JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 32, 27-45 (2016)

Low Storage and Traceback Overhead IP Traceback System

S. MALLIGA, C. S. KANIMOZHI SELVI AND S. V. KOGILAVANI
Department of Computer Science and Engineering
Kongu Engineering College, Perundurai
Tamil Nadu, 638 052 India
E-mail: mallinishanth72@gmail.com; *{kanimozhi; kogilavani}@kongu.ac.in*

Using IP spoofing, a person masquerades as another by falsifying source IP address and gains an illegitimate access. Denial of Service (DoS) is an attack that is launched to bring down a network by flooding it with useless traffic. This attack can be easily exploited by IP spoofing. To prevent DoS, it is necessary to determine the source of the attacks. IP traceback is a mechanism that attempts to reconstruct the path traversed by a packet to find the real source. Two predominant traceback mechanisms are packet marking and logging. Packet marking records the path information of the intermediate routers in the packet, which can then be used to reconstruct the path. Packet logging logs the packets at the intermediate routers. Hybridizing these two methods gives the benefits of both. This paper refines a hybrid IP traceback method, Modulo and Reverse modulo and proposes a few changes in the way the packets are logged and tracked back. Revised-MORE uses subnet address to create hash values rather than source IP. This reduces the amount of packets to be logged at the routers. Time-To-Live is used for tracing exactly. The simulation results show that the refinements reduce logging overhead, storage requirements and improve traceback accuracy.

Keywords: IP spoofing, DoS, IP traceback, packet marking, packet logging, logging overhead, traceback accuracy

1. INTRODUCTION

The prime motive of a DoS attacker is to degrade and damage the resources of a server, so that legitimate users are denied from the services they requested. The attacker will exploit the inherent weaknesses of IP protocol to launch DoS attacks. One among the weaknesses is the non authenticity of source address which leads to spoofing. IP spoofing is forging of one's IP address. DoS attackers spoof source IP address in order to hide their identity which complicates the process of identifying the real source much difficult. In their book, Kevin and Chris [1] have described that the DoS attacks have the following properties: destructive, resource consumption and bandwidth consumption. They have also discussed the devastating effects of DoS attacks. The distributed nature of DoS attacks (DDoS) employs and instructs a large number of weak hosts on a wide area network to flood a specific target. Victims of a DDoS attack consist of both the end targeted system and all systems maliciously used and controlled by the hacker in the distributed attack.

Gong and Sarac, 2008 [2] has classified the DoS attacks into two categories namely: flooding attacks and software exploits. Flooding attacks flood a victim by huge amount of packets whereas software exploits use the vulnerabilities of the TCP/IP protocol suite. Tracing of packets helps identifying the origin of both flooding attacks and software ex-

Received September 22, 2014; revised December 1, 2014; accepted January 2, 2015.
Communicated by Hung-Min Sun.

ploits. Even a single, well targeted attack packet can disable routers and operating system [3].

The impact of these attacks has encouraged many researchers to address the issues and led to the development of many solutions. One such solution is IP traceback. IP traceback is a mechanism that identifies the true origin of a packet. Such method would also be useful for identifying flood-based attacks. IP traceback is achieved either by packet marking and/or packet logging [2].

Packet marking is a technique based on the idea that routers in the intermediate network mark, either probabilistically (PPM) or deterministically (DPM), the packets that pass through them. These marks are used to reconstruct the path traversed by the packets.

The main idea of PPM is to mark the packets probabilistically as they traverse through the routers. Hence, a packet can carry only partial path information. This is done with the belief that after having received ample number of packets, the path can be reconstructed using the marking information present in the packets. Several PPM techniques have been advocated in literature [4-7].

In DPM, a router would mark all the packets that pass through it. Belenky and Ansari [8] outlined a DPM method and put a single mark in the packet at the network ingress point. The idea is to write either upper or lower half of the IP address of the ingress edge into the packet with a random probability and a reserved bit indicates which portion of the address is placed in the ID field of the packet. This approach claims that it is able to find the attacking origin with only 7 packets. Choi and Dai [9] suggested a new DPM method that uses Huffman codes. To mark, the 16-bit IP ID and 16-bit flag and fragment offset fields in the packets are divided into 1-bit saved flag and 31-bit link sequence. The links of a router are represented by Huffman codes based on the distribution of traffic on the links of the router. When a router finds no space for marking, it logs the packets. The victim uses the markings in a packet to find the path of the attack. This method does not address how the fragmented packets would be handled as only less than 0.25% of the packets are fragmented in the Internet traffic [10].

PPM approaches may not be useful for single packet attacks like the Teardrop [11]. To traceback a single packet, logging has been proposed. In logging, every router is responsible for maintaining the information regarding the packets that traverse through it. Sager *et al.* [12] proposed a scheme to log the packets and use data mining techniques to extract the packets later during path reconstruction. This method requires a huge memory space and even if the memory required is afforded, the enormous input/output operations slow down the intermediate routers.

In order to reduce the storage overhead, Snoren *et al.* [13] suggested a logging technique, Source Path Isolation Engine (SPIE) that uses a space-efficient data structure called Bloom filter. The important issue in this method is the time span of keeping the logged information to enable tracing before expiry.

To achieve the goodness of marking and logging approaches, a few hybrid techniques have been studied. Al-Duwari and Govindarasu [14] offered two novel hybrid methods, Probabilistic Pipelined Packet Marking (PPPM) and Distributed Link List Traceback (DLLT), which use both marking and logging. In both DLLT and PPPM, the number of packets for traceback increases when the path length increases.

In this paper, we refine the proposal suggested by Malliga and Tamilarasi [15], which uses both packet marking and logging. In their proposal, they used MOdulo and

REverse modulo (MORE) technique for traceback which can also handle fragmented and transformed packets. This method is capable of tracking a single packet IP traceback. The interlogging distance between routers is high in MORE when compared to other hybrid methods. But, it is found that MORE consumes more space while the packets are logged at the intermediate routers. In MORE, the routers log the packets using hashing technique. For hashing, the routers use the contents of the packets. When the logged packets are tracked, routers have to use the contents of the packet being tracked back. The contents of the packet being tracked should be same as the contents of the packet when it was logged. If they do not match, the retrieval logged information would not be correct. This leads to incorrect identification of source of the packet. This is a flaw found in MORE. Also, every packet is logged when its ID field is insufficient, which increases storage requirements enormously at the routers.

