DDoS Attack Detection Using Unique Source IP Deviation

Ram Charan Baishya, Nazrul Hoque, and Dhruba Kumar Bhattacharyya
(Corresponding author: Dhruba Kumar Bhattacharyya)

Department of Computer Science and Engineering, Tezpur University
Tezpur University, Napaam, Sonitpur, Assam-784028, India
(Email: dkb@tezu.ernet.in)

(Received Apr. 19, 2016; revised and accepted July 19 & Aug. 29, 2016)

Abstract

In this paper we present a low cost yet robust DDoS detection method to identify all classes of DDoS attacks. Our method attempts to detect DDoS attack by monitoring the deviation of the count of unique source IPs and the count of source IPs whose transmission rate is higher than a given threshold value. Unlike other similar existing methods, our method does not need to maintain a list of source IPs which makes our detection method faster. Another advantage of our method is the ability to detect attack performed by small size bot net. In case of such an attack the packet rate of the attack sources deviate from its mean value significantly and thus we can detect this change. We use a non-parametric change point modeling technique to identify flooding attacks of all types in real time. An other contribution of this work is the development of an attack took referred to as TU-CANNON, to generate different variations of DDoS attack under a controlled test-bed environment.

Keywords: DDoS; DDoS Attack Detection; Low-Rate DDoS Attack; DDoS Attack Tool

1 Introduction

In a distributed denial of service (DDoS) attack, one or more group of compromised users send legitimate traffic to the victim to degrade or even shut down the service of the victim. The goal of such a DDoS attack typically varies from creating simple inconvenience to the user of a website, to incur major financial losses to the on-line service providers. Several such incidents can be found in history [10, 11], which show the great threat of DDoS attack to the Internet service providers as well as the Internet users. Today, our day to day activities are becoming more and more dependent on the Internet, starting from e-shopping to e-banking. Most emergency services are also dependent on the Internet. Hence, the impact of a DDoS attack is becoming more threatening. The availability of large no of DDoS attack tools in the public domain has made it very easy to launch such attack with various levels of intensities, even for unskilled users with malicious intention [2, 21]. Such tools are well equipped with automatic scanning of the vulnerable machines over the Internet, exploitation and deployment of attack code in those machines and then performing the actual attack. A detailed description of DDoS attacks and their classifications can be found in [28].

The design goal of TCP/IP was to deliver packets from source to destination in a fast and accurate way. The payload of the packets are of little concern to TCP/IP. A DDoS attack generates a huge volume of TCP/IP packets from a large number of sources. These attack packets are generally indistinguishable from that of normal traffic packets. Thus when all these attack packets merge at the victims site, they occupy most of the victim's network bandwidth and forces the victim to degrade its service or at worst shut down its services temporarily. Typically a DDoS attack is launched from different sources in a coordinated manner and the attack traffic from individual source is comparatively low which make it difficult to distinguish from normal traffic.

Most defense solutions [13, 24, 25] to DDoS attack detection have been found less effective due to the distributive nature of the attack.

Thus we consider two key challenges for DDoS attack defenders. First, how to detect the attack as close as possible to the sources, so that the traffic from such source can be blocked early. Second, to detect the DDoS attack in the victim site as early as possible so that the victim gets enough time to take appropriate action to such an attack.

One obvious way taken by most researchers [5, 15, 27] is to monitor the volume of traffic that are received by the victim site.

However, such methods are not robust against the bursty nature of Internet traffic. In case of the bursty nature of internet traffic such methods often identify it as an attack. On the other hand, such bursty traffic may ac-

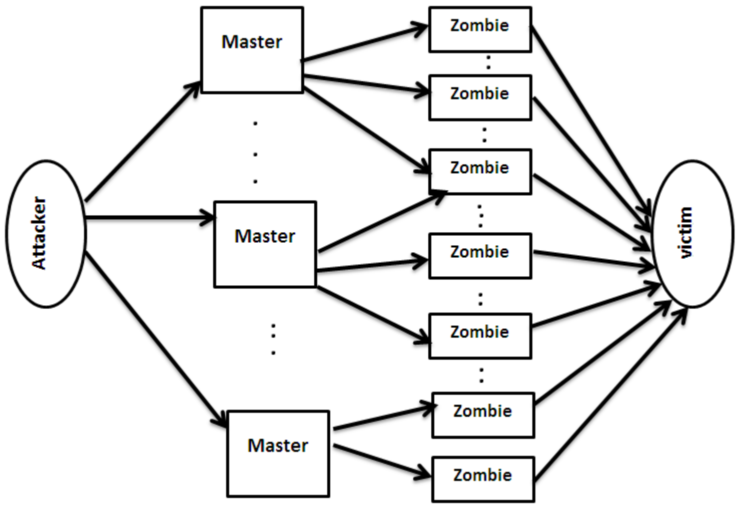


Figure 1: The main elements of a DDoS attack

tually be attack traffic, and a delayed decision may turn out to be very risky for the victim. Since a DDoS attack is highly distributed, the number of source IPs involved in the attack is much larger than the number of source IPs under normal condition. Peng et. al. [22] used the arrival rate of new source IP addresses in the traffic during each observation period. However, this approach needs to maintain a database of trustworthy source IP addresses, which might itself be vulnerable to attack. In this paper, we propose a mechanism that monitors the number of unique source IP addresses during each observation period. Our assumption is that during an attack, the number of source IP will increase abruptly, and by detecting this change we can detect the attack. Also, if the attacker attempts to launch a DDoS attack with less number of sources, the rate of transmission from each source must be high to achieve a bandwidth attack. We monitor the count of sources which transmits above a threshold. Under a less distributive attack the mean value of this count deviates significantly, which indicates the presence of an attack.

Our contribution in this paper is a simple and fast approach to detect DDoS attacks by monitoring the deviation of (i)the count of the unique source IPs from its mean value and (b)the count of the sources transmitting at a high rate. We also introduce an effective tool to launch DDoS attacks, called (TUCANNON) of all types. We establish that our method shows significant detection performance for both benchmark DDoS dataset as well as for our own datasets than most of the previous schemes. To detect changes in our observed features we adopted the non-parametric CUSUM approach and applied it by following the idea of wang et al. [26].

