A Novel Hybrid IP Traceback Scheme with Packet Counters

Tomoyuki Karasawa[1], Masakazu Soshi[2], and Atsuko Miyaji[3]

[1] Internet Initiative Japan (IIJ) Inc.
tomoyuki-k@iij.ad.jp
[2] Hiroshima City University
soshi@hiroshima-cu.ac.jp
[3] Japan Advanced Institute of Science and Technology (JAIST)
miyaji@jaist.ac.jp

Abstract. In this paper we shall propose a novel hybrid IP traceback scheme with *packet counters*. In our scheme, a (packet) counter is used to improve correlation of packet sampling in order to reconstruct the attack tree efficiently. Our scheme has the remarkable advantages: (1) it is simple and efficient, (2) it is significantly resistant to attacks, (3) it requires a lower sampling rate compared with previous work, e.g., only 1% is enough, (4) its false positive/negative rates are also lower.

1 Introduction

One of the most serious threats to the Internet security is a *DoS (Denial of Service) attack*, where an attacker attempts to make a target host (called a *victim*) fail by sending a huge number of packets to the host [1]. In particular, in recent years, a *DDoS (Distributed DoS) attack*, where there are many attackers scattered over the Internet, has become more prevailing [1]. Such a DDoS attack can be represented by an *attack tree*, the leaves and the root of which are the attackers and the victim, respectively. Furthermore, we call a path along which an attack packet traverses from one attacker to the victim an *attack path*.

A promising countermeasure against DoS/DDoS attacks is called *IP traceback* [2–8]. In IP traceback schemes, each router on attack paths stores information about the paths on itself or on packets. Then the victim uses the information to recover the attack tree and to find out the attackers.

IP traceback schemes are roughly classified two-fold: *probabilistic packet marking* (PPM for short) protocols and *logging* ones. In PPM protocols, each router probabilistically writes path information onto the packets it receives [3, 6]. On the other hand, logging IP traceback protocols make each participating router sample packets and store path information on itself [4, 7]. PPM and logging protocols have some advantages, although, they have serious disadvantages (discussed in Sect. 2). Therefore to take advantages of PPM and logging approaches, *hybrid IP traceback schemes* have attracted much attention these years [2, 5].

In this paper we shall propose a new hybrid IP traceback scheme. A novel idea of our scheme is the introduction of a *counter* on each packet header. Such a counter is used to improve correlation of packet sampling in order to reconstruct the attack tree efficiently. Our scheme has the following remarkable advantages: (1) it is simple and efficient, (2) it is highly resistant to attacks, (3) it requires a lower sampling rate compared with previous work, e.g., only 1% is enough, (4) its false positive/negative rates are lower than those of Li et al. [5]. To show these advantages theoretically, we conduct information theoretical analysis of our scheme in detail. Furthermore, we make simulation experiments to investigate the performance of our scheme in practical environments. These theoretical and practical evaluations of our scheme show that our scheme is truly effective.

This paper is organized as follows. We discuss related work in Sect. 2 and shall propose a novel IP traceback in Sect. 3. Then we thoroughly evaluate our scheme in Sect. 4 and Sect. 5. Finally we give conclusion in Sect. 6.

2 Related Work

As stated in Sect. 1, IP traceback schemes are roughly classified into *probabilistic packet marking* (PPM) protocols [3, 6] and *logging* ones [4, 7]. PPM does not need storage resource of routers, although, it generally requires the victim to receive a large number of packets before he can reconstruct the attack tree. On the other hand, in logging schemes, the number of packets for attack tree recovery can be small. However, logging schemes impose heavy load and require extremely large storage space on the routers.

Now, for a recent example of IP traceback protocols, let us consider the work by Yu et al. [8]. The protocol exploits entropy variation for IP traceback and is very interesting itself. However, the proposed flow monitoring algorithm and IP traceback algorithm are rather intricate. Furthermore, the false positive/negative rates in the traceback process are not discussed in detail in [8].

In summary, we still do not have an established IP traceback scheme. However, we consider *hybrid IP traceback schemes* to be promising because they can take advantage of both PPM and logging approaches [2, 5]. In particular, Li et al. proposed one of the most important hybrid IP traceback schemes [5]. The protocol of Li et al. was successful in improving sampling correlation. Unfortunately, they consider only the correlation between neighboring routers. In this paper, we shall show that we can develop a highly efficient IP traceback scheme *by considering correlation of packet sampling all over a whole attack path.*

3 Our Proposed Protocol

In this section we propose a novel hybrid IP traceback scheme.

3.1 Basic Idea

First of all, we briefly give a basic idea of our proposed scheme. Our scheme attempts to improve correlation of packet sampling over a whole attack path for efficient recovery of the attack tree. For that purpose, we designate five bit space on each packet header as a *counter*[4]. Throughout the paper, we denote the counter of packet P by $P.counter$.