The objective of this article is to revise the earlier proposal (MORE); Revised-MORE, such that it overcomes the problems mentioned earlier. We propose a hashing technique that helps to find the logged packets correctly and also the egress or outbound router through which packets are injected into the Internet. Through experimental results, we present the overhead of packet logging and traceback process and the accuracy of the proposed traceback approach. We also compare the performance of the proposed approach with recent hybrid traceback approaches. The results show that Revised-MORE logs the packets only at very few routers and also incurs less overhead on traceback when competing with other hybrid approaches. We also demonstrate that the storage requirements at the logging router are also less in Revised-MORE. We carry out the experiments through a topology provided by CAIDA.

We organize the rest of the paper as follows: Section 2 presents the motivations and design goals for an IP traceback system. Section 3 summarizes the recent approaches which are based on marking and logging. The proposed system, Revised-MORE, is explored in Section 4. The marking and traceback algorithms of the proposed system are also addressed in this section. The details of simulation experiments and the performance evaluation of the proposed system against different metrics are presented in Section 5. Finally, in section 6, we provide concluding remarks.

2. MOTIVATIONS AND DESIGN GOALS

2.1 Motivations and Terminology

IP Spoofing is a technique used to gain unauthorized access to computers, whereby the intruder sends messages to a computer with an IP address indicating that the message is coming from a trusted host. IP spoofing makes it hard to find the true origin of a packet thus leading the DoS. The DoS attacks flood a network with the intentions of denying the legitimate users from accessing the resources or services on specific system. DDoS can cause more damages on the victim by employing a group of hosts, called zombies. To disguise the true locations, the attackers tend to spoof the packets, thus complicating the detection process. Detecting the source or at least the outbound router is essential to prevent these attacks.

IP traceback is a technique that helps to detect the origin of a packet. That is, the path traversed by the packet needs to be detected. Let R_1, R_2, …, R_n be the ordered list of

routers traversed by the packets from source to destination. The process of reconstructing the path of the traffic is known as traceback process. IP traceback process reconstructs the path consists of R_n, R_{n-1}, …, R_1.

To understand further, we use the following terms: an attack packet is the one whose origin is to be traced. A victim is the destination of attack traffic. Edge or outbound router is a system that attaches a local area network with the Internet. Let d be the number of inbound interfaces of a router assigned with IDs 0, 1, …, $d - 1$. d is also referred to as degree of the router. *Intf* is the interface through which the packets enter into a router.

2.2 Design Goals

An IP traceback method is expected to have the following features:

a. Providing the information about the path traversed to traceback
b. Ability to perform single packet IP traceback
c. Support for backward compatibility: As the packets may undergo fragmentation and valid transformations when they move towards the destination, a traceback system should be able to run under such cases.
d. Ability to locate the real source of attacking traffic or at least the outbound router.
e. Low packet logging and storage overhead
f. Low traceback process overhead
g. High traceback accuracy (*i.e.*) the original path traversed by the packet being traced with less number of false positives. False positive means tracing back the routers which were not involved in the marking at all thus leading wrong path.

A traceback method possessing these design goals would definitely address the security threats imposed by DoS/DDoS attacks.

3. MARKING AND LOGGING BASED HYBRID TRACEBACK APPROACHES

In this section, we outline the features of recent hybrid IP traceback methods.

3.1 Source Path Isolation Engine (SPIE)

SPIE [13] has the ability to identify the source of a specific IP packet given it to be traced, its destination and time of the receipt of the packet. To do so, the routers compute and store the digest of each of the packets in a time-stamped digest table. The key contribution of SPIE is to reduce the memory requirement at intermediate routers by employing space-efficient data structure called Bloom filter (Bloom, 1970). On saturation, the digest tables are paged out and archived for some time period for post mortem analysis.

3.2 Distributed Link List Traceback (DLLT)

DLLT [14] is based on a store, mark and forward approach. It employs both packet

marking and logging. When a router receives a packet, it would mark the packet with some fixed probability. If the packet has been already marked, the router would store the marking information in the packet before remarking. DLLT maintains a digest array and a marking information table at every router. The fraction of the traffic is logged at each router and needs significant amount of memory. DLLT also requires long term storage at each router.

3.3 Probabilistic Pipelined Packet Marking (PPPM)

In PPPM [14], the marking information that belongs to certain packet is transferred by propagating it from one router to another using subsequent packets traversing to the same destination. A PPPM enabled router requires 57 bits for transferring marking information, which may be practically infeasible.

3.4 Logging and Storage Based Bybrid IP Traceback

Gong and Sarac [2] has proposed a hybrid traceback method which is based on both packet marking and logging. It maintains single traceback ability of SPIE and, at the same time, it reduces the storage overhead and access time requirements for storing the digests at the routers. The key contribution of this approach is to record the path information partially at routers and partially in packets. Every router marks the packets whereas every alternative router logs the packets. Thus, compared to SPIE, this approach reduces the storage overhead of packet digest to one half and access time requirement for logging the digests by a factor of 2*d*.

3.5 Precise and Practical IP Traceback (PPIT)

PPIT [19] is a hybrid IP traceback approach. In PPIT, the intermediate routers do marking or logging alternatively in a certain manner (*i.e.*) routers, at every three hops, compute packet digests and record the digests of the packets. Similar to SPIE, PPIT stores packet digests in digest tables that are implemented with Bloom Filter. Since every three hop routers store the packet digests, the logging overhead is high.

3.6 RIHT

RIHT [20] is also a hybrid IP traceback method, which proposes a traceback method that marks interface number of router and performs packet logging with a hash table (RIHT) to deal with the logging and marking. RIHT marks interface numbers of routers on packets so as to trace the path of packets. Since the marking field on each packet is limited, this packet marking method needs to log the marking field into a hash table and store the table index on the packet. This marking/logging process is repeated until the packet reaches its destination. After that, the process is reversed to trace back to the origin of attack packets. RIHT uses IP ID and offset fields to mark packets, which is meant for fragmentation. So it inevitably suffers from fragment and drop issues for its packet reassembly.