The rest of the paper is organized as follows. Section 2 gives an overview of distributed denial of service attacks. Section 3 mentions some of the related work done and in Section 4 we provide our attack tool called TUCANNON. In Section 5 we present our detection algorithm along with the necessary theory. Section 6 presents the results of performing our algorithm on various network traces. Finally, conclusions are drawn in Section 7.

2 Background of DDoS Attack

The goal of a DoS attack is to overwhelm the victim by occupying its resources like network link,different queues, processing unit etc. As a consequence, the victim's performance degrades, and the legitimate user of the service observes a denial of service from the victim side. Typically, an attacker sends a few malformed packets to the victim. These packets are crafted in a way which is unexpected for the application or a protocol on the victim machine. Processing of such packets forces the victim application or protocol to freeze or to restart. However, in this paper we study the characteristics of flooding attacks. In such an attack the attacker sends a large volume of legitimate traffic to the victim site. As a result, the victims network link might go into congestion. Also the communication protocol may allocate resources to such attack traffic and might run out of resources. In case of a DDoS attack, the attacker generates a high volume of traffic from different sources in a coordinated manner. To perform a DDoS attack, the attacker uses different protocols like Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP)and Transmission Control Protocol (TCP).

The main elements of a DDOS attack are the attacker, the victim and the intermediate network of machines as shown in Figure 1.

Here the attacker is the real source of the attack. The victim could be a single machine or an entire subnetwork. The intermediate network composed of many compromised machines. Generally, the intermediate network contains masters and zombies. The masters are compromised machines, and are loaded with programs to control multiple attacking machines, called as zombies. The zombies are the compromised machines which send attack traffic to the victim in response to the command received from their masters.

The following steps take place while preparing and conducting a DDoS attack:

1) Discovery and Compromise: In this step the attacker discovers vulnerable machines over the Internet to get access to them. Later on, such systems are loaded with programs which can further detect and compromise other vulnerable machines over the net. Different scanning tools available to discover vulnerable machines such as Nmap, self-propagating tools like the Ramen worm [19] and Code Red [18] are used to discover and compromise victim machines.

2) Communication: Once machines are compromised, the attacker communicates with the masters on various occasions to know what are the agents which are on line, starting time of the attack, the rate of attack traffic, the end time of the attack etc. The agents are configured to communicate with one or more masters. Protocols like TCP, UDP and ICMP are used for these communications. Another mode of communication is by making use of the IRC servers.

3) Actual Attack: In this phase the attacker sends command to the masters which inturn activates the online zombies to carry out the actual attack.

   A detailed description of different types of DDoS attacks and along with different tools to perform such attacks can be found in [28].

3 Related Work

DDoS attack is different from a DoS attack in that a DoS attack can be characterized by high volume of traffic from a single source, however in case of a DDoS attack the volume of traffic seen by the victim is actually generated from a large number of sources over the Internet. Hence the traffic from each source is very dilute. Even if the number of sources are small, use of IP spoofing can create the same effect. Hence detecting the attack based on the source IP seems to be a better option over traffic volume based detection mechanisms [13, 24, 25]. Peng et al. [22] develop a source IP based detection mechanism, SIM (Source Addres Monitoring). It maintains a database of known source IP addresses and updates this database periodically. For each observation period they calculate the percentage of new source IP addresses in the traffic and used non-parametric CUSUM algorithm to detect abrupt change in the percentage of new IP addresses in the network. The working of this method is dependent on the content of the IP database. The attacker can send traffic in a legitimate way and thus get itself an entry in the IP database, which might turnout to be a loophole in the detection mechanism. Also they used a threshold based mechanism to detect high speed sources as attacking source, which may not be the case always. In normal traffic burst from single source can be seen quite often.

In [1] the authors use rank correlation measures to discriminate DDoS attack traffic from legitimate traffic. In [20] the authors propose a DDoS mitigation technique based on correlation pattern suitable for cloud computing environment. In [14] authors propose an entropy based DDoS mitigation technique, which is capable of discriminating flash crowd in VOIP network. In [4] authors discuss a method to prevent DDoS attack which uses IP spoofing to perform the attack.

In this paper we present an effective detection scheme called Violating Source IP Count (VSC) which is based on the observation that under normal condirion the number of unique source IPs over a fixed observation period remains stable. At each interval VSC checks whether the count significantly deviates from its mean value or not. Our detection schema detects such significant change to identify the presence of an attack. Our detection schema also keeps track of the count of such high speed sources. If the attacker uses a small size bot to escape the unique IP count deviation system,the sources has to transmit at high speed. Hence a deviation can be seen in the mean value of the count of high speed sender under an attack. VSC detects such a change and confirms an attack. To evaluate our detection mechanism we perform several experiments on different network traces and are presented in section 5. The evaluation results indicate that VSC has a short detection time and a high accuracy rate. Also since the complexity of this method is very low both in terms of space and time, this method can easily be deployed in a distributed manner in the first mile and intermediate routers to detect the attack in the beginning stage itself.

4 DDoS Attack Generation Tool TUCANNON

Based on the protocol we can classify the basic attack types as follows.

1) TCP Flood: A stream of packets with various flags (SYN, RST, ACK) are sent to the victim machine. The TCP SYN flood works by exhausting the TCP connection queue of the host and thus denying legitimate connection requests. TCP ACK floods can cause disruption at the nodes corresponding to the host addresses of the floods as well. TFN [12] is a popular DDoS tool for this type of attack.

2) ICMP Flood (e.g ping floods): A stream of ICMP packets is sent to the victim host. A variant of the ICMP floods is the Smurf attack in which a spoofed IP packet consisting of an ICMP ECHO_REQUEST is sent to a directed broadcast address. TFN [12] is a popular DDoS tool for this type of attack.

