Tracing Attackers with Deterministic Edge Router Marking (DERM)

Shravan K Rayanchu[1] and Gautam Barua[2]

[1] Samsung India Software Operations,
Bangalore 560052, India
shravan.kr@samsung.com

[2] Dept. of CSE, IIT Guwahati
Guwahati 781039, India
gb@iitg.ernet.in

Abstract. Tracing the attackers in a distributed denial-of-service (DDoS) attack is particularly difficult since attackers spoof the source addresses. We present a novel approach to IP Traceback - Deterministic Edge Router Marking (DERM). The proposed scheme is scalable to thousands of attackers, is very simple to implement at the routers, has no bandwidth overhead and needs minimal processing and storage requirements at the victim. As each complete mark fits into a single packet, our scheme can also be used for per-packet filtering and as a congestion signature in a pushback protocol. The traceback procedure requires a small number of packets and can be performed during the post-mortem analysis of an attack. Only limited co-operation is required from Internet Service Providers (ISP). They do not have to reveal the topology of their internal networks.

1 Introduction

DDoS attacks are among one of the hardest security problems to address because they are simple to implement, hard to prevent, and difficult to trace. Ideally, the network traffic of an attack should include information identifying the sources. The Internet protocol (IP) specifies a header field in all packets that contains the source IP address, which would seem to allow for identifying every packet's origin. However, the lack of security features in TCP/IP specifications facilitates IP spoofing - the manipulation and falsification of the source address in the header. Thus, an attacker could generate offending IP packets that appear to have originated from almost anywhere. IP traceback methods provide the victim with the ability to identify the address of the true source of the packets causing a DoS attack. A perfect solution to this problem is complicated especially because of the use of zombies and reflectors [1]. The exact origin of the attack may never be revealed as even the MAC source addresses may be spoofed. Hence, the traceback schemes try to solve the more limited problem of *identifying the closest router(s) to the attacker(s).*

2 Evaluation Metrics and Assumptions for Traceback Schemes

We now present some of the important evaluation metrics essential in comparing IP Traceback approaches. These were originally proposed in [2].

ISP Involvement: An ideal traceback scheme must be inserted with little infrastructure and operational changes and the actual traceback process must involve little or no burden on the ISP. *Number of attack packets needed for traceback:* Once the attack has been identified, the traceback scheme should require very few packets to identify the attacker. *Post-mortem Analysis:* It must be possible to initiate the traceback procedure and identify the attacker after the attack as the victim might not be in a position to perform the analysis during the attack. *Processing, Bandwidth and Memory requirements:* Processing and memory overhead on routers must be minimal for the practical deployment of the scheme. Since bandwidth is one of the bottlenecks during flooding attacks, the scheme must not introduce additional bandwidth overhead. *Ease of evasion:* It must be very difficult for an attacker who is aware of the scheme to orchestrate an attack that is untraceable. *Ability to handle major DDoS attacks:* This reflects how well a scheme can perform traceback under severe circumstances. An ideal scheme should be able to identify all the attackers involved. *Scalability:* An ideal scheme should be easily scalable.

Assumptions mentioned here are largely borrowed from the previous schemes [3, 4]. The following are the basic assumptions for DERM:

1) An attacker may generate any packet, 2) Attackers may be aware that they are being traced, 3) Packets may be lost or reordered, 4) An attack may consist of just a few packets, 5) Packets of an attack may take different routes, 6) Routers are both CPU and memory limited, 7) Routers are not compromised.

We first propose a basic scheme which we term Basic Deterministic Edge Router Marking (Basic DERM) and then improve the scheme by introducing multiple hash functions.

3 Basic Deterministic Edge Router Marking

The 16 bit packet ID field and the 1 bit RF in the IP header are used for marking packets.Each and every packet that enters the network is marked; this removes the problem of an attacker spoofing any mark. The packet is marked by the edge ingress router to which the source is connected. Every incoming packet is marked whereas outgoing packets are not marked.

