Source Identification of Spoofed DDoS Attacks using an Image Processing Approach

Tae Hwan Kim a,*, Dong Seong Kim b, Sang Min Lee c, Joon Heo a

aFuture Internet Research Team, National Institute for Mathematical Sciences, Daejeon, Korea
bDepartment of Computer Science and Software Eng., University of Canterbury, New Zealand
cDepartment of Computer Eng., Korea Aerospace University, Korea

Abstract

This paper presents a novel source identification scheme for spoofed DDoS attacks using an image processing method. The key idea is that “DDoS attack traffic” that uses subnet spoofing is represented as “lines” on the spatial image planes, and they can be recognized by an edge detection algorithm. Applying the clustering technique to the lines makes it possible to identify multiple attack source networks simultaneously. On the identified networks which the zombie hosts reside, we then employ a signature based pattern extraction algorithm, called pivoted movement, and the DDoS attacks are filtered by correlating IP and MAC pairings signature. Unlike previous IP traceback schemes such as packet marking, which tried to diagnose the entire attack path, our proposed scheme focuses on identifying only the attack source. Our approach can achieve an adaptive response to DDoS attacks, thereby mitigating them at the source, while minimizing the disruption of legitimate traffic. The proposed scheme is analyzed and evaluated on the IPv4 and IPv6 network topology from CAIDA and the results show superior effectiveness.

Keywords: DDoS attacks, edge detection, intrusion detection, IP traceback, security

1 Introduction

In launching DDoS attacks, zombie hosts are recruited and their IP (Internet protocol) addresses are spoofed (i.e., IP spoofing) with the attacker controlling this process. This can not only conceal the zombie hosts identity but also weaken the attempts of defenders to respond to these attacks. The “spoofed project” in [10] shows that a large part of the Internet is vulnerable to IP spoofing; it deploys ingress/egress filtering which is considered its typical countermeasure. Both the ingress and egress filtering may throttle the strength of DDoS attacks and ensure that traffic leaving a local network may carry only IP addresses that belong to the subnet [9]. However, the attackers are more sophisticated. If a zombie host spoofs IP addresses from the address space assigned to its subnet, called “subnet spoofing,” the spoofed packets cannot be detected, in spite of ingress/egress filtering, and the IP addresses of the zombie hosts are still effectively hidden. It is rather hard to detect spoofing at any point between the exit router and the victim systems, and to the best of our knowledge, there are only a few feasible solutions [13] for subnet spoofing.

In this paper, we propose a novel source identification scheme to combat subnet spoofed DDoS attacks. The motivation of the proposed idea is that incoming spoofed DDoS traffic can appear as “lines” on an image space. Recognition of these lines by an edge detection algorithm (we use the Canny edge detection algorithm) enables one to identify DDoS attack sources. We can then filter out DDoS attacks using a signature based pattern extracting algorithm named “pivoted movement,” which correlates information, such as IP and MAC (Media Access Control) pairings. The proposed method of identifying DDoS attack sources comprises the following steps:

i) Incoming packets to victim systems are mapped into a two dimensional image space called the NetSCENE (network scene); DDoS attacks are presented as lines on the NetSCENE.

ii) DDoS attack source networks are identified by an edge detection process (we use the Canny edge detection algorithm), and each edge is linked to the local network addresses at which the zombies are located.

iii) The victim system sends instructions to the source router connected to the identified local network to perform the pivoted movement algorithm; the outgoing packets from the local network are then marked with a signature by the source router.

iv) The router can filter out attack packets based on the attack signature. The proposed scheme handles multiple sources simultaneously and repeats this process until the attack is mitigated or an administrator stops it.

The main contributions of our paper are three-fold: First, we propose a network visualization method using NetSCENE, which enables DDoS attacks to be
visualized clearly. It facilitates intuitive judgment, and the analysis of a large amount of traffic data. Second, the proposed identification scheme can not only detect DDoS attacks but also simultaneously trace them back to the multiple zombie hosts. Third, it supports the DDoS response paradigm in which the attack detection module is placed close to the victim system, while the countermeasure (such as packet filtering) can be implemented at the source. The rest of the paper is organized as follows. Related work is introduced in Section 2. The proposed source identification scheme is presented in Section 3. The experimental results are presented in Section 4, and we present our conclusions in Section 5.

