An Efficient Probabilistic Packet Marking Scheme for IP Traceback

Basheer Duwairi, Anirban Chakrabarti, and Govindarasu Manimaran

Department of Electrical and Computer Engineering
Iowa State University, Ames, IA 50011, USA
{dbasheer,anirban,gmani}@iastate.edu

Abstract. Denial of Service (DoS) attacks represent a major threat to the availability of Internet services. Identifying the sources of these attacks is considered an important step toward a DoS-free Internet. In this paper, we propose a new scheme, called *Distributed Link-List Traceback*, which combines the good features of probabilistic packet marking [6] and Hash-based traceback [9]. The main idea used in the scheme is to preserve the marking information at intermediate routers in such a way that it can be collected in an efficient manner. We evaluate the effectiveness of the proposed scheme for various performance metrics through combination of analytical and simulation studies. Our studies show that the proposed scheme requires small number of packets, adjustable amount of memory. At the same time, offers high attack source detection percentage.

1 Introduction

In DDoS attacks, the attacker's machine (the master) instructs previously compromised innocent machines (the slaves) to aggressively overwhelm the victim by high volume streams of flooding packets with faked IP source addresses, leaving the victim with no clue about the true sources of these packets. This distributed anonymous nature of the attack helps the attacker to stay behind the scenes. Attack traceback, which can be defined as the process of identifying the true physical sources of attack packets, has emerged as a promising solution to DoS attacks. This has the following benefits: first, isolating or even shutting down the attack facility, which greatly reduces the impact of the ongoing attack or stopping it completely. Second, holding attackers responsible for abusing the Internet. Personal identification of attackers can be done by further investigation and analysis of the compromised systems discovered by the attack traceback process.

The stateless nature of the Internet combined with the destination oriented IP routing increases the difficulty of tracing attacks back to their sources. This problem is also complicated by the fact of having millions of hosts connected to the Internet, which implies a huge search space. The imminent threats imposed by DoS attacks call for efficient and fast traceback schemes. A good traceback scheme should provide accurate information about routers near the attack source rather than those near to the victim, recognize and exclude false information injected by the attacker, avoid using large amount of attack packets to construct the attack path or attack tree, avoid imposing high processing and storage overhead at intermediate routers, and if packet information is to be maintained at intermediate routers then collecting this information must be efficient.

In this paper, we develop a novel concept called *Distributed link-list* (DLL), which refers to the process of keeping track of a selected set of routers that were involved in forwarding certain packet by establishing a temporary link between them in a distributed manner. We utilize this concept and develop a novel traceback scheme, called *Distributed Link-List Traceback* (DLLT) that combines the desirable features of PPM [6] and hash-based traceback [9]. The rest of this paper is organized as follows. In the next section, we discuss the related work. In section 3, we present the proposed work: distributed link-list traceback. In section 4, we provide theoretical analysis. In section 5, we describe the simulation studies. Finally, conclusions are drawn in section 6.

2 Related Work

Traceback schemes [6][9] [7][4] [2][8][3] usually rely on router assistance to determine the path followed by attack packets and eventually identify the attack source. For example, in PPM [6], routers mark forwarded packets (i.e., write their own IP addresses into the packets) probabilistically, such that the victim can reconstruct the attack path after receiving huge amount of packets. In hash-based traceback [9], bloom filters [1] were used to save packet digests at intermediate routers to be collected and searched when attack is detected.

PPM requires very large number of packets to be collected before starting the traceback process. This is due to the fact of allowing routers to overwrite marking information written by previous routers. Also, the ability of the attackers to spoof the marking information represents a major weakness of PPM [5]. In Hash-based scheme, processing of every packet passing through, imposes significant router overhead. Also, the method employed to download packet information from network routers is inefficient and requires special resources. Moreover, a major concern in Hash-based traceback is the small window of time through which packets can be successfully traced.

The main contribution of this paper is a novel concept called *Distributed Link-List (DLL)* and using which we developed a new traceback scheme, called *Distributed Link-List Traceback* (DLLT). DLLT exhibits the features of PMM [6] in the sense that routers probabilistically mark forwarded packets. Also, it exhibits the features of Hash-based scheme [9] in the sense that processing and storage at intermediate routers are necessary. The significance of DLLT is due to drastic reduction of the number of packets required in the traceback process compared to PPM, and the adjustable memory requirement and efficient marking information collection compared to Hash-based traceback.