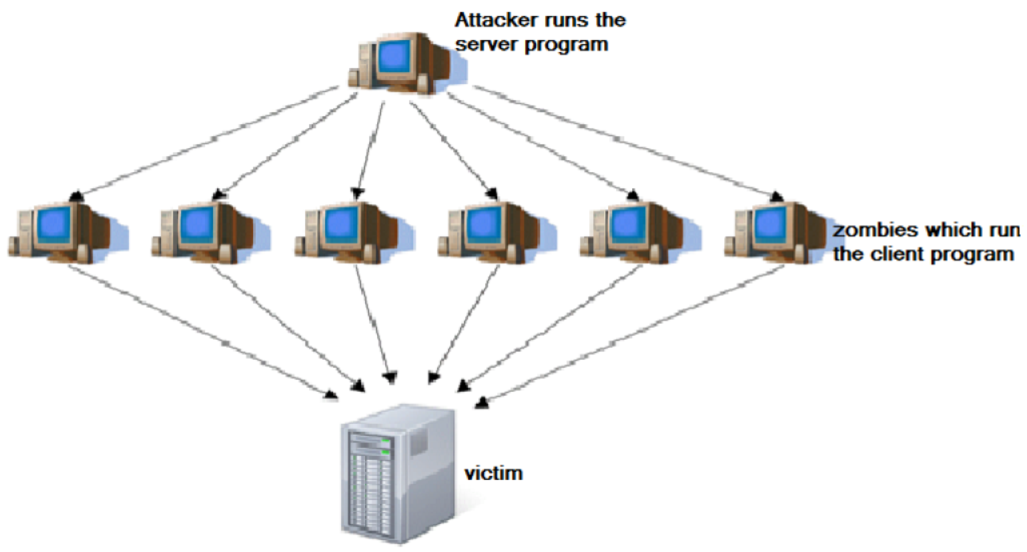

Attacker runs the server program
zombies which run the client program
victim

Figure 2: TUCANNON based direct attack strategy

3) UDP Flood: A huge amount of UDP packets are sent to the victim host. Trinoo [12] is a popular DDoS tool that uses UDP floods as one of its attack payloads.

The DDoS attack tools like trinoo, tfn, tfn2k [12] along with a lots of other tools are easily available on the internet. However these tools are not suitable for generating a coordinated DDoS attack in a testbed environment. Hence, as part of our research we developed a DDoS attack generation tool referred as TUCANNON, which can generate all the above mentioned DDoS attacks, also provide a lots of flexibility to adjust the pattern of the attack traffic like constant rate attack, increasing rate attack, pulsing attack and subgroup attack as mentioned in [16].

A pictorial description of our attack tool is shown in Figure 2. The tool comprises of two programs.

1) The first program is used by the attacker to communicate with the bots. This program uses GUI so that the attacker can easily specify various parameters like protocol type, attack pattern type etc. In this paper we will refer to this program as server program.

2) The other program is executed in each bot. This program is responsible for accepting command from server program and launch the attack accordingly. In this paper we will refer to this program as the *client program.*

4.1 Server Program

Using this program we communicate with the machines which are configured as bots in the test-bed. This program is developed with a user interface through which one can easily specify and control different properties of the attack traffic. Such properties are the protocol type (TCP, UDP and ICMP), the attack pattern (constant rate attack, increasing rate attack and pulsing attack) and the type of source IP (actual IP of the machine or randomly generate valid but spoofed IP address), no of threads (where each thread executes one copy of the slave program inside a single bot machine) and range of ports of the victim to send the traffic. When the master starts, it waits for slaves to connect to it. Figure 3 is a snapshot of the GUI of the server program.

The following is a brief description of various components of the interface:

List of Zombies: When the attacker starts the server program, it waits for the client programs to connect to it. As soon as a client program connects to the server, the clients IP address is shown in the left side panel of the interface as shown in Figure 3.

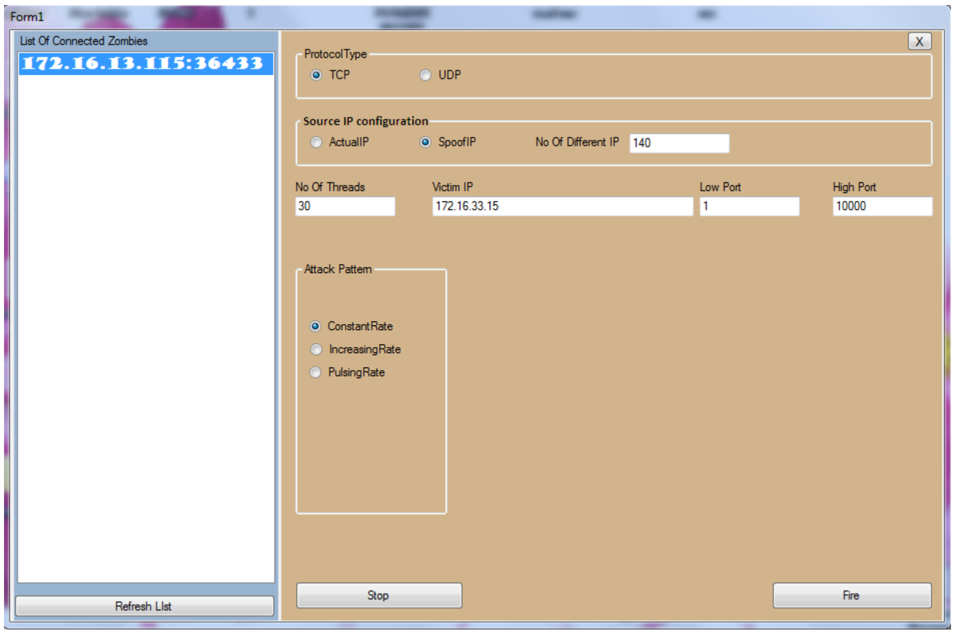
Protocol type: To launch an attack, the attacker has to select the type of protocol by selecting any one of the corresponding radio button.