3.7 Hybrid IP Traceback (HIT)

HIT [18] prevents packet fragmentation and insufficient storage for log tables by using the 16-bit ID field of an IP header. When a router receives a packet, it computes new marking information and stores it in the packet. If the IP ID field is not sufficient to hold the marking, HIT computes the hash value on marking in the packet and inserts the marking along with the incoming interface in a log table. Multitables are used to store the packets' digest in case of overflow of log tables. The filled log tables are stored with timestamp. These log tables are used while the reverse path is constructed. Since the packet digests are taken based on the source IP, there may be multiple entries for the packets from same source, if they have different marking. As log tables are to be consulted based on the timestamp, the gap between the times at which the packet's digest is recorded and the traceback request for the packet is initialized to be within a reasonable limit. Otherwise, unnecessary search of log tables may incur.

3.8 MOdulo and REverse Modulo (MORE)

MORE [15] uses two algorithms to perform traceback. The first algorithm is to mark the packet and log it, if needed. Every edge router maintains a MacToID table which has a mapping of hardware address of the hosts connected to the network to a unique ID from 0 onwards. When packets leave the network, the edge router marks the packets with the ID corresponding to the MAC address of the originating host. Every core router maintains an InterfaceToID table which has the mapping of the MAC address of the inbound interfaces of the router to a unique ID starting from 0 to $d-1$, where d is the number of interfaces. In order to log the packets, a router uses a table for each of its interfaces called Log Table associated with ID corresponding to the inbound interface. MORE uses modulo operation to mark the packets. It has the ability to mark upto 15 bits in the marking field. When the marking information exceeds 15 bits, the logging is performed. The second algorithm is to traceback the packet to the source. This algorithm uses the marking details in the packet to start traceback. Every router uses reverse modulo to find the incoming interface of the attack packet. Then, the router finds the upstream router connected to that interface. This process is repeated till the real source is identified.

4. REVISED-MORE

MORE is a hybrid single packet IP traceback method. In this method, each traceback enabled router, does packet marking and/or packet logging. The marking involves recording router identification information into the marking field of the packet. The logging operation is to compute and record the packet digest. While forwarding a packet, routers decide to mark or log the packet depending on whether there is free space available in the marking field of the packet. If so, routers mark the packet. If no space is available, routers log the packet and reinitialize the marking field. The original MORE involves two components. The first part involves packet marking and logging for enabling IP traceback. The second part uses the marking and logging information to traceback the packets to their origin.

4.1 Marking and Logging Algorithm

We revise and extend the marking algorithm proposed by Malliga and Tamilarasi [15], Revised-MORE, to increase the traceback accuracy and decrease storage overhead.

Every router maintains an InterfaceToID Table (IIDT) which shows the mapping of the hardware address of the inbound interfaces (*intf*) of the router to a unique ID starting from 0 to $d-1$. To mark a packet, the 16-bit IP ID field is used. As this field is needed only for fragmented packets and there are only 0.25% of packets are fragmented [17], IP ID field is used for marking. When IP ID field is used for marking, the flag and offset fields used during fragmentation is of no use. But, using the offset bits would cause confusion at the destination as non zero values in the fragmentation offset field would be misinterpreted as IP fragments. Also, 32-bit marking has been employed at the cost of backward compatibility. To address this issue, the only 16-bit IP ID field is used. As, all these fields are needed for fragmented packets, these packets are logged at each router (*i.e.*) fragmented packets are not marked.

It is also understood that packets may be modified during the forwarding process. According to Claffy and McCreary [16], 3% of the packets undergo transformation. The transformation includes the change in Time To Live (TTL) field, checksum re-computation *etc.* These transformed packets should also be tracked back. The Revised-MORE supports such transformations also. The marking used by the Revised-MORE for non-fragmented packets is shown below in Fig. 1.

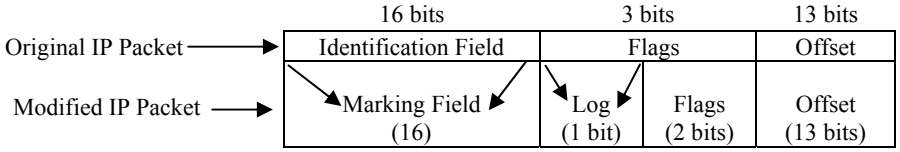

Fig. 1. Marking details.

Every router maintains an InterfaceToID Table (IIDT) which shows the mapping of the hardware address of the inbound interfaces of the router to a unique ID starting from 0 to $d-1$. The routers use modulo technique for marking the packets. The format of an IP packet that has marking is shown in Fig. 2.

Version (4)	Header Length (4)	TOS (8)	Total Length (16)		
Marking Field (16)			Log Field (1)	Flag (2)	Offset (13)
TTL (8)		Protocol (8)	Checksum (16)		
Source IP (32)					
Destination IP (32)					

Fig. 2. Format of IP packet with marking and logging fields.

The marking procedure is explained in Fig. 3.

When a router finds that 16 bits are not sufficient to embed its marking, it has to log. MORE maintains a table for each inbound interface of the routers. For each packet coming through an interface that needs to be logged, it is required to store the digest of the packet, marking information in the packet and log field value. This is done irrespective of whether the packets from the same source have been already logged or not. Thus, multiple entries exist in a table for the packets coming from a source. It is enough to store the packets coming from the same source once. This would reduce the storage requirements at the logging router to a greater extent.

The proposed Revised-MORE makes a log for each subnet address rather than for each source IP. In Revised-MORE, a router logs the marking information present in the packet, TTL and the logging field values. For logging, a router maintains a table for each interface as in MORE. But Revised-MORE uses a hash table for each inbound interface. A hash table uses a hash function to compute an index into an array of buckets or slots, where the marking information is stored. Here, we use open hashing, which defines each slot in the hash table to be the head of a linked list. All records that hash to a particular slot are placed on that slot's linked list. The records within a slot's list can be ordered in several ways: by insertion order, by key value order or by frequency-of-access order. Ordering the list by key value provides an advantage in the case of a search.