3.1 Marking Procedure for Basic DERM

In order to identify an attacker, the victim needs to know the IP address of the attacker which is 32 bits long. But all we have is the 16 bit ID field and the 1 bit RF field. The problem with other packet marking schemes which try to construct the IP address of the attacker (or the router nearest to the attacker) in multiple packets is that the markings cannot be used for filtering purposes as the victim cannot make out anything from the information available in a single packet. Hence, we must try to convey the information about the IP address of the attacker in a single packet. Instead of marking the packets with the IP address of the attacker which is 32 bits long, it would suffice if we send a 16 bit representation of the IP address. Of course, this would mean that there might be some collisions. In Basic DERM, each incoming packet on the ingress router is marked with a 16 bit hash of its own IP address. The ID field is used for this purpose. The RF bit is kept aside as of now. The hash mark (HM(IP)) serves as the representation of the IP address of the edge router. It is assumed that the hash function HM is known to everyone including the DERM enabled routers, all the destinations which would utilize HM and the attackers. It is further assumed that HM is an ideal hash. An ideal hash function minimizes the number of collisions.

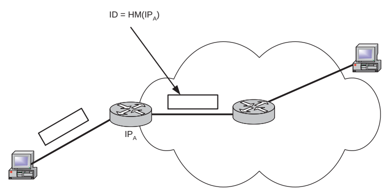


Fig. 1. Marking procedure for Basic DERM

3.2 Reconstruction by the Victim in Basic DERM

The victim has a table RecordTbl, each entry of which consists of the tuple $< HashMark$, $RECV\ bit$, $IngressAddList >$, where $HashMark$ is a possible hash mark and IngressAddList is the list of all ingress edge router addresses that have this hash mark. *RECV bit* is initialized to zero before the attack. This bit indicates whether the victim has received a particular HashMark. The reconstruction by the victim has two phases. One is the filtering phase and the other is the Attacker Identification phase. The filtering phase starts when the victim detects that it is under an attack. As with all the schemes, we assume that there is an intrusion detection system (IDS) which helps us identify the attack packets. Whenever an attack packet is identified, the HashMark of the packet is noted and the corresponding RECV bit in the RecordTbl is marked as 1.

Filtering Phase in Basic DERM: One of the aims of the traceback schemes is to aid the victim during the attack by helping it in filtering out the attack traffic. We note down the HashMarks in the RecordTbl for which the corresponding RECV bit is 1. These HashMarks can be used to identify the attack packets and to filter them. Thus, unlike other schemes, the victim itself can filter the packets without relying on the upstream routers to filter the traffic. Also wherever possible, the upstream routers can use these HashMarks to filter the traffic before the entire bandwidth at the victim is consumed. Protocols like Pushback [5] need some kind of a congestion signature which it uses to identify attack packets. In this case the HashMarks can be used as a congestion signature. Once an attack packet is identified, the Hashmark can be used to filter further attack packets which are now not sent to the IDS, whose load now decreases. The filtering simply consists of checking whether the RECV bit corresponding to the HashMark is 1. If so, the packet may be an attack packet and hence can be dropped. However, there may be collisions with legitimate packets which will also get dropped. These are called *false positives*.

Attacker Identification Phase in Basic DERM: In basic DERM, this involves noting the list of ingress IP addresses corresponding to each HashMark which has the RECV bit set to 1. This may result in many false positives as there will be more than one ingress address corresponding to each HashMark. It is however to be noted that only one packet from the attacker is enough to carry out the Attacker Identification Phase. This is one of the advantages of DERM against schemes like PPM[3].