2 Related Work

In previous works on DDoS attack source identification and IP traceback [12], packet marking techniques were proposed. Packet marking is used to reveal internal network information to end-hosts. This information is sent by the routers in the headers of regular IP packets on their way to their destination. However, packet marking techniques have limitations from the point of view of practicality. The packet marking procedure causes an additional overhead to routers and it is not easy to deploy routers in which a marking module is installed over the networks of interest. The major drawback of these schemes is their combative weakness in cases when packet markings have been forged [1]. Visualization can be applied for various fields such as analysis of networks behavior [4, 8], intrusion detection [18] and identifying anomaly types [15]. Kim and Kang defined the signs of various network anomalies that appear on visualization tools, and proposed an algorithm called RADAR, which can extract anomalies using a simple principle [8]. It can handle various anomalies using one simple extracting algorithm and just three bits of packet signatures; however, legitimate packet signatures cannot be defined and therefore it causes a high level of false positive alarms. The NetViewer [15] is based on the idea that “by observing the traffic and correlating it to the previous normal states of traffic, it may be possible to see whether the current traffic is behaving in an anomalous manner.” The authors' previous work [14] proposed a technique for detecting a traffic anomaly by analyzing the correlation of destination IP addresses in outgoing traffic at an egress router. Their hypothesis was that a destination IP address will have a high degree of correlation, for a number of reasons, and that changes in the correlation of outoming addresses can be used to identify network traffic anomalies. Based on this hypothesis, they applied a discrete wavelet transform on the address and port number correlation data over several time scales.

3 Proposed DDoS Attack Source Identification Scheme

3.1 A system model and assumptions

We assume the followings.

i) Every host, either a client or a server, is connected to its local edge router. The edge routers are in turn interconnected by core routers. The proposed scheme is transparent to link-layer devices such as switches. Therefore, any switches that may be connected between the client and its egress router are ignored.

ii) Every route from the attack source(s) to the victim is fixed during the time frame of interest. A recent study [5] has shown that 95% of the routes observed in the Internet have fewer than five observable daily changes.

iii) According to the spoofer project [10], zombies spoof source IP addresses within their subnet range on their packets [9] to avoid ingress/egress filtering deployed by ISPs (Internet Service Provider) in a large area.

iv) The MAC address of the zombie hosts and TTL field of the packet they send are not spoofed [12].

![Fig. 1. Overall flow of proposed scheme](image)

The overall flow of the proposed scheme is depicted in Figure 1. First, incoming traffic is represented in a 2D (two-dimensional) image space named NetSCENE. Second, the source network of DDoS attacks is represented as edge(s) on NetSCENE. The network address of DDoS attacks sources is identified using an edge detection algorithm (we use the Canny edge detection algorithm). Third, we use an improved version of “pivoted movement” to identify the zombie hosts’ addresses. Fourth, filtering is performed. In the following subsections, we present an illustrative example and each step will then be described in more detail.

3.2 An illustrative example

In figure 2, we present an overview of the proposed scheme using an illustrative example. There are no zombies in local network 1, whereas zombies are located in local networks 2, 3, and 4; they generate DDoS attack packets with subnet spoofing assigned by their local network prefix. For example, zombie ‘a’ randomly chooses source IP addresses from 210.119.29.1 to 210.119.29.254 and uses them for...
each of the attack packets. In the case of the zombie ‘b’, its spoofable IP range is from 204.17.5.97 to 204.17.5.126. At the victim system, the patterns of these attacks appear as edges on the NetSCENE. These edges on the NetSCENE are then extracted by an edge detection process and linked to the network addresses as 210.119.29.0/24 and 204.17.5.96/27, according to the position and length of the edge. Through this process, the local networks linked to the router ‘A’ and ‘B’ are identified, and finally we can locate the network zone in which the zombies are residing. It is not sufficient to use only the source “network” in which the zombie hosts reside to filter the DDoS packets heading toward the victim system, because the packets from legitimate hosts located in local networks cannot bypass the filtering module, and thus, a high level of false positives will result. To cope with this issue, we perform one more step using the signature based pivoted movement algorithm. The idea of the algorithm stems from the fact that zombie hosts spoof their IP address but not their MAC address [3]. The legitimate packets pair one IP address to one MAC address, while the attack packets pivot the source IP address on the MAC address and destination IP address. Consequently, only the attack packets can be filtered out based on the signature. Each phase is described in more detail in the next subsection.

