Single Packet ICMP Traceback Technique using Router Interface

M. VIJAYALAKSHMI AND S. MERCY SHALINIE
Department of Computer Science and Engineering
Thiagarajar College of Engineering
Madurai, 625015 Tamilnadu, India
E-mail: {mviji; shalinie}@tce.edu

In the modern technological world, with the increasing dependency on Internet the security threats are on the rise. Distributed Denial of Service (DDoS) attack is one of the biggest threats. The attackers tend to exhaust the network resources, while ingeniously hiding their identity, making the defense process extremely difficult. Many researchers have proposed various solutions to traceback the true origin of attack. Among them Internet Control Message Protocol (ICMP) traceback was considered an industry standard by Internet Engineering Task Force (IETF). ICMP Traceback (ITrace) does not require any change in the existing infrastructure. However it consumes considerable bandwidth and requires a large number of packets to traceback an attacker. This work proposes a Single Packet ICMP Traceback technique using Router Interface (SPITRI). It traces the origin of flooding attack with a single ICMP packet. The bandwidth overhead incurred by SPITRI is several times lesser than ITrace. SPITRI was simulated over the CAIDA Ark dataset. It can traceback the attackers with high accuracy, with zero false positive and zero false negative result. The efficacy of the proposed scheme is demonstrated by simulating and comparing it with ITrace, and the latest router interface based single packet traceback scheme.

Keywords: DDoS attack, IP spoofing, IP traceback, single packet, packet marking, ICMP traceback

1. INTRODUCTION

Internet is making things easier by bringing everything to the users with a click of a mouse. But it does not provide a highly secure environment, even if the servers are protected by firewalls and Intrusion Detection Systems (IDS). Anonymous attackers tend to disrupt the service offered by a network or server, thereby disrupting business and availability of the system. Distributed Denial of Service (DDoS) attack is one such attack which disrupts the service, by exhausting the network bandwidth and system resources. The attacker uses spoofed source IP address, and frustrates the victim from deploying any defense measure.

According to the survey conducted by Arbor Networks [1], DDoS attacks are becoming bigger, faster and more complex year after year. The effect of denial of service attack is severe, because the attacker hides his identity, which makes the source based defense impossible. This motivates the research on IP traceback. IP traceback is a method of identifying the origin of a packet without relying on the source IP address of the packet.

Numerous researchers have proposed various potential traceback solutions, such as...
packet marking, packet logging, hybrid method and ICMP traceback. Of these ICMP traceback [2] was considered more practical and an industry standard by IETF. Most of these existing traceback schemes including ICMP traceback, require a large number of packets to traceback an attacker. ICMP traceback requires out of band message. The messages generated for the purpose of traceback itself will pollute the network with additional packets during large scale DDoS attacks. As they rely on numerous packets to trace an attack origin, there is the prospect of producing false positives when the number of attackers increases. This paper aims to traceback an attacker using a single ICMP packet.

Several efficient schemes [3-7] have the capability to traceback even a single packet. But these invariably tax the routers in attack path with additional storage and computation. Moreover such schemes require computation in every packet which the router forwards. It is highly impractical. These single packet traceback schemes require hop by hop back traversal towards the attacker. They are not capable of identifying the attacker from the victim itself. So the traceback process again requires computation in every router in the attack path. This additional storage and computation requirement at the routers affect the throughput of the routers. Besides this, most of the single packet traceback schemes overload the identification field in an IP packet, overriding its conventional usage, which in turn harm the fragmented traffic. Though logging based single packet traceback [3] schemes avoid overriding the identification field, they demand heavy storage at the routers and end up as false positives due to collisions in the log. To reduce the storage cost at the routers some researchers have proposed [5-8] the use of router interface ID number instead of IP address. Although the storage cost is significantly reduced in this router interface based techniques, it has not been completely eliminated. In spite of additional storage requirement, they involve the routers in computation of trace information of every packet it forwards. Furthermore they trouble the routers during the traceback of attacker also. Hence, the existing IP traceback schemes either require numerous packets to identify an attacker, or incur significant storage, bandwidth and computation overhead at the routers or overload the IP packet fields, thus affecting normal transactions. This paper, attempts to overcome the shortcoming in ITrace and router interface based traceback methods and propose a log less Single Packet ICMP Traceback technique using Router Interface (SPITRI) to trace the origin of flooding attack with a single ICMP packet, with minimal computation cost and high accuracy.

The contribution of the paper can be summarized as follows:

- A traceback scheme named SPITRI (Single Packet ICMP Traceback scheme using Router Interface) is proposed which
  - can traceback with a single ICMP packet, thereby reducing the bandwidth overhead caused by ITrace
  - requires no storage at the routers
  - can traceback the attack origin from the victim itself.
  - can handle major DDoS attack.
  - does not require changes to the existing packet processing at the routers
  - can traceback with zero false positive and zero false negative
- The proposed system was simulated and analyzed using CAIDA Ark dataset [9]
- The proposed system was also compared with the related traceback schemes.
Section 2 discusses the related research work. Section 3 expounds the proposed Single Packet ICMP Traceback technique using Router Interface (SPITRI). Section 4 details on the experimental setup and Section 5 analyses and compares the proposed technique with ITrace and the state-of-the-art router interface based traceback technique using mathematical evaluations and experimental results. Section 6 summarizes the findings and discusses the future enhancements on the proposed work.

2. RELATED WORK

IP traceback schemes can be broadly classified into Link Testing, Packet Marking, Packet Logging, Hybrid Approach and ICMP based technique.