3 Proposed Solution: Distributed Link-List Traceback

Distributed Link-List Concept: The main idea of DLL is to keep track of some of the routers that were involved in forwarding certain packet by establishing a temporary link between them in a distributed manner. DLL is based on "store, mark and forward" approach. A single marking field is allocated in each packet. Any router that decides to mark the packet, stores the current IP address found in the marking field along with the packet ID in a special data structure called *Marking Table* maintained at the router, then marks the packet by overwriting the marking field by its own IP address, and then forwards the packet as usual. Any router that decides not to mark the packet just forwards it.

A link list is inherently established because the marking field serves as a pointer to the last router that did the marking for the given packet and the marking table of that router contains a pointer (i.e., the IP address) of the previous marking router and so on. Therefore, each packet received by the destination contains the start point of a link list that is part of the packet path. We call it distributed link-list because each router decides by its own to be on the list or not according to certain marking probability.

Details of Distributed Link-List Traceback: Distributed Link-List Traceback (DLLT) uses DLL concept to keep track of the routers that have been involved in forwarding malformed packets toward the victim. DLLT employs a probabilistic marking and storage scheme. When a router receives a packet, it makes a decision based on certain marking probability q of whether to mark the packet (i.e., write some information, called the marking information, into the packet) or not. Whenever a router decides to mark a packet it has to store the marking information found in the packet before remarking it. Therefore, packet marking and storage is an integrated procedure. Before going into details of this procedure, we show the main data structure used for storing packet information.

Logging packet information at intermediate routers is not a new idea. Storing packet digests was considered in [9]. However, our storage scheme is probabilistic in nature, which means that only fraction of the traffic is to be logged at each router. Also, we store this information in such a way as to ensure that it can be collected in a predetermined manner. We borrow the idea of using bloom filters [1] from [9], and we modify it to satisfy our requirements:

– Storing the packet digests to be able to verify that a given packet has been actually forwarded by the router.
– Mapping the digests of a given packet to certain memory location where the marking information of that particular packet can be stored.

The first requirement can be achieved exactly the same way as in [9], where a bloom filter computes j distinct packet digests for each marked packet using j independent uniform hash functions, and uses the n-bit result to index the 2^n-sized bit Digests Array (DA). The array is initialized to all zeros, and bits are set to one as packets are received.

The second requirement can be achieved by storing the marking information of a given packet in the Marking Information Table (MIT) at the memory location indexed by the first hash function that maps to zero bit in the digests array. Fig. 1 depicts both the DA and MIT with j hash functions. It also shows the marking information of a given packet before and after being marked. The marking fields reserved in each packet and in the MIT are shown also.

It can be realized that *probabilistic edge marking* (an edge is composed of two adjacent routers on the packet path) is simple to implement in our scheme. Whenever a router decides to mark a packet we enforce the subsequent router to mark the same packet. This can be achieved by maintaining a 1-bit field called *marking flag* as part of the marking information to be held in the packet. This flag is used to enforce deterministic marking when it is on. When it is off the marking becomes probabilistic. With this flag, the probabilistic edge marking in DLLT can be implemented as follows: When a router

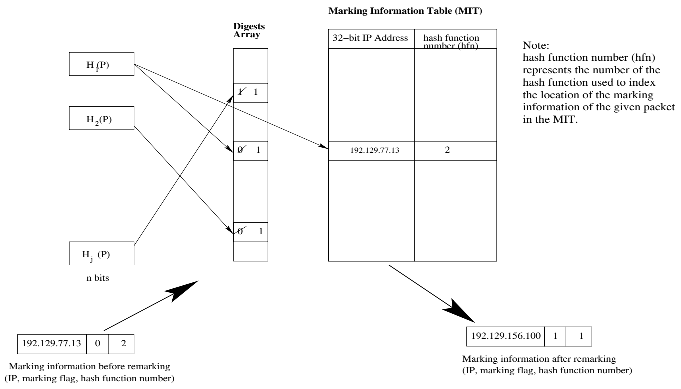
Marking Information Table (MIT)
Digests Array
32−bit IP Address
hash function number (hfn)
Note: hash function number (hfn) represents the number of the hash function used to index the location of the marking information of the given packet in the MIT.
H1(P)
H2(P)
Hj (P)
n bits
192.129.77.13
2
1
1
1
192.129.77.13 | 0 | 2
Marking information before remarking (IP, marking flag, hash function number)
192.129.156.100 | 1 | 1
Marking information after remarking (IP, marking flag, hash function number)

Fig. 1. Digests array (DA) and marking information table (MIT) at router R. The marking information of a given packet before and after being marked at router R which has the IP address of 192.129.156.100

receives a packet, it checks the marking flag. If it is on, it has to do the marking and storage procedure and then reset the flag. Otherwise (i.e., when the flag is off), it takes the decision based on some probability q. If the decision outcome is to mark the packet it will do so, and then set the flag such that the next adjacent router will do the marking deterministically.

