c)2 new IP addresses

Fig. 4. The DDoS attack detection sensitivity in the first-mile router using the Auck-IV-out trace

4.1 DDoS Detection Using Detection Engine Two

In an attempt to avoid detection by our scheme, attackers may try to constrain the number of spoofed IP addresses that they use. Let $\mathcal{W}$ represent the number IP addresses in the attack traffic which are new to the network. We tested different values of $\mathcal{W}$ in our simulation, and the detection performance for the first and last-mile routers are shown in Figure 4 and Figure 5 respectively. We repeated the attack detection under a variety of different network conditions, and listed both the average detection accuracy and detection time in Table 1.

As we can see from the simulation results, our detection algorithm is very robust in both the first-mile and last-mile routers. For the last-mile router, we



(a)200 new IP addresses (b)40 new IP addresses (c)18 new IP addresses

Fig. 5. The DDoS attack detection sensitivity for the last-mile router using the Auck-IV-in trace

can detect the DDoS attack with $\mathcal{W} = 18$ within 81.1 seconds with 100% accuracy, and detect the DDoS attack with $\mathcal{W} = 15$ within 127.3 seconds with 90% accuracy. Given the attack traffic length is no more than 5 minutes, only the attack traffic with $\mathcal{W} < 18$ has the possibility of sometimes avoiding our detection. However, by forcing the attacker to use a small number of new IP addresses, we can detect the attack by observing the abrupt change of the number of packets per IP source address using the first detection engine which is described in Sec. 2.3.

For the first-mile router, we can achieve 99% detection accuracy even when there are only 2 new IP address in the attack traffic. The reason lies in the fact that the background traffic for the first-mile router is very clear. Generally, there will be very few IP addresses that are new to the network since all the valid IP packets originated from within the same network. Since the IP addresses in the *IP Address Database* (IAD) will expire and be removed after a certain time period, the IP addresses within the subnetworks which have not been used recently will be new to IAD. This is very similar to ingress filtering [5]. However, ingress filtering cannot detect the attack when the spoofed IP addresses are within the subnetworks. In contrast, our first-mile router detection algorithm can detect the spoofed IP addresses within the subnetworks if they are new to the IAD.

It is worth noting that we choose our detection interval $\Delta_n = 10s$ in our experiment, which is a conservative choice for a real implementation. If we decrease the detection interval by using more computing resources, we can reduce the detection time accordingly.

4.2 False Positives and Implementation Overhead

We define a *false positive* as an attack that is reported by the SIM during normal network operation. We use the following method to evaluate the false positive rate of the SIM. We use the data traces collected at the University of Auckland and Bell labs as the normal traffic input to the SIM. If any attack is detected, then a false positive is generated. We randomly choose 20 one-hour-worth normal

Table 1. Detection Performance of the first-mile router and the last-mile router

The first-mile router			The last-mile router		
W	Detection Accuracy	Detection Time	W	Detection Accuracy	Detection Time
2	99%	69.7s	15	90%	127.3s
4	100%	20.1s	18	100%	81.1s
6	100%	18.9s	40	100%	18.9s
8	100%	10s	60	100%	10s
10	100%	10s	200	100%	10s

traffic snapshots as inputs to the SIM. Each experiment runs independently and no false positive is found.

Our trace-driven experiments were run on a Linux machine with dual 900MHz Xeon and 512 MB RAM. We can achieve an average throughput of 10 Gbps. Therefore, the SIM will not be a bottleneck for network implementation. It is worth noting that we use a two-weeks of data traces to build the IAD due to the short of publicly available data traces. In practice, if we can build the IAD using traces of a longer period, we can expect better detection and reaction performances.

5 Discussion

5.1 Possible Attacks against the SIM

If the attackers know that the SIM is based on previous network connections, they could mislead the server to be included in the IP address database. For example, they can first use a certain group of IP addresses to do some reconnaissance before the real attack. The attackers can control the reconnaissance traffic to be sufficiently low so as not to trigger the IP packet filtering process. If the server considers the reconnaissance traffic to be part of the normal traffic, it will add the attacker's reconnaissance IP addresses into the IP address database. Therefore, the attacker can use the IP addresses which they used before to launch the DDoS attack. Since these IP addresses appear in the IP address database, the attack traffic can pass the filter easily, which constitutes a successful denial-of-service attack.

We can prevent this by increasing the period over which IP addresses must appear in order to be considered frequent. Moreover, we can randomize the learning time for the IAD and keep it secret to the attacker. Furthermore, we can ensure that we only include an IP address in our database if it has successfully completed a TCP connection. This prevents the attacker from using spoofed IP addresses for which no host exists. The attacker can only launch their attack using the real IP address of their computer, which makes it much easier to identify and block the source of the attack. We may also be able to use techniques from our previous work on scan detection [9] in order to identify IP addresses with unusual patterns of accesses. Moreover, we can combine additional rules for

defining frequent IP addresses in order to improve the accuracy of the SIM. For example, the type of service accessed by the user and the length of each session may be useful measures for identifying frequent IP addresses.