In Revised-MORE, in order to log, a router extracts the subnet address from the source IP and applies hash function on the subnet address. The hashed value is used to find an entry in the hash table associated with the incoming interface. If the entry is null, the router stores marking information, TTL and log field values at the entry. If the entry is not null, the marking information and TTL already present in the nodes of the linked list attached to that entry is compared with the marking information and TTL present in the packet to be logged. If they match, then the packet is not logged, thus reducing the storage requirement. If they do not match, then a node is added to linked list attached to that entry. The node has the fields for storing the marking information in the packet, the TTL value in the packet and the value of the logging field. Time to live (TTL) or hop limit is a mechanism that limits the lifetime of packet in a computer. The default value of the field depends on the operating system. The value of this field is decremented by every router on the path towards the destination. The logging of TTL field helps in the traceback process. The marking algorithm is detailed below in Fig. 3.

```
1.  For each packet p entering into the egress router
2.  {
3.      For each core router, R_c
4.      {
5.          Find the ID assigned to the intf at R_c using IITD.
6.          If p is not a fragmented packet
7.          Calculate the new marking information (new_mark) as
8.          new_mark = p.marking field * number of interfaces of R_c + ID
9.          If p.marking field is sufficient to hold m and p is not transformed at R_c
10.                 {
11.                     p.marking field= new_mark
```

```
12.                 }
13.         If p.marking field is not sufficient to hold m and p is not transformed at Rc
14.            {
15.            Extract the subnet address from the source IP and compute the hash value of it
16.            Check whether the hash table of intf has entry for p.marking field or not.
17.                 If entry is null
18.                 {
19.                    log (p.marking field, p.ttl, p.log field)
20.                 }
21.                    If entry is not null
22.                 {
23.            Check the list attached to the entry for the presence of value of p.marking
                     field and p.ttl
24.                    If present
25.                 { // entry exists already
26.                    Do not log
27.                 }
28.                    If not present
29.                 {
30.                  Add a node to the list containing (p.marking field, p.ttl,p.log field
31.                  Clear and set the p.marking field=ID assigned to the intf of p at Rc
32.                 }
33.                  }
34.                                    Set the p.log field=1
35.                              }
36.                  }
37.      }
```

Fig. 3. Marking algorithm.

The fragmented and transformed packets are handled as in SPIE and HIT. Even though it incurs additional logging overhead on routers, it is unavoidable to traceback such packets to curtail packet fragmentation attacks.

4.2 Traceback Algorithm

When a node (*i.e.*) a victim, understands that it is under attack, it invokes path reconstruction procedure to detect the source of the attack. It is assumed that the victim has some attack detection systems. The marking information in the packet helps to initialize traceback process. The traceback process starts at the last hop router of the packet, which uses reverse modulo to find the incoming interface of the packet in question. Through this interface, the traceback request is forwarded to the upstream router, which in turn uses reverse modulo to determine the inbound interface of the traceback requested packet at this router. The reverse modulo technique is successively applied at each upstream router to locate the egress router of the traceback requested packet. Once reached, the egress router may take further steps to prevent the packets going out of it.

To retrieve the logged information at a router, if needed, the hash table is used. The subnet address is used to find the entry in the hash table. Since there may be more than

one node in the linked list associated with the entry, it is important to find the appropriate entry in the list. TTL is used to find this [19]. Suppose a packet, whose initial TTL value is 64, is forwarded by five routers (R_1, R_2, R_3, R_4 and R_5). Then, the value of the TTL is decremented at each router that is 63 at R_1, 62 at R_2 and so on and is 59 when it reaches the destination. These TTL values are logged along with the marking information in the hash tables. In traceback process, the TTL value is incremented by one in each hop towards the source. While tracing this packet from destination to R_5, the TTL value is 59 and 60 at R_4 and so on. Assume that this packet has been logged at R_4. To find the hash table in which the packet has been logged, we reverse modulo, which determines the inbound interface of the packet. The hash table of this interface is the one that has logged the packet. Next, to locate the entry in the table, we apply hash on the subnet address of the source IP. Once the entry is located, there may be list of nodes attached to it. Next, we compare the TTL (60) in the traceback requested packet with the TTL recorded in the node while logging. Once match is found, the marking information in the matched node was the marking that was logged before reinitializing the marking. The detail of traceback algorithm is illustrated in Fig. 4.

```
1. For each packet p at router Rc.
2.  {
3.    Let d be the number of interfaces of Rc
4.    If p.marking field < d and p.log field=1
5.    {
6.     //This implies that p has been logged at Rc during its travel towards destination
7.     Set intf = p.marking field
8.     Compute the hash value for the subnet address to find the entry in the hash table of intf
9.     Find the node in the list that has same TTL as that of traceback requested packet
10.  Retrieve the marking information and log fields from the node and set it to
     p.marking field and p.log field respectively
11.       }
12.       Else If p.marking field < d and p.log field=0
13.       {
14.       Find the inbound interface as intf=mod(p.marking field, d)
15.          }
16.          Else If p.marking field ≥ d and p.log field=0
17.          {
18.             //This implies that Rc has forwarded p
19.               Find the intf that let p in to Rc as
20.               intf=mod(p.marking field, d)
21.               Set p.marking field = p.marking field / d
22.           }
23.     }
```

Fig. 4. Traceback algorithm.

4.3 Analysis of Design Goals

Section 2.2 has highlighted the desired features of a traceback system. In this sec-

tion, we analyze how Revised-MORE achieves the first four design goals. The remaining goals are addressed with appropriate performance metrics in Section 5. The first goal is to have the information about the route traversed by a packet. Since every router along the path marks and/or logs the packet, this information is used to traceback the packet when needed. As every packet is marked by every router, Revised-MORE can trace even a single packet. Revised-MORE also has considered the issues of packet fragmentation and transformation to provide backward compatibility. This is addressed in Section 4.1. Even though, Revised-MORE cannot track back to the originating host, it is capable to find the router to which the originating host is attached to.

