# Tracing Multiple Attackers with Deterministic Packet Marking (DPM)

Andrey Belenky and Nirwan Ansari
Advanced Networking Laboratory, ECE Department, NJIT, Newark, NJ 07102, USA
Email: avb0168@oak.njit.edu, Nirwan.Ansari@NJIT.EDU

*Abstract*— **The rising threat of cyber attacks, especially distributed denial-of-service (DDoS), makes the IP Traceback problem very relevant to today's Internet security. IP Traceback is one of the security problems associated with identifying the source of the attack packets. This work presents a novel approach to IP Traceback - Deterministic Packet Marking (DPM). The proposed approach is scalable, simple to implement, and introduces no bandwidth and practically no processing overhead on the network equipment. It is capable of tracing thousands of simultaneous attackers during DDoS attack. All of the processing is done at the victim. The traceback process can be performed post-mortem, which allows for tracing the attacks that may not have been noticed initially. The involvement of the Internet service providers (ISP) is very limited, and changes to the infrastructure and operation required to deploy DPM are minimal. DPM performs the traceback without revealing the internal topology of the provider's network, which is a desirable quality of a traceback scheme.**

## I. Introduction

In recent years much interest and consideration has been paid to the topic of securing the Internet infrastructure that continues to become a medium for a broad range of transactions. A number of approaches to security have been proposed, each attempting to mitigate a specific set of concerns. After several high-profile Distributed Denial of Service (DDoS) attacks on major US web sites in 2000, numerous IP traceback approaches have been suggested to identify the attacker(s) [1]. The previously proposed schemes can be categorized in two broad groups. One group of the solutions relies on the routers in the network to send their identities to the destinations of certain packets, either encoding this information directly in rarely used bits of the IP header, or by generating a new packet to the same destination. The biggest limitation of this type of solutions is that they are focused only on flood-based DoS and DDoS attacks, and cannot handle attacks comprised of a small number of packets. Additionally, for the large scale DDoS attacks, these schemes are not very effective [2], [1]. The second group of solutions involves centralized management, and logging of packet information on the network. Solutions of this type introduce a large overhead, and are complex and not scalable.

In this article, Deterministic Packet Marking (DPM), a new approach to IP traceback, is introduced. Even though DPM is classified as a scheme of the first type, the substantial differences of having only edge routers perform the marking allow DPM to perform traceback with only a few packets from the attacker and be capable of tracing thousands of attackers simultaneously. The rest of the paper is structured in the following way: Section II introduces the principle of DPM; Section III describes the limitation of the basic DPM associated with tracing multiple attackers simultaneously; Section IV explains the main principle in dealing with multiple simultaneous attackers and other instances of source address inconsistency; Section V introduces and analyzes the performance of a modification to DPM that enables the traceback of simultaneous attackers; Section VI presents the conclusion.

## II. Basic Deterministic Packet Marking (DPM)

The basic DPM is a packet marking algorithm, which was first introduced in [3]. This section provides the general principle behind DPM and discusses the most basic implementation of the proposed scheme.

### A. Assumptions

The assumptions in this section were largely borrowed from [4]. The two key assumptions driving this effort are:

- An attacker may generate any packet
- Routers are both CPU and memory limited

### B. DPM Principle

As mentioned above, DPM is a packet marking algorithm. The 16-bit Packet ID field and 1-bit Reserved Flag (RF) in the IP header will be used to mark packets. *Each* packet is marked when it enters the network. This mark remains unchanged for as long as the packet traverses the network. The packet is marked by the interface closest to the source of the packet on an edge ingress router, as seen in Figure 1. The mark is a partial address information of this interface, and will be addressed later in Section II-C. The interface makes a distinction between incoming and outgoing packets. Incoming packets are marked; outgoing packets are not marked. This ensures that the egress router will not overwrite the mark in a packet placed by an ingress router.