3.3 Construction of NetSCENE: network as an image

Certain attacks are interpretable only when the aggregate traffic flow is considered [17] since, when only individual flow behavior is considered, DDoS attack flows are not distinguishable from normal flows. In order to address these issues and improve scalability, our approach considers the analysis of aggregate packet header data. To analyze and visualize the aggregated network traffic, we use NetSCENE as our network visualization tool, which monitors the packet headers of network traffic at regular intervals and analyzes the aggregate data. When DDoS attacks occur, the network usage pattern may change and the peculiarities (e.g., edges) can become visible on the image. Furthermore, the NetSCENE is specially designed to achieve the goal that each edge on the image is linked to the specific local network addresses that is the attack sources. NetSCENE employs IP header data collected downstream of the network at the victim system. The collected data in each domain may include the source IP address and hop count value \((H_c)\). Although the hop count value is not directly stored in the IP header, it can be easily computed according to [5] based on the TTL field. In particular, to describe the specific dispersion of network traffic by IP spoofing, the IP address domain is divided into two portions: the network address \((I_n)\) and the host address \((I_h)\). In the /24 networks of IPv4, for example, the upper 24 bits of the IP address represent the network address and the lower 8 bits of the IP address represent the host address. Each packet is mapped into the NetSCENE, two-dimensional image space, according to their values using the two features, \(I_n\) and \(I_h\). The hop count \((H_c)\) and measured traffic volume \((T_c)\) of each flow are stored in the pixel of the NetSCENE, and \(T_c\) is presented as the pixel density. Consequently, Each pixel in NetSCENE represents a “flow”, which is defined as a set of packets sharing the same \(I_n\), \(I_h\), and TTL value.

Fig. 2. An illustrative example

Fig. 3. Example of NetSCENE, which is an image generated using processed packet header information

The size of the NetSCENE is sufficient if all passing traffic can be represented without any overlapping between the IP addresses of flows. To cope with this,
we employ hash table-based space folding techniques which eliminate the blank in the NetSCENE. It looks up the hash table when the $I_n$ of the packet comes to the input interface of the hash table. If the $I_n$ is “not found”, which means that the packet is unobserved within the current time window, it is registered in the corresponding hash table as a new sighting and returns the hash result of $I_n$, that is vertical-axis value of the NetSCENE. If $I_n$ is “found”, then the corresponding vertical-axis value is returned according to the hash table. In this way, the size of the NetSCENE can be reduced considerably and the empty blank in the space is eliminated. Consequently, each packet can be maintained collectively, and each edge can be linked to the local network address by its vertical-axis value.

3.4 Edge detection

The NetSCENE may help network administrators to identify the signs of ongoing attacks intuitively. However, simply mapping each and every arriving packet onto the NetSCENE is not enough. This approach has the more important limitation that low intensity attacks and attacks in the proximity of the dominant legitimate traffic may not be sufficiently visible in the NetSCENE. In order to capture only significant (i.e., attack) information, it is crucial to extract the attacks from the legitimate background traffic. We therefore adopt image processing approach for extracting the attacks from the packet flow shown in the NetSCENE. In order to extract edges which represent DDoS attacks, we adopt the Canny edge detection algorithm. There are several edge detection algorithms, such as the Hough transform [11] or the Laplacian method [6]. However, the Canny edge detection algorithm, based on the convolution operation of pixels, is the most suitable on the NetSCENE. The Canny edge detection system uses multi-steps consisting of smoothing, stressing, and linking processes.