5. SIMULATION RESULTS AND PERFORMANCE EVALUATION

Here, we evaluate the performance of Revised-MORE and compare with promising single packet IP traceback approaches presented by [15, 18, 19]. As MORE outperforms SPIE, DLLT and PPPM, we take only those mechanisms which were proved to be better than MORE to substantiate Revised-MORE. First, we present the metrics used for evaluation.

5.1 Performance Metrics

We adopt the metrics used to evaluate the earlier proposal, MORE

i. Router overhead on marking: This measure helps to determine how fast the attack is detected. Lower the overhead, faster will be the detection
ii. Packet logging overhead: This measures the storage requirements at a logging router.
iii. Traceback process overhead: the number of routers queried contributes to this measure.
iv. Traceback process accuracy: This measure determines how accurately the attack path is reconstructed.
v. Convergence time: The number of packets needed to reconstruct the attack graph will be the convergence time. This should be as minimum as one.

To simulate the internet topology, the topology distributed by CAIDA Skitter project [21] has been used as sample data set. The data set consists of paths to a specific host of the topology. The topology has an average hop count of paths of about 14.42 and the average degree is 2.63. One of the routers has degree 434, which is the largest in the data set, while the second largest degree is only 157. Hence, according to CAIDA, the router whose degree is 434 requires the largest storage and Revised-MORE will manage to meet these requirements.

5.2 Router Overhead on Marking

The overhead of routers on marking is determined by the number of bits to be marked. Revised-MORE records 16-bit marking information in the marking field. PPIT marks 14-bit information in the marking field, where as size of the marking information in HIT is 16-bit. The original MORE marks 15-bit information in the marking field. There is no big difference in marking information in all the approaches taken for evaluation.

5.3 Packet Logging Overhead

Packet logging is performed under the following situations:

a. When the marking details exceeds 16 bits
b. When the packet undergoes transformation and/or fragmentation

From MORE, we have found that the packet logging overhead incurred is between 0 and 0.5 for Revised-MORE. For PPIT, the probability for packet logging overhead is from 0.382 to 0.392. It is clear from [18] that the increase in the number of packets to be logged would be more for HIT than that of Revised-MORE. To support this statement, an analysis of frequency of logging at the intermediate routers is presented below:

The number of bits to be marked depends on the number of interfaces and marking information. Let d be the average degree of a router (*i.e.*) the number of interfaces of the router. These interfaces are assigned IDs from 0 to $d - 1$. Let us assume worst and average case for analyzing the increase in marking information. The worst case is that all packets are coming through the interface numbered, $d - 1$, at all the routers. Then, the increase in the size of the marking information for Revised-MORE and HIT is given in Table 1.

Table 1. Size of Marking information.

Intermediate routers	Revised-MORE	HIT
Router 1	$0*d+d-1=d-1$	$0*(d+1)+d = d$
Router 2	$(d-1)d+d-1(d+1)(d-1)$	$d(d+1)+d=d(d+2)$
Router 3	$(d-1)(d+1)d+d-1=(d-1)(d^2+d+1)$	$(d(d+2)*(d+1))+d=d(d^2+3(d+1))$

According to the studies [22, 23], the average length of number of hops is around 16 with less than 32 hops as path length for almost all the paths and average number of interfaces of a router (*i.e.*) neighbors of a router is 3.15. To find the interlogging distance between routers, we here consider a few average degrees of routers and present the frequency of logging.

With d as 4 and all the packets pass through $(d - 1)$th interface, the marking information would have 16 bits up to 8th router and can be embedded in a packet without logging (*i.e.*) the packet requires to be logged at 9th router from the first hop router. This indicates that the interlogging distance is 8. This means that every 9th router needs to log the packets. Since this is at the worst case, the interlogging distance would further increase if the packets come through other interfaces. But, the original MORE requires to log at the every 8th router. HIT requires every 7th router to log the packets. In PPIT, the interlogging distance is 2, that is, it records the digests of the packets at every three hop router.

The packet logging overhead of Revised-MORE has also been validated for a router-level topology from CAIDA [21]. This topology shows that the average degree of a router is 2.63 and average path length is 14.42. With this average degree 2.63 (*i.e.* 3), it is found that the interlogging distance is 10 for Revised-MORE, which means every 11th router needs to log the packet. For the same scenario, HIT requires logging at every 9th router. For MORE, the inter-logging distance is 9 (*i.e.*) every 10th router needs to log in. This clearly indicates that logging overhead incurred by Revised-MORE in considerably

less than MORE, HIT and PPIT.

We have also conducted simulation experiments on the router level topology ITDK0304 by CAIDA [25]. The topology consists of 192244 nodes and 636643 directed links. The average and maximum node degrees of this topology are 6.34 and 1071 respectively. We have tested the performance of Revised-MORE against packet logging and traceback overhead. Packets have been sent between several pairs of nodes and packet logging overhead has been calculated. For doing so, we have assigned maximum (d – 1) and average ((d – 1)/2) ID to the interface through which packets enter into the router.

For the worst case scenario, the percentage of logging at routers is as follows: When 'd – 1' is assigned to the interface, the simulation results show that only 21.8% of the routers require logging of the packets in Revised-MORE. Whereas, PPIT requires logging at 31.6% of the routers for the same ITDK0304 topology. HIT logs at 24% of routers. The precursor of Revised-MORE, that is MORE is required to log at 23.4% of router.

Assume that all the packets are coming through the interfaces numbered (d – 1)/2, then the percentage of logging done by Revised-MORE, MORE, PPIT and HIT is presented below: For Revised-MORE, 20.1% of routers performs logging. Irrespective of worst and average case behavior, PPIT requires 31.6% of routers logging whereas HIT needs 21.8% of routers to log. 21.2% of routers perform logging in case MORE.

To determine the interlogging distance for several pairs of nodes involved in communication for ITDK0304 topology, we conduct many test runs for different path lengths. The results of these simulation experiments for worst and average cases of the packets' incoming interface are depicted in Figs. 5 and 6.