For illustrative purposes, assume that the Internet is a network with a single administration. In this case, only interfaces closest to the customers on the edge routers will participate in packet marking. Every incoming packet will be marked. Should an attacker attempt to spoof the mark in order to deceive the victim, this spoofed mark will be overwritten with a correct mark by the very first router the packet traverses.

0-7803-7978-0/03/$17.00 

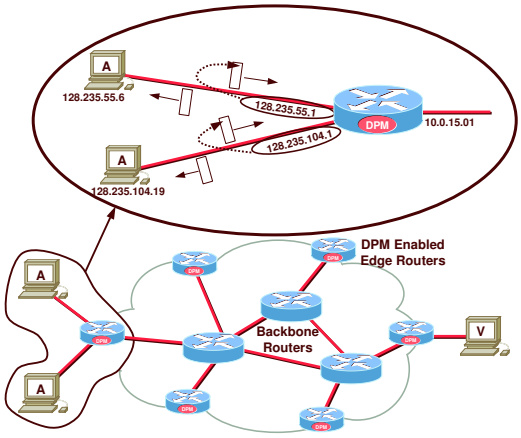


Fig. 1. Deterministic Packet Marking

### C. Procedure

A 32-bit IP address needs to be passed to the victim. A total of 17 bits are available to pass this information: 16-bit ID field and 1-bit RF. Clearly, a single packet would not be enough to carry the whole IP address in the available 17 bits. Therefore, it will take at least two packets to transport the whole IP address. An IP address will be split into two segments, 16 bits each: segment 0 – bits 0 through 15, and segment 1 – bits 16 through 31. With probability of 0.5, the ID field of each incoming packet will be populated with either of those two segments, and then the remaining bit, designated to be a segment number, will be set to the number of the segment in the mark.

At the victim, we suggest that the table matching the source addresses to the ingress addresses is maintained. When a marked packet arrives to the victim, the victim will first determine if the given packet is an attack packet. If it is, the victim would check to see if the table entry for a source address of this packet already exists, and create it if it did not. Then, it would write the segment from the mark, according to the value of the segment number, into the ingress IP address value. After both segments corresponding to the same ingress address have arrived to the destination, the ingress address for a given source address becomes available to the victim.

## III. Shortcomings of the Basic DPM with respect to DDoS Attacks

The problem with the basic DPM, which causes the inability to handle a certain type of DDoS attacks, lies in the fact that the destination would associate segments of the ingress address with the source address of the attacker. If it could be guaranteed that only one host participating in the attack has a given source address, even though it might have been spoofed, and that the attacker would not change its address during the attack, there would be no problems. There are two situations when the reconstruction procedure of the basic DPM will fail. First, consider the situation when two hosts with the same Source Address (SA) attack the victim. The ingress addresses corresponding to these two attackers are $A_0$ and $A_1$, respectively. The victim would receive four address segments: $A_0[0]$, $A_0[1]$, $A_1[0]$, and $A_1[1]$. The victim, not being equipped to handle such attack would eventually reconstruct four ingress addresses, since four permutations are ultimately possible: $A_0[0].A_0[1]$, $A_0[0].A_1[1]$, $A_1[0].A_0[1]$, and $A_1[0].A_1[1]$, where '.' denotes concatenation. Only two of the four would be valid.

A typical metric of evaluation of the traceback schemes for DDoS attacks is the *rate of false positives* or *false positive rate*. In the context of DPM, a false positive is defined as an incorrectly identified ingress address. The rate of false positives refers to the ratio of the incorrectly identified ingress addresses to the total number of identified ingress addresses. In the example described above, the false positive rate for that particular attack is 50%. Clearly, the false positive rate would increase even further if the number of attackers, with the same SA, was larger.

Second, consider a (D)DoS attack, where the attackers change their source addresses for every packet they send. The basic DPM will be unable to reconstruct any valid ingress addresses, since none of the entries in the $IngressTbl$ would have a complete ingress address.

## IV. General Principle of Handling DDoS Attacks

A general principle in handling (D)DoS attacks of these types is to rely *only* on the information transferred in the DPM mark. The DPM Mark can be used to not only transfer the bits of the ingress address but also some other information. This additional information should enable the destination to determine which ingress address segments belong to which ingress address.