In our scheme, if a router on an attack path decides to sample a packet P with some probability[5], then it increments P.counter by one, stores some information on itself, and finally passes the packet to an adjacent router. Consequently, *the greater the value of $P.counter$ is, the larger the number of routers on the attack path that store information of P is.* In other words, a packet with a large counter value is *useful* when we recover the attack tree. Therefore, when we sample a packet, if we *probabilistically* prefer a packet with a larger counter value to one with a smaller value, then we can improve the correlation factor of packet sampling and have more useful packets for traceback later.

3.2 Sampling

Based on the discussion in Sect. 3.1, in this section we shall propose a novel sampling algorithm as in Fig. 1. Each participating router R executes the sampling algorithm when it receives a packet P. It is really interesting that such a simple sampling algorithm as ours is highly effective, as shown later in this paper.

```
Sampling procedure at router $R$:
for each packet $P$
    $x$ is chosen uniformly at random between 0 and 1
    $p \leftarrow$ compute_prob($P$.counter)      ▷ 'compute_prob' is discussed later
    if ($x < p$) then                           ▷ i.e, with probability $p$
        $P$.counter $\leftarrow$ $P$.counter $+1$
        Store digest of $P$                      ▷ discussed in Sect. 3.3
```

Fig. 1. Our Proposed Sampling Algorithm

Suppose that R receives a packet P. Then as depicted in Fig. 1, with a probability p, which is computed by the procedure 'compute_prob', R increments P.counter by one and stores the digest of P (see also Sect. 3.3).

Next let us take a closer look at 'compute_prob' itself. As discussed in Sect. 3.1, we should preferentially sample a packet with a larger counter value. Furthermore, in order to reduce load on routers, we must make 'compute_prob' as simple as possible. In addition, it must be efficiently computable.

Therefore, in this paper we propose to implement 'compute_prob' as in Fig. 2, where $\alpha > 1$, β, and M are some constant values, which are discussed in Sect. 4.3.

As we can obviously see from Fig. 2, a sampling probability returned by 'compute_prob' becomes larger in a polynomial order of a counter value c. Note

[4] Dean et al. [3] have pointed out that 25 bits in each IPv4 packet header are available for IP traceback. So five bits can easily be accommodated by each packet header.

[5] How to compute this probability is vital to our scheme and is discussed in Sect. 3.2.

```
function compute_prob($c$)                ▷ $c$ is a counter value
    return $(c^\alpha + \beta)/M$            ▷ return the sampling probability $(c^\alpha + \beta)/M$
```

Fig. 2. Our Proposed Function for 'compute_prob'

that since we assume that $\alpha > 1$, the probability grows significantly larger than the case proportional to c. Intuitively speaking, it means that *in a polynomial order of counter values, a packet with a larger counter is sampled with a higher probability, but a packet with a smaller counter is sampled with a lower probability.* In this way we give a preference to a packet with a larger counter value when sampling attack packets.

3.3 Storing packet digests

In our sampling algorithm in Fig. 1, we compute the digest of a packet in the form of Bloom filter [9]. For a set S of packets, Bloom filter is represented by an array A of m bits, each of which is expressed as $A[i] \in \{0, 1\}$ and initially every $A[i]$ is set to zero $(i = 1, ..., m)$. Moreover, we assume that Bloom filter in this paper uses k hash functions $h_1, ..., h_k$ that are independently and randomly chosen. Every hash function h_i $(i = 1, ..., k)$ has the range $\{1, ..., m\}$.

Packet digesting is carried out as follows. When we insert the digest of packet P into S, we set $A[h_i(P)]$ to one $(i = 1, ..., k)$, where $h_i(P)$ is the output of h_i when it takes as input the first invariable 28 bits of P as in [7]. Now when we want to know if a packet P' is a member of S or not, we check the bits $A[h_i(P')]$ $(i = 1, ..., k)$. If any of the bits is zero, then P' is not in S. Otherwise, i.e., if every bit is one, P' is in S with a *high probability.* That is, even in such a case, it *is* possible that P' is actually not in S.

In summary, Bloom filter cannot exhibit *false negatives.* Therefore, when P is actually in S, the Bloom filter does never report that P is not in S. On the other hand, every Bloom filter can have *false positives.* In other words, it can conclude (with a very low probability) that $P \in S$, in spite that actually $P \notin S$. Throughout the paper, we estimate that the probability of false positives in our Bloom filter is given by 2^{-k} as in [5, 9].