Source IP Configuration: These options are used to specify whether the attack packet carries the actual source IP or a spoofed one. Also in case of spoof source IP, the attacker can specify the number of different unique spoofed IPs used in the attack. This option allows the attacker to spread the required attack traffic over a specified number of source IP.

No of Threads: The number of machines in our test bed is very limited (around 50). Hence to increase the amount of traffic each client program sends traffic by using multiple threads. The number of threads used by each client can be specified by the attacker through this input. This feature is used by the attacker to control the traffic rate in the attack.

Victim IP: This input field is used by the attacker to specify the IP address of the victim machine.

Low Port and High Port: The attacker can specify


Form1

List Of Connected Zombies

172.16.13.115:36433

ProtocolType: [✓ TCP] UDP

Source IP configuration: ActualIP [✓ SpoofIP] No Of Different IP 140

No Of Threads 30

Victim IP 172.16.33.15

Low Port 1

High Port 10000

Attack Pattern: [✓ ConstantRate] IncreasingRate PulsingRate

Figure 3: GUI of TUCANNON server program

the range of ports where to send the traffic via these input.

Attack Pattern: As mentioned earlier there can be four different traffic pattern. The Attacker can select the pattern from this list.

Fire: when the attacker clicks on this button, attack command along with the specified input is sent to all clients currently connected to the server.

Stop: The attacker can stop the attack by clicking on this button.

4.2 Client Program

This program is responsible for actually sending the attack traffic as specified by the command sent from the master. When the client program starts it connects to the server whose IP is specified as input to the client program. After connecting to the server it waits for command from the server.

5 Our Solution: Violating Source IP Count (VSC)

For a DDOS attack, the attacker's main goal is to overwhelm the server by sending illegitimate network traffic using different protocols.

One common characteristic of DDoS attack is that the volume of the traffic during an attack is very high. To generate high volume of traffic the attacker either has to use a large botnet consisting of lot of compromised machines or the attacker may send traffic from a small botnet but at very high speed. We are making the assumption that under normal condition the deviation of the number of unique source IP addresses from its mean value is bounded by an upper bound. And also during normal condition the deviation of the number of source IP addresses sending traffic above a threshold from its mean value is also bounded by an upper bound. In this paper we refer to such IP addresses as violating IP.

Based on these two assumptions we present a detection mechanism called violating Source IP count (VSC) to detect a distributed denial of service (DDoS).

The key features of VSC are highlighted below.

1) Researchers have already used source IP addresses as detection feature such as in Peng et al. [22]. However, we use the count of unique source IP addresses, in an observation period, as detection feature. This approach does not require to maintain a database of trustworthy IP addresses for its operation. Thus the memory requirement and speed of this algorithm is comparatively better, which is a key goal for a detection system. This feature makes VSC very suitable to be used in a distributed manner.

2) A DDoS attack may either use a small size bot sending traffic at a high speed or a large size bot consisting of many zombies. VSC monitors changes in both the number of source IP addresses and count of sources transmitting at high speed. Thus VSC traps the attacker from both the directions, hence reducing the scope of the attacker.

5.1 Overview of Violating Source IP Count

VSC attempts to detect the presence of attack by monitoring two features of the traffic, namely the number of unique source IP addresses and the number of violating source IP addresses. VSC collects the incoming packets during every observation period, say δT and inserts the packets into a binary search tree based on their source IP address. Each node in the tree has a count field that specifies the number of packets from the source IP represented by the source IP field of the node, as illustrated in Figure 4

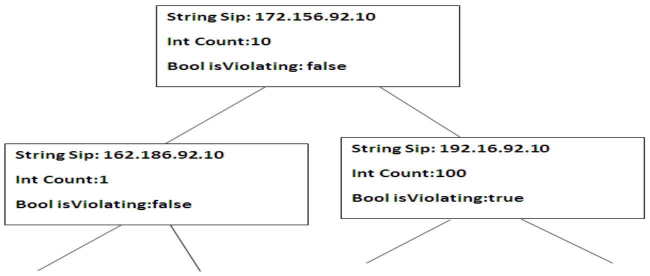


Figure 4: The binary search tree used in the detection engine

If the number of packets for a source IP address is greater than a certain threshold, that IP address is marked as violating IP. This information is used by VSC to detect DDoS attack that uses a small size botnet to carry out the attack. Also, at the end of each observation period the number of nodes in the binary search tree gives the number of unique source IP addresses during the observation period. Thus at the end of each observation period the BST (Binary Search Tree) gives us a) the number of violating source IP addresses V_i, and b) the number of unique source IP addresses X_i in the current observation period t_i.

5.1.1 System Architecture

Figure 5 provides an overview of VSC mechanism. The VSC mechanism consists of three basic components, viz, detection engine, decision engine, and response engine.

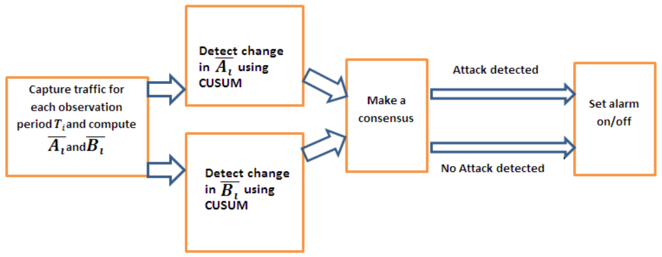

Capture traffic for each observation period T_i and compute $\overline{A}_t$ and $\overline{B}_t$
Detect change in $\overline{A}_t$ using CUSUM
Detect change in $\overline{B}_t$ using CUSUM
Make a consensus
Attack detected
No Attack detected
Set alarm on/off

Figure 5: Architecture of VSC

The detection engine processes the incoming traffic to detect any attack. The task of the decision engine is to combine the results from the detection engine and to reach a consensus about the occurrence of an attack. The response engine in turn sets an alarm on or off based on the output of the decision engine.

5.1.2 Placement of the Detection Mechanism

The VSC can be deployed in different locations in a network as shown in Figure 6.