Link Testing requires testing the link hop by hop upwards to the attacker by checking the packet signature [10] or by thrusting packets [11] into the network to identify the packet drop rate. This method consumes time and bandwidth and the attack has to be alive till it traces the attacker. Packet marking is the technique used to identify the attacker by writing the attack path details in the rarely used fields of an IP packet. It neither pollutes the network nor causes storage overhead, but it requires modification in packet processing. It generally uses the existing IP header fields by overriding its conventional purpose, which requires appropriate modifications for the normal functioning of Internet. In deterministic packet marking [12, 13] the packets are marked in a deterministic manner. It identifies the border router of an attacker domain which is sufficient to mitigate an attack. Probabilistic packet marking [8, 14-17] marks the packet with a probability and attempts to find the attack path from the collected packets. Though it can identify the attack path, it takes a longer time to reconstruct the attack path when the number of attackers increases. When the number of attackers or the distance between source and destination increases the number of false positives also increases. Moreover it requires heavy storage and processing at the victim end during the traceback process. The traceback scheme proposed by Al-Duwairi and Govindaraju [18] has reduced the number of packets required to traceback to the attacker. It still requires more than one packet. Numerous researchers have proposed a number of efficient methods to traceback with a single packet [3-7]. Packet logging schemes [3, 19] stores the packet digest in the routers in the attack path. When the attack is detected the logged data is mined to retrieve the required information. Though these methods are efficient enough to traceback even a single packet attack, they entail exorbitant storage overhead at the routers, affecting its forwarding capacity. To reduce the storage requirement retaining the single packet traceback capability the researchers proposed hybrid method, which is a combination of packet marking and packet logging. Despite the fact that the amount of packets logged is reduced, it still creates significant storage overhead at the routers. Chao Gong and K. Sarac [4] proposed a hybrid method which has reduced the storage overhead, but it still needs to log the packets at certain routers. To further reduce the storage at the routers, router interface ID number has been used as the primary key to traceback, rather than the router IP address [5-8]. Chen et al. [8] proposed RIM where the router interface is marked in a probabilistic manner and the victim collects the required amount of packets to reconstruct the attack path. It does not cause storage overhead but the false positive ratio goes high when the number of attackers increases. Malliga and Tamilarasi have
proposed hybrid methods namely MRT [5] and MORE [6], which can identify the attacker with a single attack packet. The problems in MRT and MORE were analyzed in detail and solved by Ming-Hour Yang and Ming-Chien Yang [7], using RIHT. RIHT has reduced the storage overhead and computation time at the routers, and logged the packets only when the marking field overflows. It causes significant storage overhead at the routers. They also require modification in the packet processing done by routers since they override the existing fields and mark their trace information in the packets.

ICMP based scheme (ITrace) suggested by Bellovin [2] proposed that the routers in the attack path generates ICMP packets with a probability ‘p’ carrying the upstream and downstream link details along with the traced data packet. Then the ICMP packets are collected at the victim site and connected to form attack tree. ICMP based traceback does not require changes in the existing infrastructure, so this was considered an industry standard by IETF. Although it does not require changes in the existing infrastructure, it causes significant bandwidth overhead by insisting the routers in the attack path to generate an ICMP traceback message. A single traceback process is dependent on multiple ICMP traceback messages. DDoS attack would generally employ thousands of attackers targeting a host. Each router in the attack path would generate an ICMP packet which, in turn, consume a considerable bandwidth and it is obvious that when a traceback process is dependent on multiple packets it would lead to considerable amount of false positives. Moreover to traceback 1000 attackers a victim is expected to receive at least 138 million packets [17]. Enhanced ICMP traceback scheme called ITrace-CP [20] was proposed, which focused on traceback with a single ICMP packet. This scheme requires the router to buffer the IP packet digests forwarded by the router for a limited time period to check whether the traceback message also has chosen the path of the IP packet. Hence like the logging and hybrid methods this method also demands storage at the routers which is not recommended for better throughput.

Hence the proposed SPITRI has attempted to trace an attacker with a single ICMP packet using router interface. Thus it reduced the bandwidth overhead produced by ITrace, and has eliminated the storage overhead incurred by other single packet router interface based traceback methods. It does not demand any change in the processing of IP packets at the router and does not override the IP header fields affecting its normal functionality. Unlike any other single packet traceback scheme, it can identify the path to reach the origin of flooding attack from the victim itself.

3. OVERVIEW OF SPITRI

As in ITrace, the proposed SPITRI also uses ICMP packet to carry the trace information of the packets. Hence the existing infrastructure need not be modified and the normal operation of IP packets will not be disturbed. Unlike ITrace, instead of appending the upstream and downstream IP address of the routers, the proposed SPITRI manipulates the trace information with the router interface ID number like other router interface based traceback schemes. The overview of the proposed technique is given in Fig. 1. It consists of three phases. First is generation of ICMP traceback message at the border router. Second, construction phase where the traceback data is constructed by repeated update at all the routers in the path. Thirdly the reconstruction phase comes into picture...
when the victim would like to find the origin of a packet. At the destination, the path to reach the attacker can be retrieved from the trace information available in an ICMP packet, by applying the traceback algorithm. It results in a sequence of router interfaces to be traversed to reach the attacker. SPITRI differs from ITrace by the capability to traceback with a single ICMP packet. As a result the bandwidth overhead and traceback time is also reduced.

**3.1 Basic Requirements**

The proposed SPITRI was motivated by some assumptions made by the existing router interface based traceback approaches. The assumptions are as follows:

- Every router builds a router interface table and assigns the interface ID from 0 to \( n - 1 \), where \( n \) is the total number of links. Even ITrace uses interface identifier in Forward/Backward Link element using character string format.
- The traceback message construction algorithm is implemented at all the routers.
- Routers are not generally compromised.
- Routers can identify whether the packet is from a local network or any other router.