3.4 Traceback

Our scheme traces back the attack sources as follows. Once faced with DoS/DDoS attacks, with a some predefined threshold T, the victim adds each packet P from a set $\mathcal{N}$ of its received packets to a set $\mathcal{N}'$ of packets if P.counter $\geq T$. Note that from the discussions from Sect. 3.1, we see that if the victim sets T to a large value, then he can use packets with higher sampling correlation and it leads to efficient recovery of the attack tree. On the other hand, if he makes T too large, then the number of the attack packets available in performing traceback becomes smaller. Taking into consideration the discussion, the victim can choose any T that is appropriate for his environment.

After constructing a set $\mathcal{N}'$ of packets, the victim sends $\mathcal{N}'$ to every neighboring router R. Next if R finds out that some packet in $\mathcal{N}'$ is also in its sampling log (i.e., in the form of Bloom filter), then R supposes that it is on an attack path and sends $\mathcal{N}'$ to every router R' that is adjacent to R. Such a traceback process is repeatedly performed until it cannot go any further.

3.5 Security Consideration

In this section we briefly discuss security of our scheme. First note that all attackers are supposed to be located at the leaves of an attack tree. Therefore, an attacker could affect security of our scheme *only by forging packet counters*. More specifically, all that the attacker can do is only to set to some value the counter of a packet that he injects into the network. However, such a forge by the attacker cannot be a threat to our scheme for the following reason. Namely, a large value of a packet counter given by the attacker only leads to an increase of the sampling probability of the packet, which in turn ends with increasing the correlation of packet sampling and then finally with a more efficient traceback process. Therefore, such an “attack” can *never* be an attack in a true sense, but rather it leads the victim to a situation that is more beneficial to him.

Thus a possible attack that attackers at the leaves can make would be to always set the lowest values to packet counters, that is, to always initialize the counters to zero. This is indeed the worst case scenario to our protocol because in such a case correlation of packet sampling would be forced to be as low as possible and then traceback processes later would be made more difficult.

Fortunately, again, such an attack above cannot be effective. As shown in Sect. 4.3, even if packet counters are initially zero, our scheme exhibits excellent performance (for instance, an efficient traceback process, low false positive/negative rates, and so on).

In summary, we can conclude that our scheme is highly resistant to attacks.

4 Information Theoretical Analysis

In this section, we conduct theoretical analysis of our proposed scheme based on information theory[6].

Now, let us consider the attack path from an attacker to the victim. Henceforth in the paper R_1 denotes the nearest router to the attacker on the attack path and R_2 the second nearest router, which is adjacent to R_1. We denote by R_3, …, R_{n-1} and R_n the remaining routers on the path in order. Thus R_n is the neighboring router to the victim and n is the (maximum) number of routers between the attacker and the victim.

[6] Notice that although the analysis in this section is done in a similar manner to [5] at a glance, the former is more involved than the latter because we must take into account the probabilistic behavior of packet counters.

4.1 Formalization of Sampling

In this section we analyze the sampling procedure of our scheme. Let X_{c_i} be a random variable that represents the counter value of a packet P when P arrives at router R_i. Then, in order to be fair, we shall evaluate our scheme in the worst case scenario to it. That is, as discussed in Sect. 3.5, the case where the counter value of every packet that R_1 receives is zero. Therefore we have

$$\Pr(X_{c_1} = c) = \begin{cases} 1 & \text{if } c = 0 \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

and

$$\Pr(X_{c_2} = c) = \begin{cases} \beta/M & \text{if } c = 1 \\ 1 - \beta/M & \text{if } c = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Generally, we have

$$\begin{aligned} \Pr(X_{c_i} = c) &= \Pr(X_{c_i} = c \mid X_{c_{i-1}} = c-1) \cdot \Pr(X_{c_{i-1}} = c-1) \\ &\quad + \Pr(X_{c_i} = c \mid X_{c_{i-1}} = c) \cdot \Pr(X_{c_{i-1}} = c) \\ &= \frac{(c-1)^\alpha + \beta}{M} \cdot \Pr(X_{c_{i-1}} = c-1) + \left(1 - \frac{c^\alpha + \beta}{M}\right) \cdot \Pr(X_{c_{i-1}} = c) \ . \end{aligned} \tag{3}$$

Note that the counter values of packets R_i receives are non-negative and the (possible) maximum of the values is $i-1$. Thus if $c < 0$ or $i - 1 < c$, then it holds that $\Pr(X_{c_i} = c) = 0$.

Next in regard to router R_i and a packet P, we define a random variable (indicator variable [9]) X_{p_i} as below:

$$X_{p_i} = \begin{cases} 1 & \text{if } R_i \text{ samples } P \\ 0 & \text{otherwise.} \end{cases}$$

The probability distribution of X_{p_i} can be obtained as follows by using random variable X_{c_i} (Eq. (3)) and the relationship $\Pr(X_{p_i} = 1 \mid X_{c_i} = c) = (c^\alpha + \beta)/M$:

$$\Pr(X_{p_i} = 1) = \sum_{c=0}^{i-1} \frac{c^\alpha + \beta}{M} \cdot \Pr(X_{c_i} = c) \ . \tag{4}$$

4.2 Evaluation of Traceback

In this section we evaluate the traceback procedure in our proposed scheme using information theory.