The first step involves smoothing the NetSCENE by convolving with a Gaussian smoothing operator. Since it removes small-scale textures and noises as effectively as possible for a given spatial extent on the NetSCENE, this process can be considered to remove legitimate traffic and noise while preserving the significant traffic patterns. The convolution operation is described in the following equation (1). The $K_{gus}$, kernel of a Gaussian operator with a standard deviation of $\sigma = 1.4$, is shown in equation (2). After this processing, the edges can be extracted more clearly.

$$S'(x, y) = g(k, l) \otimes S(x, y)$$

$$= \sum_{k=N}^{N} \sum_{l=N}^{N} g(k, l)S(x-k, y-l) \quad (1)$$

where, $g(k, l) =$ convolution kernel, $S(x, y) =$ unprocessed NetSCENE, $S'(x, y) =$ processed NetSCENE

$$K_{gus} = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \quad (2)$$

$$K_{sobel} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (3)$$

![Unprocessed Image](image1.png) ➔ ![Smoothing](image2.png) ➔ ![Stressing the Edges](image3.png) ➔ ![Linking](image4.png)

Fig. 4. Canny edge detection process
The second step is to approximate the gradient in the $x$-direction respectively by applying the Sobel operator shown in equation (3). Only the $x$-direction Sobel operator is used, since we are interested only in the horizontal edges in the NetSCENE. The gradient magnitudes (also known as the edge strengths) can be determined by a convolution operation using the Sobel operator, and a NetSCENE which has strengthened edges results. The strong legitimate traffic that could not be eliminated by the Gaussian smoothing process. On the other hand, some pixels of an edge may be set to zero, since some IP addresses may not be chosen by the zombies. The simplest way to distinguish between these pixels is to implement the linking process. In the NetSCENE, the edge length and location are restricted, because the lower 8 bits of the IP address that can be a network address are several. Also the host IP address range is fixed. This rule is from subnetworking and is depicted in Figure 3. Therefore, the pixels which remain until Step 3 can be judged as to whether the edge should be included or eliminated. After processing all these steps, we obtain a processed NetSCENE that includes only the edges.

3.5 Hop count based source network identification

The edges extracted by the Canny edge detection algorithm can identify the attack sources quite accurately. However, there is a possibility that false identification will occur, since the hop count values are not considered in the edge linking step of the edge detection. Note the assumption, defined above, that every route from the attack sources to the victim system is fixed during the timeframe of interest. Hence packets from the same sources have the same $H_c$. For example of false identification case, some pixels which have different $H_c$ each other can be represented as a single edge in the NetSCENE. We call these edges as weak edges. One way to remove the weak edges could be to check the $H_c$ of pixels. In order to analyze the edge attribute quantitatively, we calculate the edge correlation and the completion degree. The edge correlation is calculated by equation (4-1), and the completion degree of edge by equation (4-2).

$$C_o = \frac{n\left(\left\{x_i = x | x_i \in X, x \text{ is a mode}\right\}\right)}{N_o} \quad (4-1)$$

$$P_o = \frac{n\left(\left\{x_i \in X | x_i = 0\right\}\right)}{N_o} \quad (4-2)$$

where $X$ is the set of the pixels included in the edge $E_{ijn}$ and $N_{ijn}$ is the number of pixels included in the edge $E_{ijn}$ ($i$ and $j$ is the $x$ and $y$-axis value of pixel which is in the $n$-th edge). $x$ is the mode value that occurs most frequently in the dataset as the representative value. We set 0.80 as the correlation threshold. This means that more than 80% of the corresponding packets, which are organizing the edge, should have the same hop count value in order to be a strong edge.

3.6 Signature based pivoted movement

<table>
<thead>
<tr>
<th>Type</th>
<th>$S_1$</th>
<th>$S_n$</th>
<th>$D_1$</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDoS attack packet</td>
<td>spoofed IP</td>
<td>real</td>
<td>victim</td>
<td>$&lt;0,1,1&gt;$, $&lt;0,0,0&gt;$, $&lt;1,1,1&gt;$: false negative</td>
</tr>
<tr>
<td></td>
<td>spoofed IP</td>
<td>real</td>
<td>else</td>
<td>----</td>
</tr>
<tr>
<td></td>
<td>real IP</td>
<td>real</td>
<td>victim</td>
<td>$&lt;0,0,0&gt;$, $&lt;0,0,1&gt;$, $&lt;1,0,1&gt;$, $&lt;1,1,1&gt;$</td>
</tr>
<tr>
<td></td>
<td>real IP</td>
<td>real</td>
<td>else</td>
<td>$&lt;0,0,0&gt;$, $&lt;0,0,1&gt;$, $&lt;1,0,1&gt;$, $&lt;1,1,1&gt;$</td>
</tr>
</tbody>
</table>