**3.2 ICMP Traceback Message**

As in ITrace proposed by Bellovin *et al.* [2] the traceback information is loaded in an ICMP packet in the proposed SPITRI. The ICMP message used is shown in Fig. 2. As mentioned in ITrace the type field will be of traceback type and code is always set to zero.

- Generation of ICMP Traceback Message
- Construction of Traceback Message
- Construction of Traceback Message
- Reconstruction of Path till the Attacker LAN

![Fig. 1. System overview.](image)

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 2. ICMP traceback message.

In ITrace, the message body consists of sequence of elements in Type length value format to store the details of

- Back Link
- Forward Link
- Timestamp
- Traced Packet Contents
- Probability
Each of these elements may have sub elements. In the proposed SPITRI message most elements are eliminated. The SPITRI message holds only two elements, with no sub elements.

- Path Information (512 bits)
- Traced Packet Content (at least 160 bits)

The job of Back Link, Forward Link, Timestamp, Probability, and Router Id elements of ITrace message is taken care of by a single element called Path Information element in SPITRI. Thus the ICMP traceback message size is reduced. Path Information element is the actual traceback key with which the path will be reconstructed. It will be updated by every router in the path. The size of Path Information element should be at least 512 bits. Similar to ITrace, Traced Packet Content element generally holds the content of the traced (original) packet and contains at least the 20 byte IP header of the traced packet. This is used to associate the ICMP packet to the original packet.

### 3.3 Generation and Construction of SPITRI Traceback Message

Similar to ITrace, the border router generates an ICMP traceback (SPITRI) message with a probability ‘p’ and sends to the destination of the traced packet. The algorithm used for generating and constructing traceback messages is given in Fig. 3.

<table>
<thead>
<tr>
<th>Algorithm to generate and update traceback message</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Begin</td>
</tr>
<tr>
<td>2. Let $R = {R_i}$ be a set of routers in the path, $UIId$ be Upstream Interface Id through which the traceback message entered the router, $PI.value_{new}$ be the newly computed value of the Path Information element of SPITRI message, $PI.value$ be the current value of Path Information element of SPITRI message.</td>
</tr>
<tr>
<td>3. At each router $R_i$</td>
</tr>
<tr>
<td>4. If $R_i$ is border router then</td>
</tr>
<tr>
<td>Generate an ICMP traceback message with probability ‘p’</td>
</tr>
<tr>
<td>$PI.value = UIId$</td>
</tr>
<tr>
<td>else</td>
</tr>
<tr>
<td>$PI.value_{new} = \left( \frac{1}{PI.value + 2} \right) + UIId + 1$</td>
</tr>
<tr>
<td>5. End If</td>
</tr>
<tr>
<td>6. $PI.value = PI.value_{new}$</td>
</tr>
<tr>
<td>7. Forward SPITRI message to next router.</td>
</tr>
<tr>
<td>8. Go to Step 4</td>
</tr>
<tr>
<td>9. End</td>
</tr>
</tbody>
</table>

Fig. 3. Traceback message construction algorithm.
It initializes the Path Information element as the interface ID number through which the IP packet arrived. Border router is the router which is directly connected to a local network. This is the ingress router for the packets going out of that network. Unlike ITrace, only the border router generates an ICMP (SPITRI) message. Other intermediate routers in the path need not generate new ICMP message. They only update and forward the SPITRI messages generated by the border routers. When the next intermediate or core router receives the ICMP traceback message it identifies it as a SPITRI message, by looking in the type of the ICMP message. Then it updates the Path Information element with a new value using the formula given by Eq. (1). This formula was derived in such a way that path information value has a cumulative effect of all the incoming interface id through which the packet traversed to reach the destination. Hence, during traceback process, from the victim the complete sequence of upstream Interface ID to reach the attacker could be retrieved from its corresponding Path Information Value.

\[ PI.value_{\text{new}} = \left( \frac{1}{PI.value + 2} \right) + UIld + 1 \]  

(1)

The new Path Information value \( PI.value_{\text{new}} \) is computed from the existing Path Information value \( PI.value \) and the upstream Interface ID \( UIld \). \( UIld \) is the interface ID through which the SPITRI message entered that corresponding router.

Consider the network shown in Fig. 4. The dotted line denotes the path of the SPITRI message. The network has numerous routers.

![Fig. 4. Sample network.](image-url)
The links from each router is provided with a locally unique interface identifiers starting from 0 to \( n - 1 \), where \( n \) is the total number of links available at the router. As mentioned earlier border router is the router which is directly connected to a local network, which will be the ingress router for the packets going out of that network. Core routers receive packets from other routers. It will be an intermediate router in a network path. A border router can also be a core router. For example R1 is the border router for the packets from the attacker, but if packets from LAN1 choose to travel through R2-R1-R3-R7 then R1 is a core router.

For understanding the working of algorithm, let us assume that border router R1 is generating an ICMP traceback (SPITRI) message with a probability ‘\( p \)’. The Path Information element of that SPITRI message is initialized with the Upstream Interface ID through which the traced packet entered, which in this case is 2. Let us assume that the SPITRI message traverse through the path of R1-R3-R7-R8-R10-R11-R12-R13-R14-R15 to reach the victim. When the SPITRI message reaches the next core router R3 through the interface 3, it once again computes the Path Information value using Eq. (1), which is \((1/(2+2))+3+1= 4.25\). Now the Path Information value 2 is replaced with 4.25 and forwarded to the next router R7. The packet enters R7 through the interface 4 and the Path Information value is computed again using the Eq. (1) as \((1/(4.25+2))+4+1=5.16\). As a result the Path Information value is replaced as 5.16 and forwarded to next router R8. This process is repeated till it reaches the victim. The same function is reiterated at all the routers in the path. Table 1 illustrates the Path Information value update at each router in the path R1-R3-R7-R8-R10-R11-R12-R13-R14-R15. When the traceback message reaches the victim, the Path Information element holds the value 1.014277 (refer Table 1) which serves as the key to unlock the attack path.