Suppose that we are about tracing router R_{i-1} back from router R_i by one hop. As stated in Sect. 3.4, router R_{i-1} is considered on an attack path if the number of the packets sampled by both of R_{i-1} and R_i is greater than or equal to the prespecified threshold T. In the subsequent sections, we assume that T is one for brevity.

The Model Let $\mathcal{N}$ be a set of the attack packets that the victim uses for traceback and N_p the number of the elements in $\mathcal{N}$, i.e., $N_p = |\mathcal{N}|$. Furthermore, as in [5], we set d_i to the percentage of attack packets that traverse through R_i. Although the value of d_i varies dependently on i, in this paper for simplicity we assume that d_i equals some d for all i. Moreover, the digest of each packet is stored in the form of Bloom filters and our Bloom filter is supposed to use k independent hash functions. We assume that the probability of false positives of the Bloom filter is $f = 2^{-k}$ (see Sect. 3.3). In addition, the binomial distribution with n experiments and success probability p is denoted by $\mathrm{Binom}(n, p)$.

Next we introduce some random variables as defined below.

- X_{t_i}: the number of attack packets that R_i samples.
- X_{f_i}: the number of false positives when querying each packet in $\mathcal{N}$ to Bloom filter of R_i. X_{f_i} follows the binomial distribution $\mathrm{Binom}(N_p - X_{t_i}, f)$.
- Y_{t_i}: the number of attack packets with which we trace back from R_i to R_{i-1}. Namely, the number of attack packets sampled at both of R_i and R_{i-1}.
- Y_{f_i}: the number of false positives when querying $X_{t_i} + X_{f_i}$ to Bloom filter at R_{i-1}. Y_{f_i} has a probability distribution $\mathrm{Binom}(X_{t_i} + X_{f_i} - Y_{t_i}, f)$.
- $X_i = X_{t_i} + X_{f_i}$: the number of attack packets used for a traceback process for R_{i-1}.
- $Y_i = Y_{t_i} + Y_{f_i}$: i.e., the number of attack packets both in $\mathcal{N}$ and in the Bloom filter of R_{i-1}.

Now we can define random variable Z_i, which indicates a situation where at least one of attack packets used by R_i for traceback is also sampled by R_{i-1}:

$$Z_i = \begin{cases} 1 & \text{if } X_{t_{i-1}} > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Attack Packets Since we can say that X_{t_i} follows the binomial distribution $\mathrm{Binom}(N_p d_i,\ \Pr(X_{p_i} = 1))$, we obtain

$$\Pr(X_{t_i} = j) = \binom{N_p d_i}{j} \cdot \Pr(X_{p_i} = 1)^j \cdot (1 - \Pr(X_{p_i} = 1))^{N_p d_i - j} \ . \tag{6}$$

The victim traces back from R_i to R_{i-1} by using the set $\mathcal{N}$ of attack packets. Because X_{f_i} represents the number of false positives of the Bloom filter of R_i, which occurs in querying $\mathcal{N}$ to the filter, we get

$$\Pr(X_{f_i} = \ell) = \sum_{j=0}^{N_p d_i} \Pr(X_{t_i} = j) \binom{N_p - j}{\ell} f^\ell (1 - f)^{N_p - j - \ell} \ . \tag{7}$$

Moreover, since $X_i = X_{t_i} + X_{f_i}$, Eqs. (6) and (7) yield the probability distribution of X_i as

$$\Pr(X_i = j) = \sum_{\ell=0}^{\min(j, N_p d_i)} \Pr(X_{t_i} = \ell) \cdot \Pr(X_{f_i} = j - \ell) \ . \tag{8}$$

Next we consider Y_{t_i}. If the counter value of a packet that R_{i-1} receives is c, then the probability that both R_{i-1} and R_i samples the packet is given by

$$\Pr(X_{p_i} = 1, X_{p_{i-1}} = 1 \mid X_{c_{i-1}} = c) = \frac{(c+1)^\alpha + \beta}{M} \cdot \frac{c^\alpha + \beta}{M} \ ,$$

which in turn yields

$$\Pr(X_{p_i} = 1, X_{p_{i-1}} = 1) = \sum_{c=0}^{i-2} \frac{(c+1)^\alpha + \beta}{M} \cdot \frac{c^\alpha + \beta}{M} \cdot \Pr(X_{c_{i-1}} = c) \ . \tag{9}$$

Now from Eqs. (3), (4), and (9), we can easily compute $\Pr(X_{p_i} = 1 \mid X_{p_{i-1}} = 1)$. Furthermore, we can consider that Y_{t_i} follows $\mathrm{Binom}(X_{t_{i-1}}, \Pr(X_{p_i} = 1 \mid X_{p_{i-1}} = 1))$ and its probability distribution can also be easily calculated.