To offer a mitigation method with no false positive, we employ the signature based packet filtering method at the source network identified by the edge detection process. The signature of each packet is marked by pivoted movement algorithm. The idea of the pivoted movement algorithm stems from the fact that the attacks quickly “pivot” one or more values of the flow information. In DDoS attacks, the source IP address pivots on the source MAC address and destination IP address. In order to detect the presence of pivoting in the traffic stream, our scheme generates a signature for each outgoing packet at the gateway router of the source network. The signature is simply a tuple consisting of three binary values: source IP address, destination IP address, and source MAC address. Each coordinate value in the signatures tells us whether the corresponding value in the packet was seen “recently” or not. “1” means the value was seen in finite time duration $D$ (e.g., $D = 1$ sec), while “0” means the value was not seen. For example, when the packet $A$ with source IP = 1.2.3.4, source MAC = $A:1:B:2:C:3$, and destination IP = 3.4.5.6 appears at time $t$, and the next captured packet $B$ with source IP = 5.6.5.6, source MAC = $A:1:B:2:C:3$, and destination IP = 3.4.5.6 appears within time interval $D$, then the pivoted movement algorithm generates the signature $<0,1,1>$ for packet $B$. This is because source MAC address $A:1:B:2:C:3$ and destination IP address 3.4.5.6 appear in packet $A$. But 5.6.5.6 was not used as the source IP address, and therefore $S_1 = “0”$. However, if packet $C$
with source IP = 1.2.3.4, source MAC = A:1:B:2:C:3, and destination IP = 3.4.5.6 appears at a different time interval $t'$, its signature will be $<0, 0, 0>$, since a packet data table is reconstructed and has no data. The packet data table for checking the “recently” value reconstructs in every time interval $D$. Every packet which has new values of $<S_i, S_m, D_i>$ is marked with the signature $<0, 0, 0>$, and it results in a false negative. According to this principle, in which the per-packet signature determines whether the packet constitutes attacks or not, a false positive cannot occur, since legitimate packets cannot be marked with the $<0, 1, 1>$ signature, based on the facts depicted in Table 1.

Table 2 The result of packet filtering based on pivoted movement. The attack packets are marked in red. Legitimate packets cannot have the $<0, 1, 1>$ signature, that is, there is no false positive. However, the attack packets can have the $<0, 0, 0>$ or $<1, 1, 1>$ signature and they are passed through the filtering module.

<table>
<thead>
<tr>
<th>TIME</th>
<th>Src IP (S)</th>
<th>Src MAC (S_m)</th>
<th>Dst IP (D)</th>
<th>Signature</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>210.119.57.52</td>
<td>00:24:8c:7a:0d:1e</td>
<td>203.253.146.168</td>
<td>&lt;0,0,0&gt;</td>
<td>passed</td>
</tr>
<tr>
<td>D+1</td>
<td>210.119.57.52</td>
<td>00:24:8c:7a:0d:1e</td>
<td>203.253.146.168</td>
<td>&lt;0,1,1&gt;</td>
<td>filtered</td>
</tr>
<tr>
<td>D+2</td>
<td>210.119.57.52</td>
<td>00:24:8c:7a:0d:1e</td>
<td>203.253.146.168</td>
<td>&lt;0,1,1&gt;</td>
<td>filtered</td>
</tr>
<tr>
<td>D+3</td>
<td>210.119.57.52</td>
<td>00:28:ca:a9:75:5d</td>
<td>129.46.224.211</td>
<td>&lt;1,0,0&gt;</td>
<td>passed</td>
</tr>
<tr>
<td>D+4</td>
<td>210.119.57.52</td>
<td>00:24:8c:7a:0d:1e</td>
<td>203.253.146.168</td>
<td>&lt;1,1,1&gt;</td>
<td>passed</td>
</tr>
<tr>
<td></td>
<td>210.119.57.69</td>
<td>00:08:9f:09:55:65</td>
<td>203.253.146.168</td>
<td>&lt;0,0,0&gt;</td>
<td>passed</td>
</tr>
<tr>
<td>E</td>
<td>210.119.57.77</td>
<td>00:24:8c:7a:0d:1e</td>
<td>203.253.146.168</td>
<td>&lt;0,0,1&gt;</td>
<td>passed</td>
</tr>
<tr>
<td>E+1</td>
<td>210.119.57.127</td>
<td>00:1d:a5:44:d7:92</td>
<td>184.155.203.16</td>
<td>&lt;0,0,0&gt;</td>
<td>passed</td>
</tr>
<tr>
<td>E+2</td>
<td>210.119.57.214</td>
<td>00:24:8c:7a:0d:1e</td>
<td>203.253.146.168</td>
<td>&lt;0,1,1&gt;</td>
<td>filtered</td>
</tr>
<tr>
<td>E+3</td>
<td>210.119.57.127</td>
<td>00:1d:a5:44:d7:92</td>
<td>184.155.203.16</td>
<td>&lt;1,1,1&gt;</td>
<td>passed</td>
</tr>
</tbody>
</table>