<table>
<thead>
<tr>
<th>Hop</th>
<th>Router</th>
<th>Interface</th>
<th>Path Information Value when ICMP packet entered the router (( PI_{\text{value}} ))</th>
<th>Updated Path Information Value when ICMP packet leaves the router (( PI_{\text{value}_{\text{new}}} ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R1</td>
<td>2</td>
<td>Any random value</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>R3</td>
<td>3</td>
<td>2</td>
<td>4.25</td>
</tr>
<tr>
<td>3</td>
<td>R7</td>
<td>4</td>
<td>4.25</td>
<td>5.16</td>
</tr>
<tr>
<td>4</td>
<td>R8</td>
<td>5</td>
<td>5.16</td>
<td>6.139665</td>
</tr>
<tr>
<td>5</td>
<td>R10</td>
<td>2</td>
<td>6.139665</td>
<td>3.122855</td>
</tr>
<tr>
<td>6</td>
<td>R11</td>
<td>100</td>
<td>3.122855</td>
<td>101.1952</td>
</tr>
<tr>
<td>7</td>
<td>R12</td>
<td>34</td>
<td>101.1952</td>
<td>35.00969</td>
</tr>
<tr>
<td>8</td>
<td>R13</td>
<td>21</td>
<td>35.00969</td>
<td>22.02702</td>
</tr>
<tr>
<td>9</td>
<td>R14</td>
<td>67</td>
<td>22.02702</td>
<td>68.04162</td>
</tr>
<tr>
<td>10</td>
<td>R15</td>
<td>0</td>
<td>68.04162</td>
<td>1.014277</td>
</tr>
</tbody>
</table>

As shown in Table 1, each path would correspond to a value, so there will not be any collision of values. Two packets would hold same value only when they traverse through the same path. Since a constant ‘1’ is divided by the path information value it would result in a value which is path specific. So packets from different path will always have different values. Fig. 5 illustrates that Path Information value of SPITRI message is
path specific. SPITRI message is generated from two different border routers, R1 and R2. The path information value (PI.value) remains same till it reaches router R5. At router R5 the path Information value is changed since different routers are connected to different interfaces. So the Path Information value is path specific and no two packets from different paths possess the same Path Information value.

Fig. 5. Illustration of Path Information value update from different border routers.

3.4 Path Reconstruction

Now the victim has received the ICMP traceback (SPITRI) message along with the attack packets. Every traceback message has its own key to unveil the path it traversed. Assume that an Intrusion Detection System (IDS) is running on the victim machine. When it detects an attack, path reconstruction process is initiated. The trace back packet is identified by comparing it with the value of Traced Packet Content element received from the SPITRI message. The Path Information element of that SPITRI message contains the final value after applying Eq. (1) \( n - 1 \) times, for a path containing \( n \) routers. The path to reach the attacker border router and attacker LAN is retrieved from the Path Information value of the SPITRI message itself. So even if the source IP address of the attack packet is spoofed, the Path Information value of the SPITRI message would be adequate to derive the path to reach the attacker border router and attacker LAN. The attack packet is associated to the SPITRI message by storing first few bytes of the attack packet in the Traced Packet Content element of the SPITRI message. Hence by that way logging at the intermediate routers is avoided. By matching the Traced Packet Content element of the SPITRI message with the attack packet the corresponding SPITRI message could be identified and thereafter with the Path Information element, attacker LAN could be traced out by employing the traceback algorithm shown in Fig. 6.

From the TimeToLive (TTL) field the number of hops traversed by the traceback message is computed. The process of identifying the upstream interface ID and the previous value of the Path Information element is repeated till hop count value becomes 1. The upstream interface is computed by Eq. (2)

\[
UIId = \text{floor}(PI.value) - 1.
\]
The router connected to the upstream interface ID is the next router in the path from the victim to the attacker. To discover further upstream routers, the next upstream interface ID from that router has to be identified. To identify this, previous Path Information value of the traceback message is found by Eq. (3)

\[ PI_{value_{previous}} = \frac{1}{PI_{value} - floor(PI_{value})} - 2 \]  

where, \( PI_{value_{previous}} \) denotes the previous Path Information value during the construction phase, \( PI_{value} \) denotes the current Path Information value. This process of identifying \( UIId \) is continued number of hop times. The output of this traceback process will be a sequence of upstream interface ID that has to be visited to reach the attacker from the victim.

**Traceback Algorithm**

1. Begin
2. Let \( UIId[n] \) be Upstream Interface Id Array through which the packet came crossing \( n \) routers, \( PI_{value} \) be the current value found in the Path Information of the packet, \( PI_{value_{previous}} \) be the Path Information value computed by the router at the previous hop.
3. Identify the Traceback Message
4. From the TTL of Traceback Message compute the Hop Count
5. While (Hop Count! = 1)
   \[ UIId[i] = floor(PI_{value}) - 1 \]
   \[ PI_{value_{previous}} = \frac{1}{PI_{value} - floor(PI_{value})} - 2 \]
   \( PI_{value} = PI_{value_{previous}} \)
   \( i++ \)
   \( HopCount -= \)
6. } 
7. \( UIId[i] = PI_{value} \)
8. Print the Upstream Interface Id array \( UIId[] \)
9. Generate a Filter request packet traveling through \( UIId[] \)
10. End