Conditional Entropy of Z_i With the random variables in Sect. 4.2 and from the definition of conditional entropy, we can compute $H(Z_i \mid X_i, Y_i)$ (for the definition, see also [5]). Notice that the smaller $H(Z_i \mid X_i, Y_i)$ is, the higher the success probability of a traceback process. Hence in order to evaluate our proposed scheme, we need to compute $H(Z_i \mid X_i, Y_i)$. For the purpose, remember:

$$\Pr(X_i = j, Y_i = m, Z_i = a) = \Pr(X_i = j, Y_i = m \mid Z_i = a) \cdot \Pr(Z_i = a) \ . \tag{10}$$

With respect to Z_i, as in [5] we assume

$$\Pr(Z_i = 0) = \Pr(Z_i = 1) = 1/2 \ . \tag{11}$$

Below we compute $H(Z_i \mid X_i, Y_i)$ in the cases that (c1) $Z_i = 1$ and (c2) $Z_i = 0$ respectively.

Case (c1): $Z_i = 1$. In this case, we can rewrite the first part of the right side of Eq. (10) to

$$\Pr(X_i = j, Y_i = m \mid Z_i = 1) = \Pr(X_i = j \mid Z_i = 1) \cdot \Pr(Y_i = m \mid X_i = j, Z_i = 1) \ . \tag{12}$$

Then remembering that $Y_i = Y_{t_i} + Y_{f_i}$, in regard to the second part of the right side of Eq. (12), we in turn have

$$\begin{aligned} &\Pr(Y_{t_i} + Y_{f_i} = m \mid X_i = j, Z_i = 1) \\ &= \sum_{r=0}^{\min(m, N_p d_i)} \Pr(Y_{t_i} = r \mid X_i = j, Z_i = 1) \\ &\qquad \times \Pr(Y_{f_i} = m - r \mid X_i = j, Y_{t_i} = r, Z_i = 1) \ . \end{aligned} \tag{13}$$

For a part of the right hand side of Eq. (13), we can obtain

$$\Pr(Y_{f_i} = m - r \mid X_i = j, Y_{t_i} = r, Z_i = 1) = \binom{j-r}{m-r} f^{m-r} (1-f)^{j-m} \ . \tag{14}$$

Then, in order to compute Eq. (13), we need the probability:

$$\Pr(Y_{t_i} = r \mid X_i = j, Z_i = 1) \ . \tag{15}$$

So first we introduce random variable W_i, which satisfies $X_{t_i} = Y_{t_i} + W_i$. Remember that Y_{t_i} is the number of the attack packets sampled by both R_{i-1} and R_i. Therefore W_i is nothing but the number of the attack packets which are sampled by R_i, but not by R_{i-1}. This means that W_i follows $\text{Binom}(N_p d_i - X_{t_{i-1}}, \Pr(X_{p_i} = 1 \mid X_{p_{i-1}} = 0))$. Here note that $\Pr(X_{p_i} = 1, X_{p_{i-1}} = 0 \mid X_{c_{i-1}} = c)$ can be easily calculated as $\frac{c^\alpha+\beta}{M} \cdot \left(1 - \frac{c^\alpha+\beta}{M}\right)$. Therefore we have

$$\begin{aligned}\Pr(X_{p_i} = 1, X_{p_{i-1}} = 0) &= \sum_{c=0}^{i-2} \Pr(X_{p_i} = 1, X_{p_{i-1}} = 0 \mid X_{c_{i-1}} = c) \cdot \Pr(X_{c_{i-1}} = c) \\ &= \sum_{c=0}^{i-2} \frac{c^\alpha + \beta}{M} \cdot \left(1 - \frac{c^\alpha + \beta}{M}\right) \cdot \Pr(X_{c_{i-1}} = c) \ . \end{aligned} \tag{16}$$

Consequently, from Eqs. (3), (4), and (16) we can obtain $\Pr(X_{p_i} = 1 \mid X_{p_{i-1}} = 0))$ and in turn the probability distribution of W_i.

We are now in a position to compute Eq. (15). First note that

$$\begin{aligned}&\Pr(Y_{t_i} = r \mid X_i = j, Z_i = 1) \\ &= \sum_{\ell=0}^{j} \Pr(X_{f_i} = \ell) \cdot \Pr(Y_{t_i} = r \mid X_{t_i} = j - \ell, X_{f_i} = \ell, Z_i = 1) \ .\end{aligned}$$

Therefore from Eq. (6), the binomial distributions of Y_{t_i}, and W_i, as the analysis in [5], we can compute Eq. (15) (the detail is omitted due to limited space).