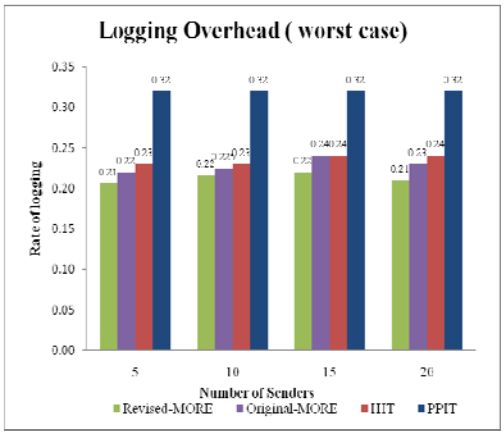

Fig. 5. Logging overhead (Worst case) for ITDK-0304 topology.

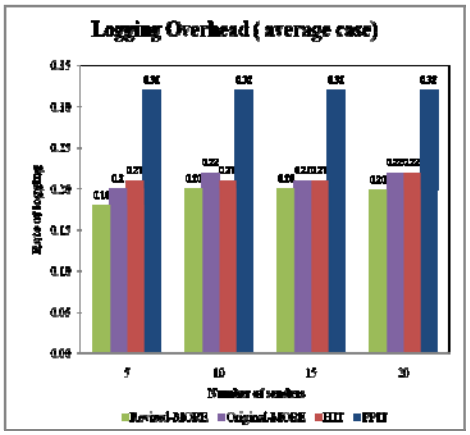

Fig. 6. Logging overhead (Average case) for ITDK0304 topology.

Storage requirements at logging routers

IP Traceback methods require logging of packets if IP fields used for marking overflow. When the degree of a router is more, we will need more bits to record it, which causes larger marks. The larger the mark, the more will be the logging frequency. Hence, more storage is required at downstream routers.

In order to analyze the storage requirements of hybrid single-packet traceback

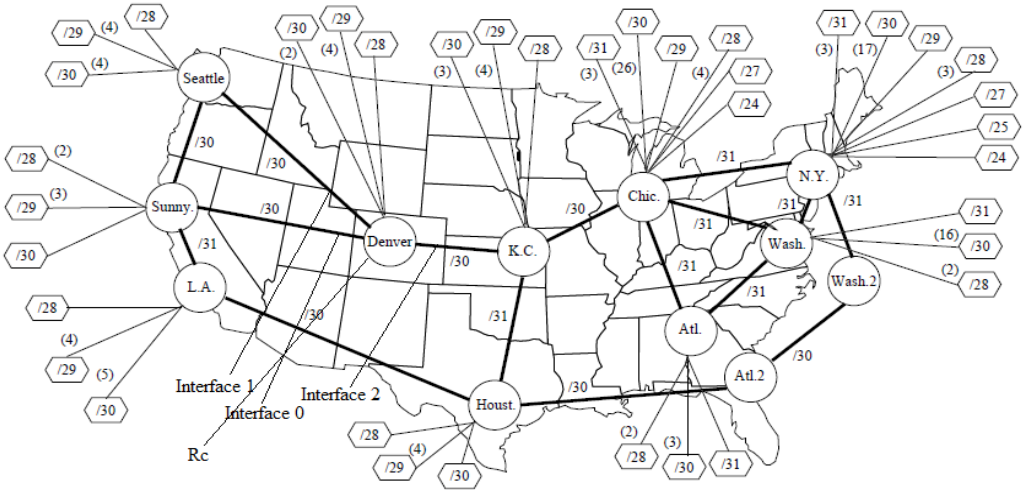

Fig. 7. Partial topology map of Internet2.

methods, we use the topology map of Internet2 back-bone from Abilene–Visible Backbone [24]. There are 547 routers with 793 IP addresses, and 150 subnets. A part of the topology is shown below in Fig. 7.

Consider the core router, named R_c. This router has degree of 3. According to Revised-MORE, this router maintains three tables, each at an interface. Let us assume that the packets need to be logged at this router. As the logging is based on hash value of subnet address, this router requires logging only 6 packets at interface 0.This is because the interface receives packets from 6 subnets connected with routers at Sunny and Seattle. If packets from these subnet carrying different marks enter, then they are logged into the links attached to the entry that corresponds to the hashed subnet address. Thus, the number of entries in the table at each interface is very less.

In HIT, a router logs using source IP addresses. Since a subnet would have a handful number of IP addresses, the number of entries would also be high. Thus, the size of the log table keeps on growing. Once all the entries are filled, a new log table is created. This indicates that storage requirements at the routers would be more than that of Revised-MORE. Similarly, in PPIT, every packet is logged at every three hop router.

An experiment has been conducted to find the storage requirements at a logging router for Revised-MORE, HIT and PPIT for the Internet2 back-bone topology. Assume that a router requires logging for all the four approaches. Revised-MORE demands 3759 bits (150*(16+1+8) *i.e.* 16 bits marking information, 1 bit for logging and 8 bits for TTL) of information for 150 subnets. HIT requires 12688 bits (793*16 bits for marking) to be stored at the router since there are 793 hosts in the topology. In case of PPIT, 11102* number of packets per host (793*14 bits) bits to be logged at the router. For MORE, the number of bits to be logged at a router is 144 bits (15 bits marking information, 1 bit logging and 128 bit packet digest) for each packet. Then, MORE requires 114192 bits (793*144 bits). This shows that the storage requirement for Revised-MORE is comparatively lower than MORE, HIT and PPIT. We have also conducted some simulations to supplement the above results. The parameters for simulation are shown below Table 2.

Table 2. Parameters for simulation.

Parameters	Values
Number of subnets	5-20
Number of hosts/subnets	20-100
Number packets/hosts	500-1000
Amount of data sent	750 MB to 1.5 GB

Table 3. Storage requirements at a logging router.