Consider the network illustrated in Fig. 4, the victim begins the traceback process with an input of Path Information value, which in this case is \( PI_{value} = 1.014277 \) as shown in Table 1. Now, upstream interface ID from the victim’s router is determined using (2) as \( UIId = floor(1.014277) – 1 = 0 \). ‘0’ is connected to R14 so the next router in the path is R14. At this instant the previous Path Information value is calculated so that the next upstream interface ID can be computed from that. \( PI_{value_{previous}} = (1/ (1.014277 – floor(1.014277))) – 2 = 68.04162 \). Now considering this as the current Path Information value the upstream interface ID can be identified using the Eq. (2). \( UIId = floor(68.04162) – \)
1 = 67. The router connected to ‘67’ is R13. This process is continued till the hop count limit is reached. Finally the list of interface IDs to be visited from the victim’s router is displayed. Thus the path to reach the attacker is traced back with a single ICMP traceback message from the victim itself without disturbing any of the routers in the attack path. Table 2 illustrates the operation of traceback algorithm for the sample network shown in Fig. 4. It shows how the SPITRI message travelled through the path R1-R3-R7-R8-R10-R11-R12-R13-R14-R15 is traced back from the victim.

Table 2. Illustration of traceback.

<table>
<thead>
<tr>
<th>Hop Count</th>
<th>Path Information Value in the ICMP packet Pl.value</th>
<th>Previous Path Information Value Pl.value\textsubscript{previous}</th>
<th>Upstream Interface Id UIId</th>
<th>Router connected to UIId</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.014277</td>
<td>68.04162</td>
<td>0</td>
<td>R14</td>
</tr>
<tr>
<td>9</td>
<td>68.04162</td>
<td>22.02702</td>
<td>67</td>
<td>R13</td>
</tr>
<tr>
<td>8</td>
<td>22.02702</td>
<td>35.00969</td>
<td>21</td>
<td>R12</td>
</tr>
<tr>
<td>7</td>
<td>35.00969</td>
<td>101.1952</td>
<td>34</td>
<td>R11</td>
</tr>
<tr>
<td>6</td>
<td>101.1952</td>
<td>3.122855</td>
<td>100</td>
<td>R10</td>
</tr>
<tr>
<td>5</td>
<td>3.122855</td>
<td>6.139665</td>
<td>2</td>
<td>R8</td>
</tr>
<tr>
<td>4</td>
<td>6.139665</td>
<td>5.16</td>
<td>5</td>
<td>R7</td>
</tr>
<tr>
<td>3</td>
<td>5.16</td>
<td>4.25</td>
<td>4</td>
<td>R3</td>
</tr>
<tr>
<td>2</td>
<td>4.25</td>
<td>2</td>
<td>3</td>
<td>R1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Not computed since hop count = 1</td>
<td>2</td>
<td>Attacker</td>
</tr>
</tbody>
</table>

The input of the process is the value of Path Information element found in the packet (1.014277) and the output is the sequence of upstream interface ID to be visited from the victim’s router. It gives an output of 0-67-21-34-100-2-5-4-3-2. When the Hop Count limit has reached the traceback process is stopped by giving out the interface to which the attacker is connected. The victim need not maintain the topology information or each router’s interface tables. A mitigation agent or a filter request packet can be directed to reach the attacker border router following the identified interface ID sequence.

Hence the path to reach the attacker is obtained from the victim itself with a single ICMP traceback message with very low computation. It is to be noted that it gives only the path that the ICMP traceback (SPITRI) message traversed and not the attack path. However, Paxson [21] has shown that above 90% of the routing paths remain stable in Internet for hours. As a result, most of the time the path traversed by the ICMP traceback (SPITRI) message and the attack packet would be the same. Suppose a SPITRI message traverse in a different path to reach the victim, the resultant sequence of upstream interface ID would still, identify the attacker’s border router and the attacker LAN, from which the attack was originated. This is because the ICMP traceback (SPITRI) message was generated from border router of the attacker only and the Path Information value is initialized with the interface ID through which the attack packet entered the border router.

Like in ITrace it does not require numerous packets to traceback a large scale DDoS attacks. One SPITRI message is adequate to traceback one attacker. Router interface
based traceback methods like MRT, MORE and RIHT can also traceback with a single packet, but they demand additional storage at the routers. This would degrade the performance of routers. They also utilize the IP header fields to mark the traceback data, overriding its conventional purpose. They burden the routers in the attack path by marking every packet it forwards, and disturbs the routers during the traceback process also. Whereas SPITRI is capable of tracing the attacker from the victim itself with a single traceback message, without demanding additional storage or disturbing the existing infrastructure. MRT, MORE and RIHT can traceback only till the edge router, as they initialize the marking field to ‘0’ at the attacker’s border router in the packet marking process. But SPITRI identifies the attacker or the attacker LAN as it initializes the Path Information with the interface ID through which the attack packet entered.

4. EXPERIMENTAL SETUP

In the following simulations, the environment consists of a PC with Intel i4 930 3 GHz, 2 G RAM and Ubuntu. The functionality of the proposed SPITRI was tested with two experimental set up. First it was simulated using network simulator NS2 [22] integrated with BRITE [23]. BRITE is a topology generating tool which generates topologies that reflect many aspects of actual internet topology. The proposed system was tested with network topology (shown in Fig. 7) consisting of 100 nodes generated by BRITE topology generator. In this experimental setup, a node designated as victim was attacked by different number of end nodes.