Putting the discussions above together, with Eqs. (14) and (15), we can evaluate Eq. (13). Then from the Eqs. (8) and (13), we are now able to compute Eq. (12).

Finally, from Eqs. (10), (11), and (12), we can calculate:

$$\Pr(X_i = j, Y_i = m, Z_i = 1) \ . \tag{17}$$

Case (c2): $Z_i = 0$. In this case, it is easy to see that

$$\Pr(X_i = j, Y_i = m \mid Z_i = 0) = \Pr(X_i = j) \binom{j}{m} f^m (1 - f)^{j-m} \ . \tag{18}$$

Therefore from Eqs. (8), (11) and (18), we can compute:

$$\Pr(X_i = j, Y_i = m, Z_i = 0) \ . \tag{19}$$

Computation of $H(Z_i \mid X_i, Y_i)$. $\Pr(X_i = j, Y_i = m)$ can be obtained as follows:

$$\begin{aligned}\Pr(X_i = j, Y_i = m) = &\Pr(X_{t_{i-1}} = 0) \cdot \Pr(X_i = j, Y_i = m \mid Z_i = 0) \\ &+ \Pr(X_{t_{i-1}} > 0) \cdot \Pr(X_i = j, Y_i = m \mid Z_i = 1) \ . \end{aligned} \tag{20}$$

Thus from Eqs. (6), (12) and (18), Eq. (20) can be computed.

We have so far obtained Eqs. (17), (19) and (20). Therefore from the definition of conditional entropy, we can compute $H(Z_i \mid X_i, Y_i)$, which is the goal of this section.

4.3 Numerical Computation of Theoretical Results

In this section we shall conduct a numerical analysis of our proposed scheme in order to evaluate its performance and behavior in a more concrete manner.

How to determine $\boldsymbol{\alpha}, \boldsymbol{\beta}$ and $\boldsymbol{M}$ As discussed in Sect. 3.2, when the router receives a packet with counter value c, it holds that

$$p = (c^{\alpha} + \beta)/M \ . \tag{21}$$

Here we discuss how to determine the values α, β and M.

First, regarding a packet P, we consider the maximum and minimum value of P.counter. Since by assumption there are at most n routers on any attack paths and initial values of packet counters are zero as discussed in Sect. 4.1, the maximum value of the counters is $n - 1$. In most literature on IP traceback schemes, the maximum length of attack paths is supposed to be less than or equal to 16 (for example, see [6]) and therefore we also suppose that the maximum counter value is 16, which means that $n = 17$. Note that as we will see below in detail, even if a hop count of some packet exceeds 16, it causes almost no problems in our scheme. From an implementation point of view, as discussed in Sect. 3.1, it is fairly easy to allocate 5 bit space in each header to the counter.

For each packet P, the sampling probability p is less than or equal to one when P.counter is maximized. Hence from Fig. 2, α, β and M must satisfy:

$$p = (16^{\alpha} + \beta)/M \leq 1 \ . \tag{22}$$

However, note that as discussed in Sect. 3.5 and later in this section, in general sampling probabilities in our scheme are far below one, namely, just 1% or so.

Next we consider the minimum value of a packet counter, i.e., zero. Let p_I be the sampling probability p when the counter c is zero, that is, $p_I = \beta/M$ (see Eq. (21)). In order to be fair when we evaluate performance of our scheme, we set p_I to a smaller value than, for example, the sampling probability of the scheme of Li et al. [5], i.e, 3.3%. More specifically, we set $p_I = \beta/M = 0.01$.

From the discussion above, in this paper we consider the following three cases to satisfy Eq. (22) and $p_I = 0.01$:

- $\alpha = 2, \beta = 2.58586, M = 258.586$,
- $\alpha = 3, \beta = 41.3737, M = 4137.37$, or
- $\alpha = 4, \beta = 661.98, M = 66198.0$.

Actually we investigated other settings than the above, although, as we show later, we have already obtained satisfactory results in the above three settings.

Probability Distributions of Counter Values and Sampling Probability First we show the probability distributions of the counter values of routers R_{16} (hop count 15) and just for reference, R_{32} (hop count 31) as depicted in Fig. 3 (a) and (b) respectively, according to Eq. (3). From Fig. 3 we see that almost all counter values are at most 2 with the parameters discussed in Sect. 4.3. Such a small counter value results in a small sampling probability discussed below.