The reconstruction procedure utilizes the data structure called *Reconstruction Table* ($RecTbl$). The destination would first put the address segments in $RecTbl$, and then only after correctly identifying the ingress address, out of many possible address segments permutations, would transfer it to $IngressTbl$.

## V. Hash-based DDoS Modification to DPM

The scheme described in this section utilizes a hash function, $H(x)$. To simplify the performance analysis, the hash function is assumed to be *ideal*. An ideal hash function minimizes the chances of collision, an occurrence when two different ingress addresses result in the same hash value. In other words, $H(x)$ is assumed to produce a collision only after all possible hash values have been produced. It is also assumed that the hash function is known to everybody, including all DPM-enabled interfaces, all destinations, which intend to utilize DPM marks for traceback, and the attackers. The constraint of 17 bits still remains, so a longer digest would result in fewer bits of the actual address transmitted in each mark, and consequently, the higher number of packets required for traceback.

### A. Mark Encoding

Recall that in the basic DPM, the ingress address was divided in two segments. In this modified scheme, the ingress address is divided in $k$ segments. Also, more bits would

be required to identify the segment. Instead of a single bit required for two segments in the basic DPM, $log_2(k)$ would be required for this scheme. The remaining bits would be used for the digest. Independently of what segment of the address is being sent to the victim, the digest portion of the mark will always remain the same for a given DPM interface. This would enable the victim to associate the segments of the ingress address with each other to reconstruct the whole address.

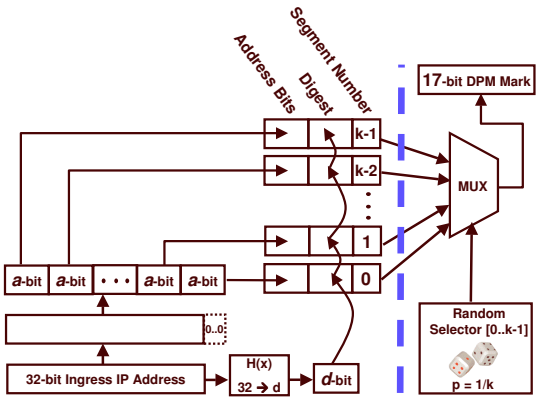


Fig. 2. Single Digest DDoS Modification

Figure 2 shows the schematics of the approach. The DPM mark consists of three fields: $a$-bit address bits field, $d$-bit digest field, and $s$-bit segment number field. Some padding may be required so that the address is split into segments with equal length. For example if the ingress address is divided in 5 segments, it would be necessary to pad it with '000' to make it 35-bit long.

At startup the DPM-enabled interface prepares $k$ marks for all segments of the address. A $d$-bit hash value, or digest, of the ingress address is calculated once and then inserted in the digest field of every mark. Each of $k$ marks will have address bits set to a different segment of the ingress address. The segment number field will be set to the appropriate value. These operations are shown to the left of the bold dotted line in Figure 2. The processing required for every packet will be limited to generating a small random number from 0 to $k-1$ and inserting a corresponding mark into the packet header.

### B. Reconstruction by the Victim

The reconstruction procedure of this scheme will consist of two separate processes: Mark Recording and Ingress Address Recovery. The reason for separating these two tasks is the fact that the attack packets may arrive to the destination faster than they can be analyzed. The mark recording process will set the appropriate bits in $RecTbl$ to indicate which marks arrived to the destination. Address recovery will check those bits and will compose address segment permutations and determine which ones are valid ingress addresses.