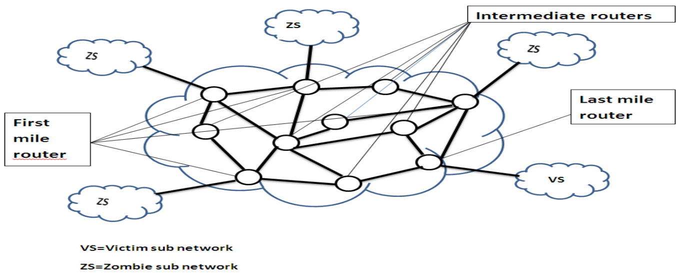


Figure 6: Different possible placement location of VSC

If it is deployed in the first mile router, the detection rate will depend on the size of the bot. If a small bot is used then the number of packets from one IP must be high, which our detection engine can detect by estimating the change in the count of violating source IP count. However, in case of a large bot, the detection rate might decline.

If deployed in intermediate routers, the detection rate increases as it moves towards the victim site. One can thus deploy VSC at different intermediate routers and detect the occurrence of an attack in a distributed manner. Chen et al. [9] describe such an approach in their work.

Another point of deployment is at the victim site, i.e, the last mile outer. Since all the attack traffic aggregate in the last mile router, deviation of $\overline{A}_n$ and/or $\overline{B}_n$ can easily be detected, if there is any.

Among these different detection points the intermediate routers and the last mile router carries the greatest interest from the point of view of the victim. In this paper we present the experimental results performed on the last mile router. Detection of the attack by placing VSC at different intermediate routers in a distributed manner is out of thee scope of this paper.

5.2 Theory Behind VSC

As mentioned above, VSC monitors the number of unique source IP address X_i and number of violating source IP addresses V_i in each observation period to detect the occurrence of an attack in the network. Under the normal condition, the deviation of X_i and V_i from its mean value is less, however under an attack these parameters deviate from their mean largely. Our detection engine thus monitors and detects (if any) such a significant change in these two parameters and confirms as an attack based on some threshold value.

The following section describes the approach we use to detect such change in the above mentioned framework.

5.2.1 The Non-Parametric CUSUM Algorithm:

Let $X_n, where\, n = 0, 1, 2, 3..$ and $V_n, where\, n = 0, 1, 2, 3..$ be the number of unique source IP addresses and the number of violating source IP address in an observation period t_n. Since X_n and V_n are highly dependent on different attributes of the network (such as size, time of the day, etc) from which they are collected, we first normalize X_n and V_n by the average value of X_n and V_n respectively. Let $\overline{X}_n$ and $\overline{V}_n$ represent the mean value of X_n and V_n. Then $\overline{X}_n$ and $\overline{V}_n$ can be computed as follows $\overline{X}_n$=α*$\overline{X}_n$ − 1+(1-α)*X_n, $\overline{V}_n$=α* $\overline{V}_n$ −1+(1-α)*V_n Where α is the memory factor and lies between 0 and 1.

Thus from X_n and V_n we define A_n=$X_n/\overline{X}_n$ B_n=$V_n/\overline{V}_n$ Since A_n and B_n are normalized values, no longer they are dependent on the current network characteristics.

We use the concept of sequential change point detection [10] in our detection algorithm. The goal of the change point detection mechanism is to detect the presence of a change of the mean value in a observed time series data. In our algorithm, it detects change in A_n and B_n. However,accurate estimation of A_n and B_n are challenging task, hence we use non-parametric CUSUM method [24] in our detection algorithm. Non-parametric CUSUM is not model specific and hence suitable for our purpose. The basic idea of using non-parametric CUSUM to detect abrupt change in a time series data is based on the model presented in Wang et al. [10]. The details of non-parametric CUSUM can be found in [24]. Here we demonstrate how to apply non-parametric CUSUM on A_n to detect change. Similar approach is taken in case of B_n.

Under normal condition, the mean of A_n denoted by c ($i.e$, $c = E(A_n)$) is near by 1. We chose a parameter which is the upper bound of c. From A_n we derive another random sequence $\overline{A}_n$ such that $\overline{A}_n = A_n - c$. This transformation will make the mean value of $\overline{A}_n$ negative under normal condition, which is a basic assumption of non-parametric CUSUM algorithm [6]. Now consider h as the lower bound of the amplitude of increase in the mean value of $\overline{A}_n$ during an attack and $h \gg c$. As presented in wang et al. [26] the nonparametric CUSUM algorithm can be written as

$$Y_n = S_n - \min \ S_k, 1 \le k \le n.$$

Where $S_k = \sum_{i=1}^{k} \overline{A}_i$, $S_0 = 0$ at the beginning and Y_n is the accumulated positive values of $\overline{A}_i$. Thus if Y_n is very large it is a clear indication of the deviation of the observed value of the random sequence from its mean value.We use a threshold value N which is compared against Y_n at the end of each observation period. If Y_n exceeds N an attack is detected. Thus we can now formally define the detection function as

$$D_N(Y_n) = \begin{cases} 0 & \text{if } Y_n \le N \\ 1 & \text{otherwise.} \end{cases}$$

Where 1́ndicates an attack and 0́detects normal traffic.

5.2.2 Parameter Specification

Two key measures of greatest interest for a DDoS attack detection system are given below.

1) False alarm rate, i.e, the number of normal instances reported as attack over a specific period of time.

2) Detection time, i.e, time duration between the starting of an attack and the detection of the attack.

However, both these design goals are mutually conflicting, as expected! To achieve one other one often has to compromise up to extent. in practice (1,4,30 p) CUSUM is considered as optimal in terms of both false alarm rate and detection time. As presented in Brodsky et al. [6]

$$\begin{aligned} \tau_N &= \inf n : D_n(.) = 1 \\ \rho_N &= \frac{(\tau_N - m)^+}{N}, \end{aligned} \qquad (1)$$

where τ_N = detection time; ρ_N = normalized detection time after a change occurs. Inf represents $infimum$. n is the time when the attack started.