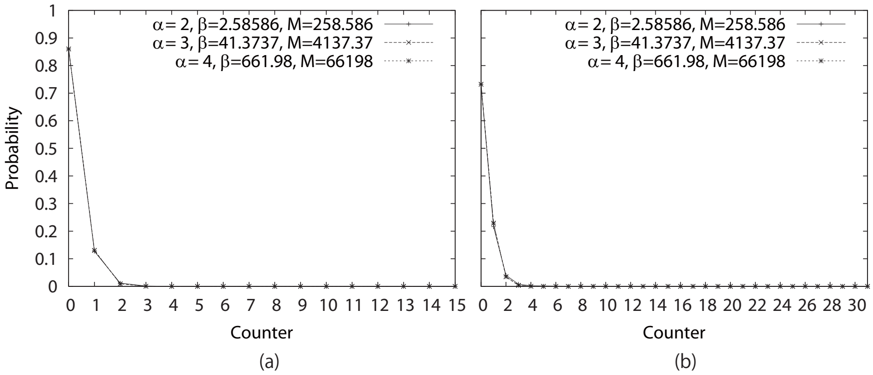

Fig. 3. Probability distributions of the counter values (Hop count = (a) 15 and (b) 31)

Now from Eqs. (3) and (4), we consider the average sampling probabilities, which are depicted in Fig. 4. Fig. 4 shows that the average sampling probabilities remain very small, that is, about 0.01, in all cases $\alpha = 2, 3, 4$. Note that the sampling probability 0.01 is so small. For example, the scheme in [5] requires the sampling probability 0.033, which proves how efficient our scheme is.

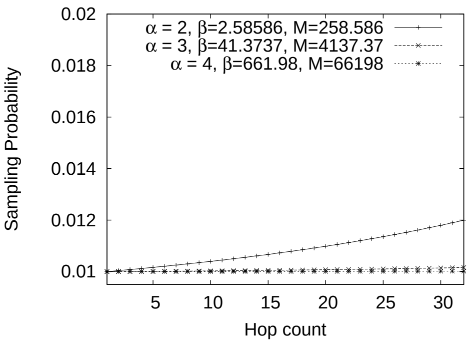

Fig. 4. Hop counts and average sampling probabilities

Evaluation of $H(Z_i \mid X_i, Y_i)$ Now we evaluate $H(Z_i \mid X_i, Y_i)$ with the parameters mentioned in Sect. 4.3 and under the conditions given below:

- $d = 0.1, N_p = 200$ (Fig. 5 (a)),
- $d = 0.2, N_p = 100$ (Fig. 5 (b)), or
- $d = 0.4, N_p = 50$ (Fig. 6).

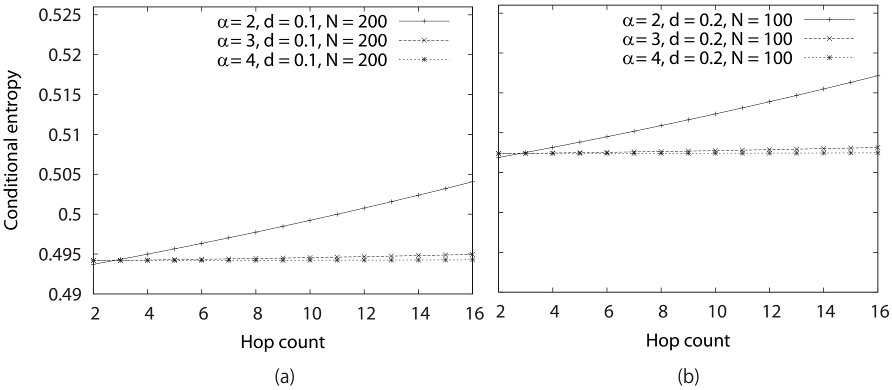

Fig. 5. Estimation of $H(Z_i \mid X_i, Y_i)$: (a) $d = 0.1, N_p = 200$, (b) $d = 0.2, N_p = 100$

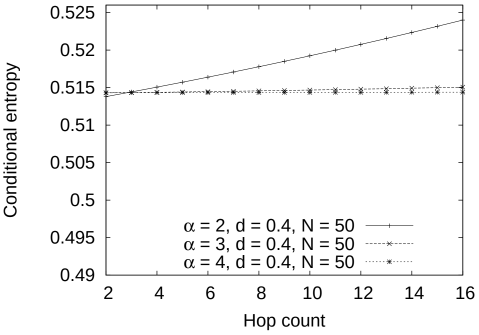

Fig. 6. Estimation of $H(Z_i \mid X_i, Y_i)$ $(d = 0.4, N_p = 50)$

That is, we set the above parameters such that the number of packets to be sampled are the same by letting $N_p d$ be 20. Remember that the smaller the value of conditional entropy $H(Z_i \mid X_i, Y_i)$ is, the smaller the number of attack packets required for traceback is.