3.3 Analysis

False positives during the Attacker Identification phase: Here we calculate the number of legitimate user IP addresses that we falsely identify as an attacker. Let M be the number of edge routers and d (=16) be the length of the HashMark. Let N be the number of attackers. If there is only one ingress address corresponding to each HashMark, then there will be no false positives because of the properties of the assumed ideal hash function HM. Hence, the rate of false positives is 0, when M is less than or equal to the number of possible HashMarks, 2^d. Suppose that M is greater than 2^d. The expected number of different HashMarks, $E(HashMarks)$ which have $RECVbit = 1$ after the Filtering phase can be thought of as the expected number of faces turning up on a 2^d sided die after N throws. This is a special case of the classical occupancy problem which is discussed in [7]. The expected number of different HashMarks is given by

$$E(HashMarks) = 2^d - 2^d(1 - 1/2^d)^N$$

Let the number of ingress addresses that match a particular HashMark be $N_d = M/2^d$. Thus the number of false positives would be

$$E(false\ positives) = (2^d - 2^d(1 - 1/2^d)^N) * N_d - N$$

False positives during the Filtering Phase: Here we calculate the number of legitimate user packets that might get discarded during the filtering phase, as a result of falsely identifying them as attackers. In Basic DERM, the number of false positives during the Filtering Phase is the same as the number of false positives during the Attacker identification Phase.

Storage requirements for RecordTbl: The amount of storage required for $IngressAddList$ is $N_d * 32$ bits and one bit for storing the $RECV$ bit.Hence, the total amount of storage required is $(N_d * 32 + 1) * 2^d$ bits.

4 Multiple Hash DERM

In order to reduce the false positives that arise while identifying attackers, we modify the scheme to use multiple hash functions of an IP address, HM_1, HM_2 .. HM_f. As before all these are assumed to be ideal hash functions. The 16 bit field now consists of a d bit HashMark and a log(f) bit Hash function identifier, where f is the number of Hash functions used. Hence, $d + log(f) = 16$

4.1 Marking Procedure for Multiple Hash DERM

At startup, each DERM enabled router calculates f HashMarks by hashing its IP address with the functions HM_1, HM_2, .. HM_f. The router deterministically marks the packets with any one of these f HashMarks. The processing required for each packet would be limited to generating a small random number from 1 to f and then inserting the corresponding d bit HashMark in the packet along with the $log(f)$ bit hash function identifier.

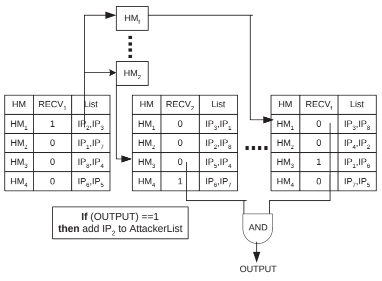
4.2 Reconstruction by the Victim in Multiple Hash DERM

Instead of having a single $RecordTbl$, each victim must have f tables $RecordTbl_1$, $RecordTbl_2$, .. $RecordTbl_f$. Each of the tables will have as entries the tuple $< HashMark, RECV\ bit, IngressAddList >$ which have the same meaning as stated before. As before, the reconstruction will consist of two phases, the Filtering Phase and the Attacker Identification Phase.

Filtering Phase in Multiple Hash DERM: As already stated, we assume that there is an attack identifying algorithm that gives us the identified attack packets. Whenever the victim gets such a packet, the hash function identifier is noted and then the corresponding $RecordTbl$ is identified. Then, the $RECV\ bit$ corresponding to the HashMark in the packet is set to 1. These HashMarks are then used for aiding the victim in filtering the attack traffic. It is important to note that the number of false positives in this case would more than Basic DERM; in fact it would be multiplied by f times. This is because of the fact that during filtering the victim has to decide whether to drop the packet or not depending on the HashMark of the packet and nothing else. As the number of different HashMarks would now be multiplied by f, the false positives would also increase f times.