A reconstruction table $RecTbl$ is a $2^{17}$ bit structure, where every possible mark can be uniquely represented. It consists of $2^d$ areas. Each area consists of $k$ segments, and each segment

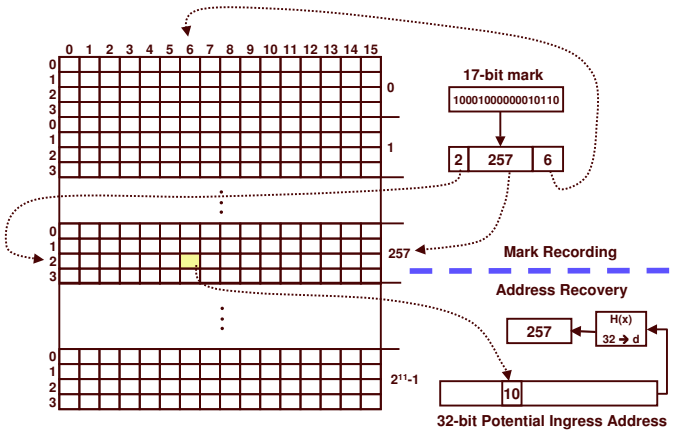


Fig. 3. $RecTbl$ with $k$=16, $d$=11, $a$=2; Mark Recording; Address Recovery

consists of $2^a$ bits. Figure 3 shows an example of $RecTbl$, where $k$, $d$, and $a$ are 16, 11 and 2, respectively. When a mark becomes available to the mark recording process, it sets the appropriate bit in the $RecTbl$. An ingress address can possibly hash into $2^d$ digest values. The digest is extracted from the mark and the area where the bit will be set is determined. The segment number field in the mark indicates the segment in the $RecTbl$ area, where the appropriate bit would be set. Finally, the value of the address bits in the mark indicates the actual bit, which will be set to '1'. This process is repeated for every mark.

The address recovery process will analyze each area of the $RecTbl$. Once again, it runs independently from the mark recording process, which allows post-mortem traceback. The value of a certain bit in $RecTbl$ indicates if the the corresponding mark arrived to the victim. For example, bit 2 in segment 6 of area 257 set to '1' means that there is an ingress address of interest, with digest of 257 having segment 6 equal to '10'$_2$ as shown in Figure 3. This segment has to be combined with other segments of this area in order to create permutations of segments. Hash function, $H(x)$, is applied to each of these permutations. If the result matches the area number, which is actually the digest, embedded in the marks (in this example 257), then the recovery process concludes that this permutation of segments is in fact a valid ingress address.

The address recovery process is an infinite loop, which starts from area 0 and performs identical processing for all areas. In every area, it would first look at segment 0, bit 0. If this bit is set to '1', the process will conclude that a potential address has segment with address bits '00'$_2$ in segment 0. The process will then examine segment 1 of this area. If, for example, bits 1 and 3 are both set to '1' in this segment, then it is concluded that the potential ingress address has '01'$_2$ or '11'$_2$ in segment 1. Therefore, there are two permutations of the first two segments, which are potentially the first two segments of a valid ingress address: '0001'$_2$ and '0011'$_2$. The process moves on to the other segments in this area of $RecTbl$. The address recovery process has to go through all permutations within a given area before moving on to another area. $H(x)$ is applied

to every permutation, and only if the result matches the area number, the permutation is transferred to the $IngressTbl$.

### C. Analysis

In this section, the number of attackers, which this hash-based modified scheme can traceback, with the false positive rate limited to 1%, is evaluated. Let us examine the origin of false positives. If there is only one ingress address with a given digest, there will be no false positives because of the properties of the assumed ideal hash functions. Therefore, the rate of false positives is 0 when the number of attackers, $N$, is less or equal to the number of possible digests, $2^d$.