### 4 Experimental results

#### 4.1 Trace and simulation environment

We adopt two different network topologies, IPv4 Routed /24 topology dataset [16] and the IPv6 topology dataset [17] from CAIDA, for our simulations. Based on these link datasets, collected on October 6th, 2009, we first chose a router as the victim’s edge router, and then randomly chose multiple distinct routers originating from the victim as the attack sources on each topology. According to the experimental findings of [5], the typical distribution of the number of hops from clients to a server can be regarded as a Gaussian distribution with a mean of 16.5 and a standard deviation $\sigma = 4$. In our simulation experiment, we set the hop count between the victim and the sources as the same distribution as for the network topologies and the upper and lower bounds of the hop counts are set to 1 and 32. The summary of the common simulation environment is:

- Topologies = IPv4 routed /24 topology, IPv6 topology from CAIDA
- The number of attack sources ($N_a$) = 1000, each attack source contains several zombie hosts at a maximum of 10
- The number of zombie hosts ($N_z$) = 4000
- The number of legitimate sources ($N_l$) = 3000
- The attack traffic rate from one zombie host ($R_{ab}$) = 10 to 250 pps
- The hop count value of each attack source = Gaussian distribution ($M = 16.5, \sigma = 4$).

And, the simulation settings were:

- Setting A: the window size of the NetSCENE is 1 s and the IPv4 routed /24 topology dataset is used;
- Setting B: the window size of the NetSCENE is 5 s and the IPv4 routed /24 topology dataset is used;
- Setting C: the window size of the NetSCENE is 1 s and the IPv6 topology dataset is used;
- Setting D: the window size of the NetSCENE is 5 s and the IPv6 topology dataset is used.

#### 4.2 Figure of merit

To validate the edge detection results objectively, the NetSCENE can be verified using Pratt’s figure of merit (FM). It returns a number between 0 and 1 based upon the quality of the edge detection, with 1 being the best. In our approach, the FM can be considered as the false negative rate of source identification, since the false positive of identifying legitimate traffic as an attack source can hardly occur. To obtain a false positive result in our approach, at least several dozens of clients which are located in same sub-network would have to access the victim system simultaneously and this behavior would have to be continued for a while. However, this pattern cannot occur in natural behavior. The false negative possibility of FM is computed as

$$P_{FN-FM} = \frac{1}{\text{max}[I_i, I_f]} \sum_{i=1}^{I_i} \frac{1}{1 + \alpha d_i^{-2}}$$

where $I_d$ is the number of detected edges, $I_i$ is the number of ideal edges, $d_i$ denotes the distance from $i$-th actual edge to the corresponding detected edge. In our study, the distance is the number of pixel discrepancies between $I_i$ and $I_f$. $\alpha$ is a scaling constant set to 1/9 in this study.