Number of subnets	Number of hosts/subnet	Number of packets sent/host	Storage requirements (in KB)			
			Revised-MORE	HIT	PPIT	MORE
5	20	500	0.02	0.20	87.50	900
10	50	750	0.03	1.00	656.25	6750
15	75	850	0.05	2.25	1673.44	17212
20	100	1000	0.06	4.00	3500.00	36000

The amount of memory required at a logging router for the above scenario is illustrated in Table 3.

It is clear from Table 3 that the storage requirements at a logging router for Revised-MORE are very low when compared with MORE, HIT and PPIT. The storage requirement of MORE is too high as it stores the packet digest of size 128 bits for each packet.

5.4 Traceback Process Overhead on Routers

IP Traceback is the process of querying upstream routers to reconstruct the attack path. The number of routers queried defines the overhead of traceback process.

Let R_1, R_2, …, R_n be the list of routers traversed by the packets towards destination. To reconstruct the attack path from the destination to the source, the traceback process iteratively queries the routers starting from R_n, …, R_1. The proposed method, Revised-MORE queries a router (R_i) to find the upstream router which has forwarded the packet to be traced. The queried router (R_i) uses the marking information in the traceback requested packet to find the upstream router.

Based on the values in the marking and logging fields, the router may need to consult a hash table. If the logging field is 1, the router needs to consult a hash table. To find the hash table, Revised-MORE applies reverse modulo on the marking field. This implies that the log table corresponds to the interface obtained from the marking field alone needs to be queried to find the upstream router.

In PPIT, when a router receives a query request, the router examines all digest tables at this router in the relevant time interval to make matching. In HIT, every router starts with a log table. Once the table is filled up, a new table is created with a time field. This process gets repeated once the current table is filled. Based on time at which the traceback process is requested, the appropriate table is queried to find the upstream router. If the elapsed time between the logging of a packet and tracing the same is high and meanwhile, logging is done on a new table, then it may lead to querying a wrong table.

It is worth here noting that the packet logging and traceback overhead can drop to 0. This occurs when all the packets are coming through the inbound interfaces with IDs 0. At this best case, no logging is required at intermediate routers and hence no traceback process overhead is incurred.

5.5 Traceback Process Accuracy

Traceback process accuracy is determined based on the number of false nodes grafted on the attack path. A system is said to be robust only when it yields low false positives. Traceback accuracy increases when the false positive decreases. During traceback process, the queried router may return false nodes, thus leading to wrong attack path construction.

If less number of routers is queried, then the false positive rate would be less. As only a very few routers are queried than in HIT and PPIT, the rate of false positives in Revised-MORE is less than other approaches.

The simulation has been performed to find the track accuracy of the proposed system and the results have been compared with MORE, HIT and PPIT. HIT claims that it achieves zero false positive in tracking the origin of the attacks. This is possible if appropriate log tables are consulted based on the timestamp. In PPIT, the routers log the digest of a packet along with TTL. This eliminates incorrect path in the traceback process and introduces no false positive. Similarly, the Revised-MORE also logs the TTL along with marking information which helps to trace accurately. Hence, there is no false positive in Revised-MORE too. But, the time for tracing the attack path will be slightly high in case of HIT and PPIT when compared with the Revised-MORE. This is so because, the appropriate log tables have to be queried to find the upstream router on the attack path.

5.6 Convergence Time

Convergence time of a marking algorithm is the minimum number of packets needed to reconstruct the attack graph. MORE technique requires only one packet to construct an attack graph. HIT and PPIT systems also require one packet.

Table 4 provides a comparison of Revised-MORE with MORE, HIT and PPIT.

Table 4. Comparison of Revised-MORE, MORE, HIT and PPIT.

Metrics		Traceback Methods			
		Revised-MORE	MORE	HIT	PPIT
Technique		Marking and Logging	Marking and Logging	Marking and Logging	Marking and Logging
Convergence time		O(1)	O(1)	O(1)	O(1)
Granularity of traceback		First Hop Router	Source Node	First Hop Router	First Hop Router
Logging overhead	Average to Worst case	20.1% to 21.8%	21.2% to 23.4%	21.8% to 24%	31.6%
	Minimum probability	0	0	0.382	0.32
	Maximum probability	0.5	0.5	0.392	0.32
Marking overhead		16 bits	15 bits	16 bits	14 bits
Storage overhead		Low	Very High	Moderate	High

6. CONCLUSION AND FUTURE WORK

DoS/DDoS is a critical problem that needs urgent attention. These attacks can quickly debilitate a target and lead to heavy loss in terms of revenue and productivity. IP spoofing is a technique that is used by DoS/DDoS attackers to launch these attacks. Spoofing hides the attackers. This article has paid a careful attention to the problems of these attacks and presented a solution that detects the source of these attacks.

Hybrid IP traceback methods use packet marking and logging for reconstructing the path traversed by the packets to find the source of the packets. We have extended the earlier proposal, MORE, to reduce the logging requirements, storage and access time and proposed Revised-MORE. This proposal marks 16 bit information in the marking field and has ability to trace a single IP packet.

Simulation results show that the percentage of logging requirements at intermediate routers is less than other hybrid approaches. As the proposed method logs the packets based on the subnet addresses rather than source IP addresses, the storage requirements at the router is also reduced when compared with HIT and PPIT.

The proposed marking and traceback algorithm has not considered the changes in the subnet IP address. When a node moves from one subnet to another, the IP address of the node will change and thus the marking to be stored in a router will also change. Hence, a new entry is added to the hash table at the interface through which the packet from the node which has moved to another subnet, even though the node has an entry already in the hash table. When traceback is performed, the new marking will be used rather the old marking. But, a problem arises when the traceback is performed after the node has moved to another subnet, but before the new marking is recorded in the intermediate routers. Under this situation, the traceback identifies a wrong source. This issue will be addressed in our future extension. Also, the proposed approach tracks back to the subnet to which the source is attached, not the source of the packet being traced. So, tracking of the packets from the IP addresses belonging to the same subnet will lead the subnet only. Hence, some approaches may be addressed to identify the real source of the packet.