In order to test the functionality of SPITRI in actual Internet, it was simulated over the CAIDA Ark dataset [9] which gives the path from a single monitor of CAIDA to 190984 destination nodes. It consists of 327262 nodes and 2794611 links. All the destination nodes were considered as attacker and the monitor was considered as victim.
5. PERFORMANCE ANALYSIS

The proposed work tracebacks with the help of ICMP traceback message using router interface. The efficiency of the proposed system is analyzed by simulating and comparing it with both (1) Original ICMP traceback proposed by Bellovin [2] and (2) the state-of-the-art router interface based approach RIHT [7]. The proposed SPITRI is also qualitatively compared with ITrace-CP [20] which is also capable of tracing with a single ICMP packet.

5.1 Comparison to Original ICMP Traceback

SPITRI is evaluated by comparing it with the original ICMP traceback using the following performance metrics

1. Number of ICMP Packets Required to Traceback
2. Number of Packets Required to Reconstruct the Full Path.
3. Path Reconstruction Time
4. Bandwidth Overhead
5. Accuracy

5.1.1 Number of ICMP packets required to traceback

Traceback methods dependant on more number of packets is normally time consuming and have the prospect of producing false positives. As mentioned by Lee et al. [20], ITrace message with forward or backward link enables the victim to identify 2 routers in an attack path, and ITrace message with both links facilitate the victim to identify 3 routers in the attack path. Hence if an attack path is of ‘h’ hops then at least ‘h/2’ ITrace messages (Forward or Backward Link) would be required to identify one attacker. Likewise at least ‘h/3’ ITrace messages (Both Links) would be required to identify one attacker. Minimum number of ITrace messages (Forward or Backward Link) and ITrace messages (Both Links) required to trace ‘n’ attackers is denoted by Eqs. (4) and (5) respectively.

\[ N_{ITrace} \approx \sum_{i=1}^{n} \frac{h_i}{2} \]  
\[ N_{full} \approx \sum_{i=1}^{n} \frac{h_i}{3} \]  

SPITRI needs only ‘1’ ICMP packet (SPITRI message) generated from the border router to traceback one attacker. From a single packet it is capable of tracing the path till the border router and the interface to which the attacker is connected. So to trace ‘n’ attackers the number of SPITRI message required is given by Eq. (6)

\[ N_s = n. \]  

Since the number of SPITRI message needed is not dependent on the path length of the attackers and only on the number of attackers, it is apparent that the number of ICMP packets required using SPITRI is always lesser than the one needed in ITrace method.
5.1.2 Number of packets required to reconstruct the full path

This metric evaluates how many packets arrive at the victim to trace an attacker. This includes both the ICMP packet and data packet. If more number of packets is required for reconstruction of attack path then the traceback is delayed accordingly. SPITRI requires only one ICMP packet from the border router of the attacker to traceback the entire path whereas ITrace requires the traceback message from multiple hops in the attack path.

Since SPITRI needs only one message to construct the path till attacker, for an attack path of length ‘\(h\)’ hops the probability (\(P_s\)) of reconstructing the full path till the attacker with ‘\(n\)’ IP packets using SPITRI is given by Eq. (7)

\[
P_s = 1 - q^n.
\]  

Here ‘\(q = 1 - p\)’ where ‘\(p\)’ is the probability of generating a SPITRI message.

According to the analysis by Lee et al. [20] the probability of reconstructing the full attack path using ITrace forward or backward link enabled and using both the links enabled in an ICMP based traceback is given by Eqs. (8) and (9) respectively.

\[
P_1 = (1 - (1 - p)^{\frac{h}{2}})^n
\]

\[
P_2 = (1 - (1 - p)^{\frac{h}{3}})^n
\]

\(P_1\&P_2\) denotes the probability of reconstructing the full path on receipt of ‘\(n\)’ packets from a path of ‘\(h\)’ hops using ITrace (Forward or Backward Link) and ITrace (Both Links) respectively. In ITrace scheme, path reconstruction probability is dependent on the path length whereas in SPITRI method, ‘1’ SPITRI message from the border router is sufficient to trace till the attacker. According to Cheswick et al. [24] and Huffaker et al. [25] majority of the network path is less than 32 hops. So the probability of reconstruction of 16 hops and 32 hops attack path on receipt of ‘\(n\)’ packets using ITrace (Forward or Backward Link), ITrace (Both Links) and SPITRI were analysed. Fig. 8 depicts the path reconstruction probability if SPITRI and ITrace are sent at a low probability of 1/20000. Fig. 9 depicts the path reconstruction probability if SPITRI and ITrace are sent at a high probability of 1/1000. Figs. 8 and 9 show that SPITRI requires less number of packets compared to ITrace irrespective of the probability at which the traceback message was generated.

Fig. 8. Number of packets required (\(p = 1/20000\)).
5.1.3 Path reconstruction time

Path reconstruction time is one of the prime metrics used in evaluating the traceback processing overhead. It has a direct impact on the response time taken to defend an attack. In ITrace the path to reach the attacker is reconstructed only after collecting required number of ICMP packets from the routers in the attack path. The path reconstruction time of ITrace is controlled by two parameters. One is the time taken to collect the required audit trails \((T_c)\). This includes the waiting time to receive all the traceback packets in the attack path. The other is the time taken to organize the collected information and reconstruct the attack tree \((T_R)\). The collected ICMP packets are stored and retrieved to form the attack tree during the reconstruction phase. This reconstruction involves heavy storage, which in turn increases the fetch time during large scale attacks. The path reconstruction time in ITrace \((T_I)\) is given by Eq. (10)

\[
T_I = T_c + T_R. \tag{10}
\]

In the proposed method, ‘1’ SPITRI message is sufficient to traceback an attacker, so the waiting time to collect the ICMP packets is eliminated and the path reconstruction time in SPITRI \((T_S)\) is given by Eq. (11)

\[
T_S = T_R. \tag{11}
\]

Moreover the traceback algorithm of SPITRI involves only an iterative operation \(h\) (number of hop) times, whereas ITrace requires matching different packets to identify the appropriate edges to be linked. Therefore the average reconstruction time required for SPITRI is \(O(n)\) where \(n\) is the number of hops traversed by that corresponding SPITRI message whose value would be less than 32 in majority of the cases [24, 25].