Attacker Identification Phase in Multiple Hash DERM: In this phase, the victim first collects all the ingress IP addresses in $RecordTbl_1$ for which the corresponding $RECV\ bit$ is set to 1. Then it takes one IP address at a time and calculates HM_2, HM_3 .. HM_f for that IP address. Now if the $RECV$ bits corresponding to the HashMarks in the respective $RecordTbls$ are set i.e. if the

RECV bit for $HashMark_2$ is set in *RecordTbl*$_2$, *RECV bit* for $HashMark_3$ is set in *RecordTbl*$_3$, .. *RECV bit* for $HashMark_f$ is set in *RecordTbl*$_f$, then that IP address is identified as the attacker's, else it is discarded. Because of these additional checks, the number of false positives now decrease.



HM	$RECV_1$	List
HM_1	1	IP_2,IP_3
HM_2	0	IP_1,IP_7
HM_3	0	IP_8,IP_4
HM_4	0	IP_6,IP_5

HM	$RECV_2$	List
HM_1	0	IP_3,IP_1
HM_2	0	IP_2,IP_8
HM_3	0	IP_5,IP_4
HM_4	1	IP_6,IP_7

HM	$RECV_f$	List
HM_1	0	IP_3,IP_8
HM_2	0	IP_4,IP_2
HM_3	1	IP_1,IP_6
HM_4	0	IP_7,IP_5

Fig. 2. Attacker Identification in Multiple Hash DERM

4.3 Analysis

False positives during the Attacker Identification phase: From the previous analysis, the expected number of false positives in *RecordTbl*$_1$ is:

$$E(false\ positives) = (2^d - 2^d(1 - 1/2^d)^N) * N_d - N$$

Consider *RecordTbl*$_2$, the expected number of different HashMarks which have *RECV bit* set to 1 for this table is:

$$E(HashMarks) = 2^d - 2^d(1 - 1/2^d)^N$$

Consider one of the false positives generated in *RecordTbl*$_1$. For this to still be a false positive it must match one of the above HashMarks. Thus the probability that HM_2 of this IP address will be accepted as a false positive after taking *RecordTbl*$_2$ into consideration is $E(HashMarks)/2^d$ Therefore, the probability that a particular false positive of *RecordTbl*$_1$ would still be a false positive after considering all the remaining $f - 1$ tables is:

$$(E(HashMarks)/2^d)^{f-1}$$

Hence, the number of false positives that arise while identifying the attacker is given by,

$$falsepos = ((E(HashMarks)N_d - N)(\frac{E(HashMarks)}{2^d})^{f-1}$$

Thus, the maximum number of attackers N_{MAX} (actually attacking networks) that we can afford such that, the number of false postives are less than 1% of N can be obtained by setting $falsepos$ to $0.01N$ and solving for N.

False Positives During the Filtering Phase: The number of false positives during the filtering phase of Multiple Hash DERM would simply be f times the number of false positives during the filtering phase of Basic DERM: $(E(HashMarks) * N_d - N) * f$.

Storage Requirements for the Tables Storage requirements would be simply be f times the storage required for each table i.e. $((N_d * 32) + 1 + d) * 2^d * f$ bits.

Expected number of packets required to identify an attacker: In case of Basic DERM, only a single packet is required to carry out the Attacker Identification procedure. However, in Multiple Hash DERM we require that the HashMarks of all the f functions are collected. The expected number of packets $E(f)$ that are required to be sent by a particular attacker is given by a Coupon Collector problem discussed in [7]:

$$E(f) = f(\frac{1}{f} + \frac{1}{f-1} + \frac{1}{f-2} + ... + 1) \approx f(log(f) + 0.577)$$

Figure 3 shows the plot of the maximum number of attackers N_{MAX} that we

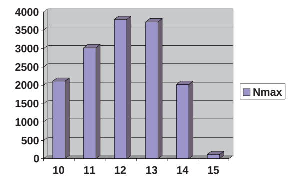


Fig. 3. Plot of N_{MAX} against d for $M = 2^{17}$

can afford against the length of the HashMark d. If we assume that there can be $128,000$ ingress edge router addresses, then the maximum value of $N_{MAX} = 3800$ occurs when $d = 12$ (and so there are 16 hash functions). The storage requirement at the victim end for this value of d is found to be 8 MB and the expected number of packets required to be sent by an attacker for it to be identified, $E(f)$, is found to be 54. This shows that the algorithm can handle a large number of attackers with reasonable space and time requirements. The identification of attackers can also be done with a relatively small number of packets.