If $N$ is greater than $2^d$, then the expected number of ingress addresses participating in the DDoS attack resulting in the same digest would be $\frac{N}{2^d}$. Since there would be more than one address resulting in the same digest, each segment associated with this digest would have a certain number of values. For example, if two addresses have the same digest, segment 0 in the area of the $RecTbl$ corresponding to this digest could have either one or two bits set to '1'. If segment 0 in these two addresses is the same, then there would be only one bit set to '1', and if segment 0 of one address is different from segment 0 of the second address, then two bits will be set to '1'. The expected number of values that a segment will assume can be thought of as the expected number of the faces turning up on a $2^a$-sided die after $\frac{N}{2^d}$ throws. This is a special case of a classical occupancy problem discussed in [5]. The expected number of different values the segment will take is

$$2^a - 2^a \left(1 - \frac{1}{2^a}\right)^{\frac{N}{2^d}}.$$

To get the expected number of all permutations of address segments for a given digest, this number has to be risen to the power $k$, since there are $k$ segments in total. Recall that after a permutation of segments is obtained, the hash function $H(x)$ is applied to it, and if the result does not match the original digest, that permutation is not considered. The expected number of permutations that result in a given digest for a given area of the $RecTbl$ is

$$\frac{\left[2^a - 2^a \left(1 - \frac{1}{2^a}\right)^{\frac{N}{2^d}}\right]^k}{2^d}.$$

The number of false positives for a given area would be the total number of permutations matching the digest less the number of valid ingress addresses with this digest, $\frac{N}{2^d}$. The total number of false positives is obtained by multiplying the number of false positive for a singe area by the number of areas, $2^d$. This number has to be less than 1% of $N$. Therefore the following inequality has to be solved for $N$:

$$\left[2^a - 2^a \left(1 - \frac{1}{2^a}\right)^{\frac{N}{2^d}}\right]^k - N \leq 0.01 \times N.$$

Recall that $a$ and $d$ can be expressed in terms of $k$. The maximum $N$, which would satisfy this inequality, $N_{MAX}$, is very difficult to express in terms of $k$. However, it is possible to find $N_{MAX}$ by substitution.

Another important consideration is the expected number of datagrams required for the reconstruction. This number will be related to $k$, the number of segments that the ingress address was split. The larger the $k$, the more different packets it would be required for the victim to receive in order to reconstruct the ingress address. The expected number of datagrams, $E[D]$, required to be marked by a single DPM-enabled interface in order for the victim to be able to reconstruct its ingress address is given by a Coupon Collector problem discussed in [5]:

$$E[D] = k\left(\frac{1}{k} + \frac{1}{k-1} + \ldots + 1\right)$$

Table I provides the summary of these parameters.

| $a$ | $k$ | $s$ | $d$ | $N_{MAX}$ | $E[D]$ |
|---|---|---|---|---|---|
| 1 | 32 | 5 | 11 | 2048 | 130 |
| 2 | 16 | 4 | 11 | 2048 | 55 |
| 4 | 8 | 3 | 10 | 1066 | 22 |
| 8 | 4 | 2 | 7 | 139 | 8 |
| 16 | 2 | 1 | 0 | 1 | 2 |

TABLE I

RELATIONSHIP BETWEEN $a$, $k$, $s$, $d$, $N_{MAX}$, AND $E[D]$

## VI. CONCLUSION

In this article, we have presented the modification to the basic DPM scheme introduced in [3]. The modification requires negligible additional processing overhead on the routers. Other important benefits of DPM such as scalability and minimal ISP involvement are preserved. While the expected number of datagrams required for tracing a singe attacker has increased to 55, it is still capable of tracing most real attacks on the Internet.

## VII. ACKNOWLEDGEMENTS

This work has been supported in part by the NJ Commission of Science and Technology via NJ Wireless Telecommunications Center and NJ Commission of Higher Education via the NJI-Tower Project.

## REFERENCES

[1] A. Belenky and N. Ansari, "On IP traceback," *IEEE Communications Magazine*, vol. 41, no. 7, July 2003, to appear.

[2] V. Paxson, "An analysis of using reflectors for distributed denial-of-service attacks," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 3, pp. 38–47, July 2001.

[3] A. Belenky and N. Ansari, "IP traceback with deterministic packet marking," *IEEE Communications Letters*, vol. 7, no. 4, pp. 162–164, April 2003.

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

[5] W. Feller, *An Introduction to Probability Theory and Its Applications*, John Wiley & Sons, Inc., 3rd edition, 1968.