It is evident from the mathematical evaluation that the time taken by SPITRI to reconstruct the path would be significantly lesser compared to the time taken by ITrace because of the elimination of \(T_c\).

5.1.4 Bandwidth overhead

In this fast moving world, bandwidth is the vital resource which is always scarce.
Subsequently one cannot afford to traceback an attacker at the cost of high bandwidth overhead. The amount of packets generated to trace back an attacker implies the additional bandwidth utilized by the methodology.

ITrace requires a traceback message from every hop of the attack path. Whereas SPITRI requires only one message to traceback an attacker irrespective of the attack path length. The bandwidth overhead is illustrated in Fig. 10 for different attack path length varying the probability in SPITRI and using the probability of 1/20000 in ITrace. It is evident from Fig. 10 that even though SPITRI is sent with the probability of 1/5000 there is only 0.02% of net increase in traffic irrespective of path length which is equivalent to the additional traffic produced by ITrace for a very low probability of 1/20000 for a shorter path length of 4 hops. In ITrace the bandwidth overhead is increasing as the path length increases but SPITRI requires a uniform bandwidth depending on the number of attack sources and not on path length. Even if ITrace is generated with a low probability of 1/20000, it exceeds the bandwidth overhead caused by SPITRI sent at a high probability of 1/1000 when the path is longer than 20 hops. The bandwidth overhead of ITrace is path length times higher than SPITRI because ITrace generates a traceback message at every hop. Let $B_I$ and $B_S$ denote the bandwidth overhead per path by ITrace and bandwidth overhead per path by SPITRI respectively. The relationship between the bandwidth overhead caused by ITrace and SPITRI is given by Eq. (12).

$$B_I = c \times h \times B_S$$

(12)

where $h$ denotes the number of hops in the attack path and $c = p/q$, $p$ refer to the probability at which ITrace message is sent and $q$ refers to the probability at which SPITRI message is sent.

5.1.5 Accuracy

Accuracy is the most important metric to evaluate an IP traceback scheme. If it produces false positives the legitimate users would be penalized unnecessarily. When a router is falsely identified as an attacker border router, it is called as a false positive. When an attacker border router is left unidentified, then it is called as a false negative.

The proposed system was simulated over the network generated by BRITE topology generator shown in Fig. 7 and found to produce zero false negative under the condition that at least one SPIRTI message is generated from the attacker border router. It is shown in Fig. 11.
In ITrace, the attack path can be reconstructed only after collecting all the ITrace messages associated to that path. When the number of attackers increases the number of expected number of ITrace messages also increases. When a router fails to send an ITrace message it would lead to incomplete path. Incomplete paths would lead to false negatives. ITrace-CP produce zero false negative under the condition at least one ITrace-CP message is sent from the attacker’s border router. It can identify only the border router and the attack path can be identified only if the attack packet and the ITrace-CP message traverse in same path and the IP packets stored at the routers are not refreshed before ITrace-CP message arrive that router.

In SPITRI, once the corresponding ICMP packet is selected, the traceback algorithm would give out the path to reach the attacker from the victim in terms of interface ID. Since it does not rely on multiple packets or logged data, there is no chance of incomplete path or collision. It produces zero false positive and zero false negative unless the Path information element is corrupted by an intruder.

In order to test the accuracy of SPITRI in the actual Internet, it was also simulated with the CAIDA Ark dataset [9] which gives the path from a monitor station of CAIDA to 190984 destination nodes. It consists of 327262 nodes and 2794611 links. It was assumed that attackers were connected to each of the end nodes. The monitor node was considered as the victim and SPITRI was simulated in the topology created from the CAIDA Ark dataset. The number of attacking nodes was varied from 1000 to 190000. In every simulation SPITRI was able to detect the attacking node accurately with zero false positive and false negative provided a SPITRI message is generated from each of its end router.

5.2 Comparison to Router Interface based Method

SPITRI is also evaluated by comparing it with RIHT which is the state-of-the-art router interface based method RIHT [7] using the following performance metrics.

1. Computation Time during Construction of Audit Trail
2. Router Involvement in Traceback
3. Storage Overhead

5.2.1 Computation time during construction of audit trail

Every router in the path is used for construction of audit trails in both SPITRI and
RIHT. However, as discussed earlier, SPITRI requires computation in every router only in SPITRI message, while RIHT requires computation in every router in every packet it forwards. The time taken by both RIHT and SPITRI to create audit trail information at a router was analyzed by simulating both RIHT and SPITRI using NS2 simulator.

The time taken by a router to process the packets from different paths was analyzed. Since the maximum log size of RIHT is given as $2^{15}$, for simulating RIHT $2^{15}$ 32 bit random marking field values were selected. It is to be noted that each of this value corresponds to a path. These $2^{15}$ values were given $k$ times as an input to the construction process, which means $k \times 2^{15}$ packets are arriving at that particular router from $2^{15}$ paths. Then the overall computation time was recorded. This simulation was repeated 50 times and the average computation time was noted for different values of ‘$k$’. SPITRI was simulated in the same way by selecting $2^{15}$ random Path Information values. To analyze the worst case, these $2^{15}$ values were also given the same $k$ times as an input to the construction process, which means $k \times 2^{15}$ ICMP traceback packets are coming from $2^{15}$ paths. However this was only done to analyze the computation overhead in a worst case scenario. It can happen only if the border router generates an ICMP packet with the probability of ‘1’. Then, the overall computation time was recorded. This simulation was repeated 50 times and the average computation time of 50 simulations was computed. It is evident from Fig. 12 that the computation time of SPITRI is shorter than RIHT because it does not need logging and hashing as in RIHT.