5.2 Other Related Issues

With the deployment of Network Address Translation (NAT), Dynamic Host Configuration Protocol (DHCP) and proxy services, multiple users can share the same source IP address. Moreover, the source IP address can still represent some level of identity, for example, a group of users with geographically proximity. Since the IP addresses in our experimental data traces have been sanitized using one-to-one hash mapping, the network information in the IP address is lost. In practice, we can use network addresses, for example, a class C network address, to represent the user's identity. Moreover, the increasing implementation of IP.v6 [4] will strengthen the correlation between source IP address and user identity.

As high profile websites, such as Yahoo and CNN, will have visitors from all around the world, maintaining the IAD is a very challenging task. Fortunately, the deployment of Content Distribution Network (CDN) [14] has limited the users to their local CDN server. Hence, the users for each CDN server will keep consistent and we can build an IAD for each local CDN server separately.

6 Conclusion and Future Work

In this paper we proposed a scheme to detect distributed denial of service attacks by monitoring the increase of new IP addresses. We have also presented a sequential change point detection algorithm that can identify when an attack has occurred. We demonstrated the efficiency and robustness of this scheme by using trace-driven simulations. The experimental results in the Auckland traces show that we can detect DDoS attacks with 100% accuracy using as few as 18 new IP addresses in the last-mile router and DDoS attacks using as few as 2 new IP address in the first-mile router. Our online detection algorithm is fast and has a very low computing overhead. Our first-mile SIM has the advantage over ingress filtering [5] that it can detect attack traffic with spoofed source IP addresses within the subnetworks. Further, with the combination of two detection engines, all the DDoS attacks can be detected. Our future work will include combining other network traffic statistics to detect bandwidth attacks and using distributed detection to detect DDoS attacks.

Acknowledgement

We would like to thank the Waikato Applied Network Dynamics Research Group, and the Internet Traffic Research group in Bell Labs for making available their data traces. This work is funded by Australia Research Council.

References

1. M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes: Theory and Application*. Prentice Hall, 1993.
2. Rudolf B. Blažek, Hongjoong Kim, Boris Rozovskii, and Alexander Tartakovsky. A novel approach to detection of "denial-of-service" attacks via adaptive sequential and batch-sequential change-point detection methods. In *Proceedings of IEEE Systems, Man and Cybernetics Information Assurance Workshop*, June 2001.
3. B. E. Brodsky and B. S. Darkhovsky. *Nonparametric Methods in Change-point Problems*. Kluwer Academic Publishers, 1993.
4. S. Deering and R. Hinden. Internet protocol, version 6 (IPv6) specification. RFC 2460, December 1998.
5. P. Ferguson and D. Senie. *Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing*. RFC2267, IETF, January 1998.
6. Waikato Applied Network Dynamic Research Group. http://wand.cs.waikato.ac.nz/wand/wits/auck/4/.
7. Waikato Applied Network Dynamics Research Group. Auckland university data traces. http://wand.cs.waikato.ac.nz/wand/wits/.
8. Jaeyeon Jung, Balachander Krishnamurthy, and Michael Rabinovich. Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites. *Proceeding of 11th World Wide Web Conference*, May 2002. Honolulu, Hawaii, USA.
9. C. Leckie and R. Kotagiri. A probabilistic approach to detecting network scans. In *Proceedings of Eighth IEEE Network Operations and Management Symposium (NOMS 2002)*, Florence, Italy, 15-19 April 2002.
10. Ratul Mahajan, Steven M. Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker. Controlling high bandwidth aggregates in the network. Technical report, AT&T Center for Internet Research at ICSI (ACIRI) and AT&T Labs Research, February 2001.
11. David Moore, Geoffrey M. Voeker, and Stefan Savage. Inferring Internet Denial-of-Service activity. In *Proceedings of the USENIX Security Symposium*, pages 9–22, August 2001.
12. Tao Peng, Christopher Leckie, and Kotagiri Ramamohanarao. Prevention from distributed denial of service attacks using history-based IP filtering. In *Proceeding of ICC 2003*, Anchorage, Alaska, USA, May 2003.
13. NLANR PMA and the Internet Traffic Research group. Bell Labs - I data set. http://pma.nlanr.net/Traces/long/bell1.html.
14. Dinesh C. Verma, Seraphin Calo, and Khalil Amiri. Policy-based management of content distribution networks. *IEEE Network*, 16(2):34 –39, March-April 2002.
15. Haining Wang, Danlu Zhang, and Kang G. Shin. Detecting SYN flooding attacks. In *Proceedings of IEEE Infocom'2002*, June 2002.
16. David K. Y. Yau, John C. S. Lui, and Feng Liang. Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles. In *Proceedings of IEEE International Workshop on Quality of Service (IWQoS)*, Miami Beach, FL, May 2002.