Figure 5 shows the $P_{FN-FM}$ as a result of edge detection on the NetSCENE. $N_{syn}$ signifies $I_i$, which is the number of edges that should be detected on the NetSCENE in every time series. We varied $N_{syn}$ from 5 to 1000 in increments of 5. From the results, it can be seen that the $P_{FN-FM}$ differs greatly according to the $WS$. The graph of a 1 sec $WS$ is quite low compared to that of a 5 s $WS$, since the NetSCENE with a 1 sec $WS$ cannot hold a sufficient number of packets. This
means that the attack traffic patterns, the edges, do not appear clearly on the NetSCENE with a 1 sec ws.

![Fig. 5. Result of \( P_{FN-FM} \) vs \( N_{as} \)](image)

### 4.3 False negative in the pivoted movement

In this subsection, we analyze the possibility of false negative results in the pivoted movement algorithm. The proposed scheme incurs no false positives under the assumptions which are defined in section 3.1, since the legitimate traffic cannot have a \(<0, 1, 1>\) signature, but only the attack packet. In the pivoted movement algorithm to identify the zombie host, the signatures \(<0, 0, 0>\) and \(<1, 1, 1>\) can be attack packet signatures and can result in a false negative. The false negative with a signature \(<0, 0, 0>\) occurs just one time on each zombie host, since it is possible to identify the zombie hosts by their MAC address and block their packets heading to the victim system. Therefore, the possibility of a false negative caused by a \(<0, 0, 0>\) signature is slight. The other possibility of a false negative, caused by the signature \(<1, 1, 1>\), occurs due to the overlapping of IP address choices. The zombie host may choose again the specific address bits within its subnet range. This possibility occurs due to the overlapping of IP address choices. The false negative with a signature \(<0, 0, 0>\) is slight. The other possibility of a false negative caused by the signature \(<1, 1, 1>\) occurs due to the overlapping of IP address choices. This possibility occurs due to the overlapping of IP address choices.

\[
P_{FN-<1,1,1>} = \frac{P}{TL}
\]

To analyze the \( PFN-<1,1,1> \) in the pivoted movement algorithm, we observed 1000 routers, the attack sources, which are processing the pivoted movement. Each attack source has several zombie hosts, with a maximum of 10, and each zombie host generates attack packets from 100 to 250 pps. The simulation results are shown in Figure 6. The false negative ratios are acceptably low for all cases of pivoted movement processing. Through our simulation experiments, we were able to observe that, broadly speaking, a false negative in the pivoted movements often occurs in the sub-networks, which have a long prefix. In addition, there is no false positive in pivoted movement processing, since the legitimate packets from clients cannot have the \(<0, 1, 1>\) signature, which should be filtered out as an attack packet. In this simulation, the average rate of egress traffic from one sub-network was 7743.99 pps and of attack packets from one sub-network 724.29 pps.

![Fig. 6. \( P_{FN-<1,1,1>} \) of the source routers](image)

### 4.4 Analysis of attack mitigation delay

We define the attack mitigation delay as the time interval between the start time of an attack and the termination time of the mitigation process. The mitigation process is terminated when the DDoS attack sources are no longer detected on the NetSCENE. We first define the following variables:

- \( ws \): The given time window size of packet capturing for the NetSCENE (e.g., 1 sec)
- \( N(i,t) \): The number of identified sources in \( t \)-th time slot, where \( t = 1, 2, \ldots \);
- \( H(i,t) \): The hop count of the \( i \)-th attack sources in \( t \)-th time slot, where \( i = 1, \ldots, N(i,t) \);
- \( D_{det}(i,t) \): The detection delay for identifying the \( i \)-th attack sources in \( t \)-th time slot with \( ws \) time-window size after the attacks have reached the victim system. If attacks from the \( i \)-th attack sources arrived at time \( t-2 \) and it is identified at \( t \)-th time slot, we have \( D_{det}(i,t) = (t-(t-2)) \times ws \);
- \( D_{path}(i,t) \): The path delay for sending an instruction from the victim to the source router on \( i \)-th source identified in the \( t \)-th time slot (including transmission delay, propagation delay, queuing delay, and processing delay on each of routers). The \( p(j) \) (e.g., 4.8 ms/s) is the propagation delay from the \((j-1)\)-th hop router to the \( j \)-th hop router. When only the propagation delay is considered, we have \( D_{path}(i,t) = \sum_{j=1}^{H(i,t)} p(j) \);


- $T_{\text{path}}(i,t)$: the required time at the $i$-th attack source router that is identified in the $t$-th time slot between the arrival time of the instruction at the source router and the time when the first $<0, 1, 1>$ signature appears at the filtering module;

- $T_{\text{fwd}}(i,t)$: the time required to forward all attack packets generated before the filtering process starting at the source router on the $i$-th source identified in the $t$-th time slot.