After detecting an attack, the victim has to collect marking information that belongs to k of the received attack packets. This information can be retrieved from intermediate routers by following the link list associated with each of the chosen k packets. The attack sources are then determined by inferring the relative ordering of routers based on the retrieved marking information.

4 Analysis

Storage Analysis: The amount of storage that needs to be allocated to a traceback scheme is a critical issue. In this section, we quantify the amount of memory required in both DLLT and Hash-based scheme [9]. First, we review some characteristics of the bloom filters (or what we call digests arrays) that would be necessary in our analysis. A bloom filter is characterized by its size s bits, the number of hash functions used j, and its capacity factor f. A bloom filter of size s and capacity factor f can be used to store the digests of at most s/f packets. The effective false positive rate of a bloom filter is directly dependent on the previous parameters. Please refer to [1] and [9] for theoretical and experimental bounds on the false positive rate. What follows is a quantification of memory requirement at each router in both schemes.

Let b denotes the number of bits required to store the marking information of one packet in the MIT (i.e., this includes 32-bit IP address plus $\lceil \lg(j) \rceil$ bits for the hash function number). To store marking information of x packets, we need an MIT of size xb bits to be *shared* among f Digests Arrays each of size x bits. Therefore, the total memory requirement (M_{dllt}) to store x packets information is given by: $M_{dllt} = x(b+f)$.

Assuming an aggregate incoming link capacity of p packets/sec and marking probability of value q at each router, x can be replaced in the previous equation by qp. Therefore, the amount of memory required to store a second's worth of digests can be rewritten as: M_{dllt} = $qp(b + f)$. In Hash based scheme, To store digests of x packets, we just need a digests array of size fx bits. Meaning that the amount of memory required to store a second's worth of digests assuming an aggregate link capacity of p packets/sec can be expressed as: M_{hash}=fp. Expressing M_{dllt} as a function of q makes it adjustable to meet the limitations imposed by current memory technology.

Number of Attack Packets Required to Identify the Attacker: Our objective is to find a bound on the minimum number of packets that has to be received by the victim such that every router on the path from attacker to victim is involved in marking at least one of these packets with high confidence probability u. Let k represents this lower bound. Let the marking probability at router R be q. Let P_f be the probability that R fails to mark any packet out of the k packets. Clearly, $P_f = (1 - q)^k$. Therefore, the probability that R will succeed in marking (we call it the success probability) at least one packet is given by:

$$P_s = 1 - P_f = 1 - (1 - q)^k \tag{1}$$

To obtain the desired bound we can safely assume that the success probability for all routers a long a path of length l is the same and equal to that of the farthest router (i.e., as given in equation 1)

If we define X to be a random variable that represents the number of routers out of l that were successful in the marking process (l is path length), then X follows the binomial distribution with success probability P_s given in equation 1. We need to find k such that:$P(X = l) \geq u$ (i.e., the probability that each router succeeds in marking at least one packet is larger than u. But, $P(X = l)$ = $(1 - (1 - q)^k)^l \geq u$, solving for k, we obtain:

$$k \geq \frac{\lg(1 - u^{1/l})}{\lg(1 - q)} \tag{2}$$

For example, for an attack path of length 15 and marking probability of 0.3 and confidence probability 0.95 the number of packets required by DLLT is 16 compared to 1340 in PPM [6]. This significant reduction is due to the mechanism employed by DLLT to maintain marking information of routers far away from the victim.

5 Simulation Studies

We have carried out several simulation experiments to evaluate the proposed scheme. *Detection Percentage (DP)* defined as the percentage of exactly detected attack sources, is the metric used to evaluate our scheme. For example, if a attack sources are exactly located out of m attack sources, then we express the detection percentage as $\frac{a \times 100}{m}$ %.

In each simulation experiment we generated a random attack tree with m attackers and one victim. The attack path length l was the same for all attackers. Packets were marked according to a specific probability q. Attackers were instructed to inject their packets simultaneously with a rate of 1000 packets/attacker. k attack packets were used to conduct the traceback process. The default values for m, q, l, and k were set to be 100, 0.15, 20, and 1000 respectively. Each of the following results represents the average of 500 independent simulation runs based on the default parameter values unless otherwise specified. The simulation experiments were designed to study the effect of the above parameters on detection percentage. The results are shown in Fig. 2.