ρ_N and h can be related by the following equation

$$\rho_N \to \gamma = \frac{1}{h - |c - a|} \qquad (2)$$

where $h - |c - a|$ gives the mean of $\overline{A_n}$, after an attack begins. As mentioned in [26] the above equation gives an upper bound of the actual detection time. Thus to achieve our design goals we have to choose optimal values for the parameters a and N. It is clear from Equation (1) and Equation (2) that once we are given a,h and *a detection time period* we can calculate N accordingly.

The parameter a is used to offset A_n to be $\overline{A_n}$, so that $\overline{A_n}$ has a negative mean under normal condition. If a is chosen to be very high the likelihood of getting positive values in the sequence $\overline{A_n}$ is less. In turn the accumulated value i.e, Y_n might not reach the threshold.

The parameter N specifies the threshold for Y_n. If N is chosen to be very high, false alarm rate will be low at a cost of high detection time. On the other hand, a small value of N may increase the false alarm rate.

As mentioned earlier CUSUM algorithm needs a, h and *detection time interval* to be specified and calculates N by using Equations (1) and (2).

Here a is the upper bound estimation of the mean of $\overline{A_n}$. From the definition of $\overline{A_n}$ can safely assume a as 1.1.

The parameter h specifies the amplitude of the minimum increase of the mean value of $\overline{A_n}$ under an attack. By following the same principle as in wang et al. [26] we set $h = 2 * a$.

We used *detection interval* as 3 sec. Assuming $c = 1$, from Equation (1) and (2) we get $N = 6.3$.

6 Performance Evaluation

To evaluate the detection efficiency of our mechanism we validate VSC by using both available benchmark datasets

Table 1: Network traces used in the experiments

Trace	Duration	Created on	Type	Label
TU_170813	1 hr	Aug,2013	Bi-Directional	normal
DARPA	3 week	1999	Bi-Directional	attack
CAIDA2007	1 hr	2007	Uni-Directional	attack

as well as our own testbed network traces. Table 1 describes the traces used in our experiments.

Also to show the effectiveness of VSC under different attack scenario, we embedded simulated attack traffic of various characteristics generated in our testbed into the normal traffic of Table 1. Following sections provide the description of our testbed followed by different experiments and their results.

6.1 Testbed Setup

Our testbed consists of two sub-networks as shown in Figure 7.

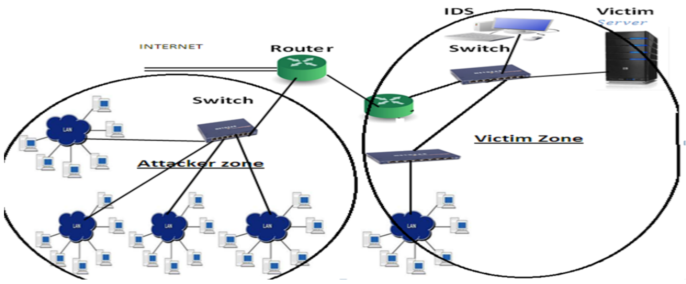


Figure 7: The testbed of our experiments

1) The first sub-network (marked by the left side oval) is used to generate the attack traffic. In this sub-network we installed TU-CANNON as explained earlier.

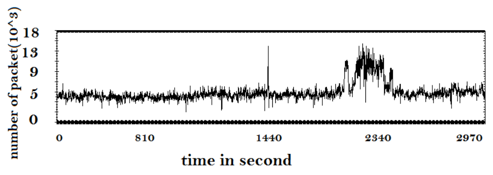
2) The second sub-network (marked by the right side oval) contains the victim server and also the capturing and detection unit (labeled as IDS in the diagram).

6.2 VSC Under Normal Condition

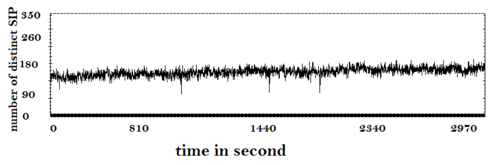
To perform our experiments we used the TU_170813 dataset as normal traffic reference. We are assuming this traffic as attack free traffic. the result of VSC when applied on TU_170813 is shown in Figure 8. We can see that both of our test statistics are much below their threshold value.

6.3 Detection of Low Rate DDoS Attack

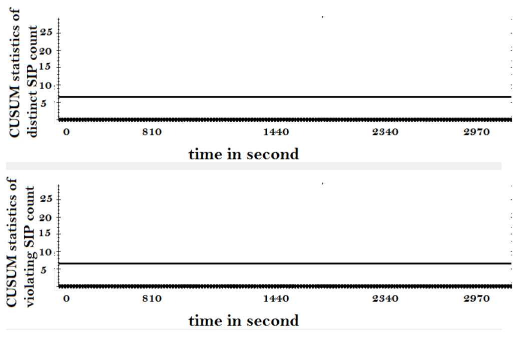
Our detection mechanism detects attack based on the deviation of the number of unique source IP address from its mean value, rather than the volume of the traffic.thus our detection mechanism can detect low rate DDoS attack involving large number of sources. To demonstrate this we embedded a simulated attack of duration 1 minutes into the normal traffic. The attack was performed at a rate of 250 packets/sec, which is much lesser than the usual traffic of the network. Hence such an attack can easily escape a traffic volume based detection mechanism. However as shown in Figure 9, our detection mechanism detects the attack within 23 sec of the starting of the attack. However, during the attack period the traffic volume does not change significantly.