Once the source is identified, source-end defenses [16, 26] can be employed to thwart the DoS/DDoS attacks at the place of origination. This would further strengthen the detection system. We also plan to incorporate a source-end defense in the proposed system as a future research work.

REFERENCES

1. K. Mandia and C. Prosise, *Incident Response: Investigating Computer Crime*, Berkeley, Osborne/McGraw-Hill, 2001, pp. 360-361.
2. C. Gong and K. Sarac, "A more practical approach for single-packet IP traceback using packet logging and marking," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 19, 2008, pp. 1310-1324.
3. Microsoft Corporation, "Stop 0A in tcpip.sys when receiving out of band (OOB) data," http://support.microsoft.com/support/kb/articles/Q143/4/78.asp, 2006.
4. T. Anderson, A. Karlin, S. Savage, and D. Wetherall, "Practical network for IP trace-

back," *IEEE/ACM Transactions on Networking*, Vol. 9, 2001, pp. 226-237.
5. A. Perrig and D. X. Song, "Advanced and Authenticated marking scheme for IP traceback," in *Proceedings of the 20th Annual Conference of IEEE Communications and Computer Societies*, 2001, pp. 878-886.
6. D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to IP traceback," *ACM Transactions on Information and System Security*, Vol. 5, 2002, pp. 119-137.
7. A. Perrig, D. Song, and A. Yaar, "FIT: Fast internet traceback," in *Proceedings of IEEE INFOCOM*, 2005, pp. 13-17.
8. N. Ansari and A. Belenky, "IP traceback with deterministic packet marking," *IEEE Communications Letter*, Vol. 7, 2003, pp. 162-164.
9. K. H. Choi and K. H. Dai, "A marking scheme using huffman codes for IP traceback," in *Proceedings of the 7th International Symposium on Parallel Architectures, Algorithms and Networks*, 2004, pp. 421-428.
10. K. Claffy and S. McCreary, "Trends in wide area IP traffic patterns: A view from Ames Internet exchange," in *Proceedings of ITC Specialist Seminar on IP Traffic Modeling, Measurement and Management*, 2000, pp. 1-25.
11. J. Joshi, *Network Security: Know It All*, Prentice Hall, ISBN 0123744636, 2008.
12. G. Sager, "Security Fun with OCxmon and cflowd," in *Internet2 Working Group Meeting*, http://www.caida.org/funding/ngi/content/security1198/, 1998.
13. A. C. Snoren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent, and W. T. Strayer, "Single-packet IP traceback," *IEEE/ACM Transactions on Networking*, Vol. 10, 2002, pp. 721-734.
14. B. Al-Duwari and M. Govindarasu, "Novel hybrid schemes employing packet marking and logging for IP traceback," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 17, 2006, pp. 403-418.
15. S. Malliga and A. Tamilarasi, "A hybrid scheme using packet marking and logging for IP traceback," *International Journal of Internet Protocol Technology*, Vol. 5, 2010, pp. 81-91.
16. S. Malliga, A. Tamilarasi, and M. Janani, "Filtering spoofed traffic at source end for defending against DoS/DDoS attacks," in *Proceedings of International Conference on Computing, Communication and Networking*, 2008, pp. 1-5.
17. Stocia and H. Zhang, "Providing guaranteed services without peer flow management," in *Proceedings of SIGCOMM*, 1999, pp. 81-94.
18. M. H. Yang, "Hybrid single-packet IP traceback with low storage and high accuracy," *The Scientific World Journal*, Article ID 239280, 2014.
19. D. Yan, Y. Wang, S. Su, and F. Yang, "A precise and practical IP traceback technique based on packet marking and logging," *Journal of Information Science and Engineering*, Vol. 28, 2012, pp. 453-470.
20. M. H. Yang and M. C. Yang, "RIHT: a novel hybrid IP traceback scheme," *IEEE Transactions on Information Forensics and Security*, Vol. 7, 2012, pp. 789-797.
21. CAIDA, "CAIDA's skitter project," http://www.caida.org/tools/skitter/.
22. K. Rothermel and W. Theilmann, "Dynamic distance maps of the internet," in *Proceedings of the 19th Annual Conference of IEEE Communications and Computer Societies*, 2000. pp. 275-284.
23. P. B. Gibbons, C. Faloutsos, M. Faloutsos, C. R. Palmer, and G. Siganos, "The connectivity and fault-tolerance of the Internet topology," in *Proceedings of Workshop*

on Network-Related Data Management; ACM SIG on Management of Data/Principles of Database Systems, 2001, pp. 1-10.
24. Abilene – Visible Backbone, http://pea.grnoc.iu.edu/Abilene/.
25. CAIDA, "The macroscopic internet topology data kit (ITDK)," Caida.org/tools/measurement/skitter/idkdata.xml, 2003.
26. S. Malliga and A. Tamilarasi, "An autonomous framework for early detection of spoofed flooding attacks," *International Journal of Network Security*, Vol. 10, 2008, pp. 39-50.

S. Malliga is working as a Professor in the Department of Computer Science and Engineering, Kongu Engineering College, Tamil Nadu, India. She has completed Ph.D. in 2010 from Anna University, Chennai. Her main research area is network and information security. She has done consultancy project for BPL and offered several courses on latest technology. Currently she is guiding three research scholars. She has also guided many UG and PG projects. She has published 12 articles in international journals and presented more than 25 papers in national and international conferences in her research and other technical areas. She is also interested in cloud and virtualization technolgies.

C. S. Kanimozhi Selvi, Professor, Department of Computer Science and Engineering. Kongu Engineering College, Erode, Tamil Nadu, India, is interested in Data Mining and has received her Ph.D. in association rule mining and classification from Anna University, Chennai. Her current research interests are security issues in cloud and data mining. Her research articles are published in national and international journals. She has involved herself in big data analytics.

S. V. Kogilavani is associated with the Department of Computer Science and Engineering as an Associate Professor at Kongu Engineering College, Tamil Nadu, India. Her research is in information retrieval and summarization. She got her Ph.D. from Anna University, Chennai in the year 2013. She has presented many papers in national and international conferences and published 6 papers in national and international journals. She conducted many courses for the benefits of students.