We measured the detection percentage of DLLT under different circumstances. Fig. 2 shows the detection percentage as a function of the marking probability q for different values of k, l, and m, respectively. In all these experiments we used the default parameter values mentioned above except for the parameter under investigation which made equal to the values shown in the figure. From theses figures we can make the following observations:

In all cases, the detection percentage of DLLT increases by increasing the marking probability. This is expected since marking routers would have better chance to appear in the information collected by the victim. Therefore, attack source identification becomes more accurate by increasing q. However, we should not forget the effect of increasing q on the amount of storage required by DLLT, and we should limit the marking probability to low values. Therefore, higher number of packets must be used to initiate marking information retrieval.

Increasing the number of packets used by the victim to identify attack sources results in better detection percentage. This can be observed in Fig.2 (left). As can be seen in Fig. 2 (middle), the attack path length seems to have negligible effect on the detection percentage. This can be explained by recalling that in short attack paths there is a low chance for any of the routers to mark a given packet, while in long attack paths there is a low chance for most of the routers to mark the given packet.

Fig. 2 (right) depicts the effect of increasing the number of attackers m while fixing the number of attack packets used by the victim k. It is clear that the detection percentage is affected negatively by this increase. In fact, the detection percentage in this case can not be increased without increasing the number of packets k used by the victim.

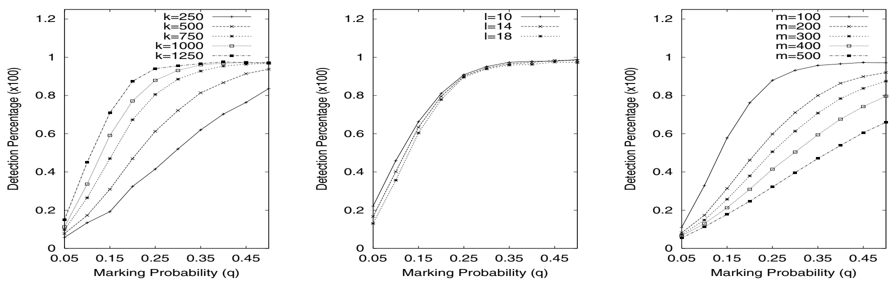

Fig. 2. Left: Effect of number of attack packets k used by the victim on the detection percentage. Middle: Effect of attack path length l on the detection percentage. Right: Effect of number of attackers m on the detection percentage.

6 Conclusion

An efficient traceback scheme is necessary to identify the sources of denial of service attacks which impose an imminent threat to the availability of Internet services. In this paper we proposed an efficient traceback scheme called DLLT. In this scheme, the probabilistic nature of marking and storage offers the advantage of minimizing router and storage overhead. Also, storing the packet digests at intermediate routers provides an authentic way to verify that a given router has actually forwarded certain packet. This prevents attackers from passing spoofed marking information to the victim even if the marking probability is very low. DLLT employs an efficient scheme to collect marking information from intermediate routers. Moreover, we showed that the number of packets required to identify the attack sources is low. Simulation studies show that DLLT offers high attack source detection percentage.

References

1. B. H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” in *Communications of ACM 13*, July 1970, 422-426.
2. H. Burch and B. Cheswick, “Tracing anonymous packets to their approximate source,” in *Proc. 2000 USENIX LISA Conf.,* Dec. 2000, pp.319-327.
3. D. Dean, M. Franklin, and A. Stubblefield, “An algebraic approach to IP traceback,” in *Network and Distributed System Security Symposium (NDSS ’01),* Feb. 2001.
4. M. T. Goodrich, “Efficient Packet Marking for Large-Scale IP Traceback,” in *Proc. of ACM CCS 2002*, Nov. 2002.
5. K. Park and H. Lee, “On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack,” in *Proc. of IEEE INFOCOM 2001*, Mar. 2001.
6. S. Savage, D. Wetherall, A. Karlin and T. Anderson, “Practical network support for IP traceback,” in *Proc. of ACM SIGCOMM*, Aug. 2000, pp. 295-306.
7. D. Song and A. Perrig, “Advanced and authenticated marking schemes for IP traceback,” in *Proc. of IEEE INFOCOMM 2001,* April 2001.
8. R. Stone, “Centertrack: An IP overlay network for tracking DoS floods,” in *Proc. of* 9_{th} *USENIX Security Symposium*, Aug. 2000.
9. A. C. Snoeren, C. Partiridge, L. A. Sanchez, C. E. Jones, F. Tchhakountio, S. T. Kent, and W. T. Strayer, ”Hash-Based IP TraceBack,” in *Proc. of ACM SIGCOMM*, Aug. 2001.