Assuming the attack packets from the source $i$ are reached at time slot $t-2$, and the source $i$ is identified at time slot $t$ with $ws$, then we can infer that the first step takes time $[D_{\text{arr}}(i,t,ws)+D_{\text{path}}(i,t)]$ to identify the source and relay an instruction to the source router on the attack source $i$. The next step takes time $T_{\text{fwd}}(i,t)$ to change its filtering status on the router. Finally, the effectiveness of the filtering attacks takes time $T_{\text{fwd}}(i,t)$ at the victim system. Hence the mitigation delay of one source $i$ is:

$$T = D_{\text{arr}}(i,t) + D_{\text{path}}(i,t) + T_{\text{fwd}}(i,t) + T_{\text{fwd}}(i,t)$$

$$= 2ws + \sum_{j=1}^{H(i,t)} p(j) + T_{\text{fwd}}(i,t)$$

(7)

The mitigation delay when multiple attack sources exist can be computed similarly. However, because it synchronizes the identification of $N(i)$ attack sources at the $t$-th time slot and performs the filtering process at multiple source routers, the time taken in each process will constitute the longest time. If the source identifying process using the NetSCENE is performed $q$ times during the mitigation process, and the set of the sources identified at time slot is $S_q$, then the total detection delay can be expressed as follows:

$$T_{\text{det}} = \max \{ [i-1]ws + D_{\text{arr}}(i,t) + M_i \}$$

(8)

where, $T_{\text{det}} = \max \{ [i-1]ws + D_{\text{arr}}(i,t) + M_i \}$,

$$\begin{cases} S_i = S_i, \\ S = S - \bigcup_{i=1}^{q} S_i, \text{ when } i > 1 \end{cases}$$

To simulate the attack mitigation delay, $T_{\text{delay}}$, we used an actual network delay distribution measured from the Internet [3]; the propagation delay between two nodes is applied to our variables, $D_{\text{path}}(i,t)$, and $T_{\text{fwd}}(i,t)$. We illustrated simulations with each setting. The results are depicted in Figure 7, which shows the transition of the traffic rate at the victim’s networks and the effectiveness of the mitigating attack. The results show that the proposed scheme can filter most attack packets and the traffic rates at the victim’s networks are stable even at the moment when the number of DDoS attacks peaks. The results show that the simulation with Settings A and C can reduce the attack mitigation delay in the early phase, as is to be expected, since the delay in detecting the attack source should be less with a $1\ s$ $ws$ than with a $5\ s$ $ws$. However, after $30\ s$, the number of throttled zombie hosts in the simulation with Settings B and D caught up to those with Settings A and C, and the total number of throttled zombie hosts is more than these.

**Fig. 7. Simulation results with each setting**

### 5 Conclusion

We presented a novel approach to detecting the source of spoofed DDoS attacks. It employs NetSCENE to reveal the DDoS attacks and their sources through image processing; moreover, it identifies zombie hosts using the signature-based pivoted movement, which is a pattern extraction algorithm. The proposed scheme is in line with the ideal DDoS countermeasure framework in which DDoS attack detection is performed at the victim system, and the mitigation is performed at the source of the attack. Our scheme is validated using the false negative rate in the figure of merits and pivoted movement, and the attack mitigation delay. The simulation results indicate that the scheme has several advantageous features, including: (1) it can identify the attack sources and even the zombie hosts simultaneously and quickly; (2) it can filter out only the attack packets without disturbing legitimate traffic; (3) it does not need any module on the network device, such as packet marking and deployments; (4) it is robust against IP spoofing; (5) it provides useful traffic visualization and analysis;
and (6) it incurs no false positive and a low rate of false negatives.

References