(a) packet rate of in normal scenario

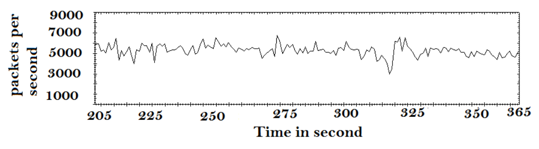


(b) unique IP rate in normal scenario

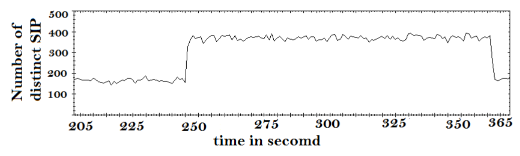


(c) CUSUM statistics under normal condition

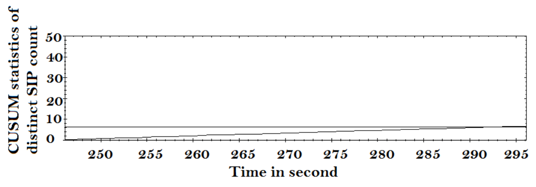
Figure 8: Result of VSC on attack scenario 1



(a) packet rate of in attack scenario

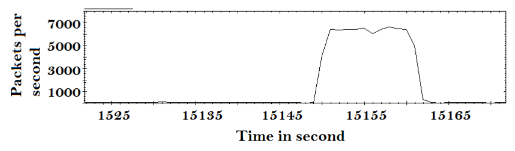


(b) unique IP rate

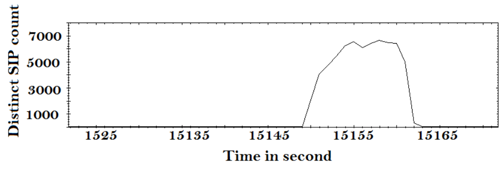


(c) CUSUM detection of attack

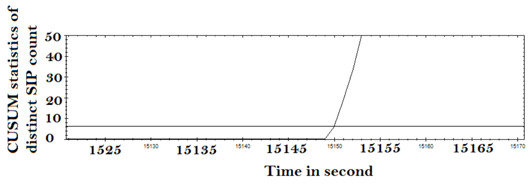
Figure 9: Result of VSC on attack scenario 1



(a) Packet rate of DARPA



(b) Unique IP rate of DARPA



(c) CUSUM detection in DARPA

Figure 10: Result of VSC on DARPA dataset

6.4 Detection of Randomly Spoofed Source IP Attack

We used the DARPA dataset [17] in our experiments to show the effectiveness of VSC in detecting DDoS attack which uses a randomly spoofed source IP addresses.Figure 11 demonstrate the result of this experiment. From the figure we can see that the attack present in the DARPA dataset is easily detectable by VSC in less than 3 seconds.

6.5 Detection of DDoS Attack From A Small Size Bot

The attacker may use a small size bot net (consisting a small number of sources) to carryout the attack. However in that case the speed of the individual source need to be high to end up in an effective DDoS attack. We used a simulated attack consisting of 7 sources, performing an attack at 2000 packets/sec, for a duration of 10 minutes. we used 150 packets/sec as the threshold to mark an IP as violating IP. From Figure 11 it is clear that VSC can detect such an attack in less than 15 seconds, by detecting a change in the violating source IP count.

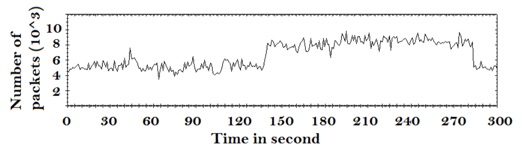
6.6 Detection of DDoS Attack in CAIDA2007 Dataset

CAIDA 2007 [7] is an widely used and benchmark DDoS network trace. We applied VSC on this dataset in our experiment. The result of VSC is shown in Figure 9(a). The length of CAIDA is around 1 hour. The first 30 minutes contains traffic at a low rate, from a small number of sources. However at the begining of the second 30 minitue half, there is an abrupt change in both number of unique source IP address as well as the traffic volume. VSC detects this change with in 8 seconds.

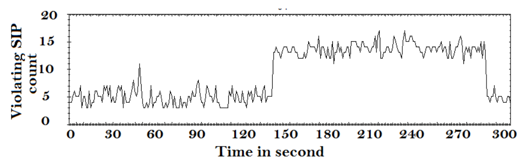
7 Conclusion and Future Work

In this paper we present a robust and low cost method to detect DDoS attack. Our method detects DDoS attack by monitoring the deviation of the count of unique source IP and the count of source IPs whose transmission rate is higher than a threshold value. Another feature of our detection mechanism is that to detect attack performed from small size bot net, i.e, where the transmission rate of each source is very high we keep track of the count of such sources. In case of an attack, this count value deviates from its mean value abruptly, and thus we can detect this change to confirm the attack.

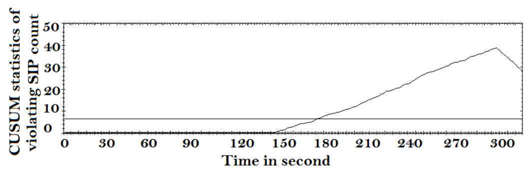
In this paper we present the experimental results of



(a) Packet rate of small size BOT

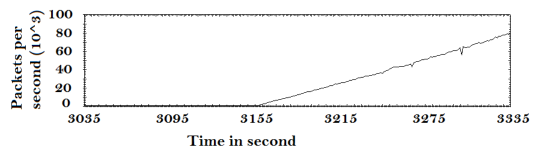


(b) High speed IP rate of small size BOT

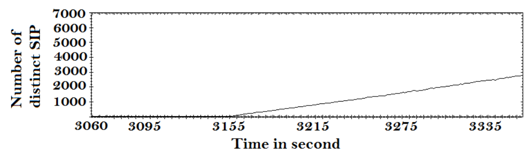


(c) CUSUM detection on small size BOT

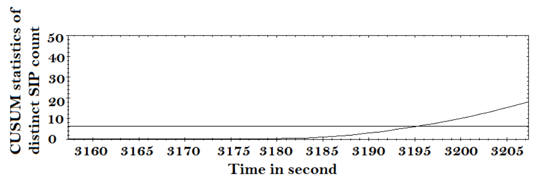
Figure 11: Result of VSC on small size BOT



(a) packet rate of CAIDA



(b) unique IP rate of CAIDA



(c) CUSUM detection on CAIDA

Figure 12: Result of VSC on CAIDA

applying VSC in a last mile router in the context of simulated network traces as well as benchmark datasets like CAIDA and DARPA . Our future research will be on the deployment of the detection mechanism in a distributed manner. A distributive approach will be able to detect the attack as it propagates towards the victim. Thus the victim will get enough time to take necessary precaution.Also a distributive approach will be able to detect the attack near to the source, which is a very desired property of an detection system.