On Low Volume Attacks

As stated earlier, Multiple Hash DERM requires only $E(f)$ number of packets to get all the f HashMarks of the attacker. In this section we discuss how DERM fares for attacks constituting less than $E(f)$ packets. Let N be the number of attackers and let each attacker send m packets, where $m \leq E(f)$. Now, the expected number of different HashMarks that the victim would receive from a particular attacker is given by,

$$f' = f(1 - (1 - \frac{1}{f})^m)$$

The expected number of different HashMarks that have $RECV$ set to 1 in each RecordTbl is,

$$E(HashMarks) = 2^d(1 - (1 - \frac{1}{2^d})^{\frac{f'N}{f}})$$

In order to identify the attackers, we carry out the attacker identification procedure as before. Since all the f HashMarks are not received, we cannot identify the attackers by checking if the corresponding $RECV$ bit is set to one in the remaining $f - 1$ RecordTbls. Instead, we identify the attackers by checking if the $RECV$ bit is set in atleast $f' - 1$ RecordTbls. However, this would result in the increase of false positives. The number of false positives is given by,

$$(E(HashMarks) * N_d - N)\left(\frac{E(HashMarks)}{2^d}\right)^{f'-1}$$

5 Related Work and Comparison

5.1 The Pushback Protocol

The main idea behind the pushback protocol [5] is that if routers can detect packets belonging to an attack, they can then drop only those packets and thus the DDoS problem would be solved. Bad traffic is characterized by an *attack signature* which we strive to identify; what can be really identified is the *congestion signature*, which is the set of properties of a subset of traffic identified as causing problems. The authors in [8] use the destination address (victim's address) as the congestion signature. Thus even legitimate traffic destined for the victim is automatically dropped. One of the advantages of DERM is that it can be used as an effective congestion signature in the pushback protocol.

5.2 Packet Marking Schemes

In these schemes, basically some traceback data is inserted in each packet so that a victim can use this information to identify the attacker. To be effective, packet marking should not increase a packets' size. Furthermore, packet-marking methodologies must be secure enough to prevent attackers from generating false markings.

Probabilistic Packet Marking (PPM): In this scheme [3], the routers enroute probabilistically mark packets. As with DERM, the 16 bit ID field in the IP header is used for this purpose. Partial address information of the edges of a router are marked. A victim reconstructs the attack path with these marked packets. There are many advantages of DERM over PPM. The number of packets required in DERM in order to identify the attacker is much less (1 packet for Basic DERM and $E(f)$ packets for Multiple Hash DERM) as compared to PPM which requires a large number of packets. In PPM, the victim can reconstruct the path to the attacker based on multiple packet markings, but there is no guarantee that an individual packet will contain a marking that can identify an attacker. One of the major disadvantages of PPM is *Mark Spoofing*. If an attacker injects a packet with erroneous information and no router on the path marks the packet, then the spoofed marks from the attacker would also reach the victim.

Deterministic Packet Marking (DPM): In DPM ([4, 9]), only the edge routers participate in the marking procedure. As in DERM, interfaces are used as atomic units of traceback. DPM tries to construct the ingress address of the router closest to the source by fragmenting the IP address and sending it in two packets. The ID field is used to carry one half of an IP address and the RF bit is used to denote whether it is the first half or the second half of the IP address. A hash of the IP address is also sent along with the fragmented IP address to aid the victim in the reassembly procedure. Constructing an IP address from fragments would require trying out all the possible permutations. This results in a high number of false positives while assembling the fragments, especially in the case of a DDoS attack where multiple fragments from multiple attackers are collected. Per-packet filtering is not possible in DPM.