![Fig. 12. Computation time (CT).](image)

### 5.2.2 Router involvement in traceback

The computation overhead incurred on the router has to be minimal in order to avoid the degradation of router performance. In most of the traceback schemes routers are expected to assist in creating audit trails. Hence at least the reconstruction of path or identification of an attacker should not once again burden the routers with additional computation. Fig. 13 illustrates router involvement during traceback process in RIHT and SPITRI. The traceback process in RIHT is done by traversing back in the attack path by involving every router in the path in traceback process whereas SPITRI is capable of giving the path to reach the attacker from the victim itself.

### 5.2.3 Storage overhead

In-spite of adding additional task for construction of an audit trail if the router is burdened further with heavy storage then it would apparently impact the performance of
the router. SPITRI requires storing the interface table like any other router interface based approaches. Even ITrace method requires the interface name to be marked in the packet which would also occupy the same negligible storage at the routers. RIHT requires an additional hash table along with the interface table. SPITRI does not require any additional storage other than the interface table. The interface table has to store the interface address and the corresponding locally unique interface identifier. CAIDA ITDK2012 dataset [26] was analyzed to find that more than 99.99% of routers have degrees fewer than 256. So assuming that the degree of the router is 256, the interface table occupies 1280 bytes to store 256 IP addresses and its corresponding unique identifier. According to the theorem in RIHT [7] along with interface table it requires additional 80N bits where N is the number of paths to be logged. Fig. 14 illustrates that in RIHT storage increases with the increase in logged paths.

In view of the fact that ITrace-CP [20] can also traceback with the single ICMP packet a qualitative comparison of ITrace, ITrace-CP, RIHT and SPITRI is given in Table 3.

![Storage requirement.](image)

**Table 3. Qualitative comparison with related traceback schemes.**

<table>
<thead>
<tr>
<th>Traceback Scheme</th>
<th>Storage at the Routers</th>
<th>No. of Packets Required to Collect Audit Trail</th>
<th>False Positive</th>
<th>Traceback Granularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITrace</td>
<td>Not Required</td>
<td>More than 1</td>
<td>False positive when handling multiple attackers</td>
<td>Attacking host/LAN</td>
</tr>
<tr>
<td>ITrace-CP</td>
<td>16MB/100 ms Required</td>
<td>1 or more than 1 (Depends on the path taken by ICMP traceback message)</td>
<td>False positive when handling multiple attackers</td>
<td>Attacking host/LAN</td>
</tr>
</tbody>
</table>
Table 3. (Cont’d) Qualitative comparison with related traceback schemes.

<table>
<thead>
<tr>
<th></th>
<th>RIHT</th>
<th>SPITRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required 320 KB</td>
<td>Not Required</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>False Positive when an interface is added to a router after packet marking</th>
<th>Nearest Router</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIHT</td>
<td>1</td>
<td>False Positive when an intruder modifies the Path Information</td>
</tr>
<tr>
<td>SPITRI</td>
<td>0</td>
<td>No False Positive unless an intruder modifies the Path Information</td>
</tr>
</tbody>
</table>

6. CONCLUSION

A novel Single Packet ICMP Traceback scheme using Router Interface is proposed to trace flooding DDoS attack. The proposed SPITRI is proved to be efficient than ITrace in terms of bandwidth overhead, reconstruction time and accuracy. Bandwidth overhead incurred by SPITRI is nearly path length times lesser than the one in ITrace. It has zero false positive and false negative rates in path reconstruction. Compared to the state-of-the-art router interface based technique it is shown to be efficient in terms of computation cost and storage cost. SPITRI is backward compatible. It occupies only 1.25KB for CAIDA ITDK dataset which is 256 times lesser than the state-of-the-art router interface based technique and it does not pollute the network with additional packets for the traceback process. It can also trace back large scale attacks faster compared to the existing schemes and can identify any number of attackers with their corresponding ICMP packet, from the victim itself thereby eliminating the processing cost at the routers during traceback.

Future work could be carried out in the following directions.

1. All the routers in the network are required to implement SPITRI to traceback the attack path which could be extended by accommodating the routers that do not conform to SPITRI by means of Autonomous System (AS).
2. Since the ICMP packet is sent with a probability it could only trace flooding attack. It could be upgraded to send ICMP packet per flow and enable single packet traceback.
3. To improve the security, elements of ITrace message such as HMAC Authentication Data and Key Disclosure List could be added.

REFERENCES


**M. Vijayalakshmi** is currently working as an Assistant Professor in Department of Computer Science and Engineering, Thiagarajar College of Engineering, India. She is currently pursuing the Ph.D. in Anna University, India. Her research interests include network security and routing algorithms.

**S. Mercy Shalinie** is currently an Associate Professor and Head of the Department of Computer Science and Engineering, Thiagarajar College of Engineering, Madurai, India. Her research interests include AI, machine learning and information Security. She has the distinction of publishing over 75 research papers in refereed International and National Journals and Conferences. Her sustained research interest has made her complete sponsored R&D projects from DRDO, AICTE, DeitY, DST, NTRO, Honeywell and Yahoo, India. Her passion to work in Free/Open Source Software has led to the development of ICT Framework for Thiagarajar College of Engineering which has received National level accolades. She received her Ph.D. (Computer Science and Engineering) in 2000 from Madurai Kamaraj University. She has Post Doctoral Research experience at University of California, Irvine, USA and Monash University, Melbourne, Australia. She is a Member of IEEE, CSI, IE, ACCS and ISTE.