Acknowledgments

We are thankful to MHRD and UGC of Govt of India for funding our work under the schemes (i) Centre of Excellence under FAST and (ii)Special assistance under DRS II.

References

[1] A. Ain, M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Rank Correlation for Low-Rate DDoS Attack Detection: An Empirical Evaluation," *International Journal of Network Security*, vol. 18, no. 3, pp. 474–480, 2016.

[2] A. Anurag, "Network neutrality: Developing business model and evidence based net neutrality regulation," *International Journal of Electronics and Information Engineering*, vol. 3, no. 1, pp. 1–9, 2015.

[3] M. Basseville, and I. V. Nikiforov, *Detection of Abrupt Changes: Theory and Application (vol. 104)*, Englewood Cliffs: Prentice Hall, 1993.

[4] Y. Chen, S. Das, P. Dhar, A. El-Saddik, and A. Nayak, "Detecting and Preventing IP-spoofed Distributed DoS Attacks," *International Journal of Network Security*, vol. 7, no. 1, pp. 69–80, 2008.

[5] R. B. Blazek, H. Kim, B. Rozovskii, and A. 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*, pp. 220–226, 2001.

[6] E. Brodsky, and B. S. Darkhovsky, *Nonparametric Methods in Change Point Problems (vol. 243)*, Springer Science & Business Media, 2013.

[7] CAIDA UCSD, *DDoS Attack 2007 Dataset*, 2007. (`http://www.caida.org/data/passive/ddos-20070804_dataset.xml`)

[8] G. Carl, G. Kesidis, R. R. Brooks, and S. Rai, "Denial-of-service attack-detection techniques," *IEEE Internet Computing*, vol. 10, no. 1, pp. 82–89, 2006.

[9] Y. Chen, K. Hwang, and W. S. Ku, "Collaborative detection of DDoS attacks over multiple network domains," *IEEE Transactions on Parallel and*

Distributed Systems, vol. 18, no. 12, pp. 1649–1662, 2007.

[10] CNN, *Immense Network Assault Takes Down Yahoo*, Feb. 2000. (`http://www.cnn.com/2000/TECH/computing/02/08/-yahoo.assault.idg/index.html`)

[11] CNN, *Cyber-attacks Batter Web Heavyweights*, Feb. 2000. (`http://www.cnn.com/2000/TECH/computing/02/09/-cyber.attacks.01/index.html`)

[12] P. J. Criscuolo, *Distributed Denial of Service: Trin00, Tribe Flood Network*, Tribe Flood Network 2000, and Stacheldraht CIAC-2319, California Univ Livermore Radiation Lab., Feb. 2000.

[13] D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to IP traceback", *ACM Transactions on Information and System Security*, vol. 5, no. 2, pp. 119–137, 2002

[14] N. Jeyanthi, and N. C. S. N. Iyengar, "An Entropy based approach to detect and distinguish DDoS attacks from flash crowds in VoIP networks," *International Journal of Network Security*, vol. 14, no. 5, pp. 257–269, 2012.

[15] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 3, pp. 62–73, 2002.

[16] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms", *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[17] MIT Lincoln Laboratory, *2000 Darpa Intrusion Detection Scenario Specific Data Sets*, 2000.

[18] D. Moore, and C. Shannon, "Code-Red: a case study on the spread and victims of an Internet worm," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment*, pp. 273–284, 2002.

[19] J. Nazario, *Defense and Detection Strategies Against Internet Worms*, Artech House, 2004.

[20] P. Negi, A. Mishra, and B. B. Gupta, "Enhanced CBF packet filtering method to detect DDoS attack in cloud computing environment," arXiv preprint arXiv:1304.7073, 2013.

[21] E. U. Opara, O. A. Soluade, "Straddling the next cyber frontier: The empirical analysis on network security, exploits, and vulnerabilities," *International Journal of Electronics and Information Engineering*, vol. 3, no. 1, pp. 10–18, 2015.

[22] T. Peng, C. Leckie, and K. Ramamohanarao, "Proactively detecting distributed denial of service attacks using source IP address monitoring," in *International Conference on Research in Networking*, pp. 771–782, Springer Berlin Heidelberg, 2004.

[23] M. Sachdeva, G. Singh, K. Kumar, and K. Singh, "DDoS incidents and their impact: A review," *International Arab Journal of Information Technology*, vol. 7, no. 1, pp. 14–19, 2010.

[24] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback", *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, pp. 226–237, 2001.

[25] D. X. Song, and P. Adrian, "Advanced and authenticated marking schemes for IP traceback", in *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'01)*, pp. 878–886, 2001.

[26] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN flooding attacks," in *Proceedings of Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'02)*, vol. 3, pp. 1530–1539, 2002.

[27] D. K. Yau, J. Lui, F. Liang, and Y. Yam, "Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles," *IEEE/ACM Transactions on Networking*, vol. 13, no. 1, pp. 29–42, 2005.

[28] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

Biography

Ram Charan Baishya is a Ph.D. student in the Department of Computer Science and Engineering at Tezpur University.

Nazrul Hoque is a Ph.D. student in the Department of Computer Science and Engineering at Tezpur University.

Dhruba Kr Bhattacharyya received his Ph.D. in Computer Science from Tezpur University in 1999. He is a Professor in the Computer Science & Engineering Department at Tezpur University. His research areas include Data Mining, Network Security and Content based Image Retrieval. Prof. Bhattacharyya has published 220+ research papers in the leading international journals and conference proceedings. In addition, Dr Bhattacharyya has written/ edited 8 books. He is a Programme Committee/ Advisory Body member of several international conferences/workshops.