Pi and StackPi : Path Identification Mechanisms: In these schemes ([10, 6]), each packet is marked deterministically by the enroute routers. The StackPi mark created by a router is a 2 *bit* message digest of the concatenation of the IP addresses of the previous router and the current router. Only the last eight marks ($16/2 = 8$) made by the routers along the path reach the destination. In any case, the packets travelling along the same path will have the same marking so that the victim needs only to identify the StackPi marks of the attack packets and filter out all further packets with the same marking. As multiple routes may exist from a source to a destination, multiple sets of marks will have to be handled. All routers need to participate in the scheme rather than only edge routers and this is a restriction. As is the case with StackPi, DERM also requires constructing a table (RecordTbl) which consists of mapping the HashMarks with the IP addresses. This task is much easier in DERM as we have to collect the edge router IP addresses and simply hash them to get the corresponding HashMarks. Also, since these marks are only dependent on the edge routers we can have the table constructed at a particular location and distribute them. But in StackPi, the marks from a particular source will be different for two different hosts. Hence, each host has to construct its own table. Moreover, in StackPi we are dealing with addresses of hosts (not edge routers) which will be much more difficult to maintain.

6 Conclusion and Future work

In this paper, we have presented Deterministic Edge Router Marking (DERM), a technique to defend against DDoS attacks. The marking procedure for the routers is simple and can easily be implemented. The processing overhead at the victim during reconstruction is also very little. The reconstruction by a victim is done in two phases, a filtering phase and an attacker identification phase. The filtering phase involves setting a flag in a table based on marks in incoming packets to help identify attack traffic and the using of these marks for filtering the attack traffic. The Attacker Identification Phase consists of noting down the IP addresses of ingress packets and checking them against filter table entries to see whether the corresponding flag bits are set to 1. Analysis shows that about 3800 attackers can be handled with less than 1% false positives. This compares favourably with other known techniques and where scaling is a major issue. The storage requirements on the victim side are not very high. The expected number of packets required to identify an attacker is also small. Further work involves dealing with reflector attacks, where victim sites are used to *reflect* attack packets to camouflage the actual sources. The issue of compatibility of the scheme with IP fragmentation is another task for the future.

References

1. Paxson, V.: An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks. Computer Communication Review 31(3) (2001)
2. Belenky, A., Ansari, N.: On IP Traceback. IEEE Communications Magazine 41(7) (2003) 142–153
3. Savage, S., Karlin, A.: Practical Network Support for IP Traceback. ACM SIGCOMM (2000) 295–306
4. Belenky, A., Ansari, N.: IP Traceback with Deterministic Packet Marking. IEEE Communication Letters, 7, (2003)
5. Ioannidis, J., Bellovin, S.M.: Implementing Pushback: Router-based Defense against DDoS Attacks. Proceedings of the Symposium on NDSS 2002, San Diego, CA. (2002)
6. Yaar, A., Perrig, A., Song, D.: StackPi: A New Defense Mechanism against IP Spoofing and DDoS Attacks. Technical Report, Carnegie Mellon University (2003)
7. Feller, W.: An Introduction to Probability Theory and Its Applications (2nd edition). Vol 1 (1966)
8. Mahajan, R., et. al: Controlling High Bandwidth Aggregates in the Network. Computer Communication Review 32(3), (2002) 62-73
9. Belenky, A., Ansari, N.: Tracing Multiple Attackers with Deterministic Packet Marking (DPM). Proceedings of the IEEE PACRIM '03, Victoria, B.C., Canada. (2003)
10. Yaar, A., Perrig, A., Song, D.: Pi: A path identification mechanism to defend against DDoS attacks. IEEE Symposium on Security and Privacy (2003)