When we compare our scheme with other work, we try to be as fair as possible as follows. First, we set k, which is the number of independent hash functions used for Bloom filters in our scheme, to 12 that is the optimum number for the scheme of Li et al. [5]. Second, we set the sampling probability p to (almost) 0.01 and the number of packets used for traceback to $N_p d = 20$. It is clear that such a setting is disadvantageous to our scheme, with comparison to the settings of Li et al. [5], where $p = 0.03$ and $N_p d = 50$. In such a setting, the optimum value of the conditional entropy of Li's scheme is about 0.63 as shown in [5].

In the above settings, in our scheme $H(Z_i \mid X_i, Y_i)$ is at worst about 0.525 as shown in Fig. 6, and is about 0.495 at best as in Fig. 5 (a). The comparison clearly shows that our scheme is superior to that of Li et al.

5 Simulation Experiments

In this section, in order to evaluate our scheme in a practical environment, we conduct simulation analysis of it. In the simulation, we utilize real world Internet topological data by Skitter project [10], which are also used in [5]. The topological data are given as follows:

- a-root: this map includes the Internet topological data on Nov. 25, 2001 from `a-root.skitter.caida.org` to 192,900 destinations, and
- e-root: the Internet topological data on Nov. 28, 2001 from `e-root.skitter.caida.org` to 158,181 destinations.

In those simulation experiments, we let the number of attackers be 1,000, and the number of attack packets N_p be 50,000. We randomly generate attack trees 1,000 times based on the above topological data and place attackers at the leaves of the trees. The lengths of the attack paths are supposed to be greater than or equal to 16 at random.

Now we give the simulation results in Figs. 7 (a) a-root and (b) e-root. In order to compare our scheme with the scheme of Li et al. [5], the relationship between the number of hash function k and the error level are given in the figures. Here we mean by 'error level' the sum of false negative ratio (FNR) and false positive ratio (FPR), where FNR represents the ratio of the number of routers which are not in the reconstructed tree but actually in the real attack tree to the number of the routers in the reconstructed tree. Similarly, FPR means the ratio of the number of routers which are in the reconstructed tree but actually not in the real attack tree to the number of the routers in the reconstructed tree. As we can obviously see from the results in Fig. 7, our scheme exhibits remarkably low error levels in all settings given above. The remarkable result would be mainly due to the low entropy value of $H(Z_i \mid X_i, Y_i)$, as discussed in Sect. 4.3.

6 Conclusion

In this paper we proposed a novel hybrid IP traceback scheme by using packet counters. Our scheme has the several remarkable advantages, that is, (1) it is simple and efficient, (2) it is highly resistant to attacks, (3) it requires a lower sampling rate compared with previous work, e.g., only 1% is enough, (4) its false positive/negative rates are lower than previous work.

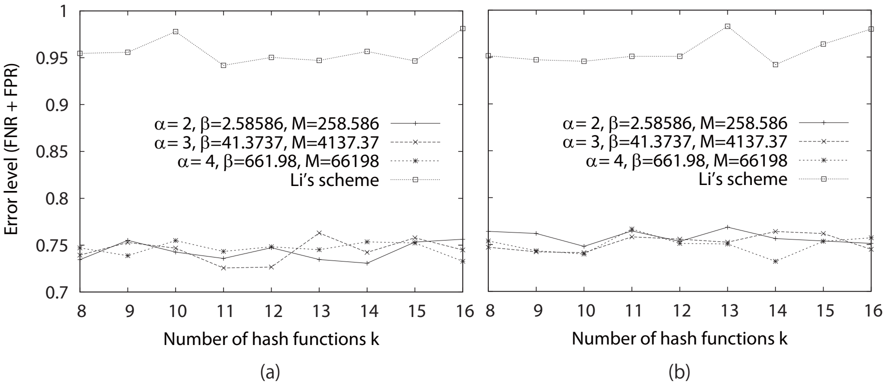

Fig. 7. Simulation results

References

1. Peng, T., Leckie, C., Ramamohanarao, K.: Survey of network-based defense mechanisms countering the DoS and DDoS problems. ACM Computing Surveys **39**(1) (April 2007)
2. Al-Duwairi, B., Manimaran, G.: Novel hybrid schemes employing packet marking and logging for IP traceback. IEEE Transactions on Parallel and Distributed Systems **17**(5) (May 2006) 403–418
3. Dean, D., Franklin, M., Stubblefield, A.: An algebraic approach to IP traceback. ACM Transactions on Information and System Security **5**(2) (May 2002) 119–137
4. Gong, C., Sarac, K.: Toward a practical packet marking approach for IP traceback. International Journal of Network Security **8**(3) (May 2009) 271–281
5. Li, J., Sung, M., Xu, J., Li, L.: Large-scale IP traceback in high-speed Internet: Practical techniques and theoretical foundation. In: Proceedings of the IEEE Symposium on Security and Privacy. (May 2004) 115–129
6. Savage, S., Wetherall, D., Karlin, A.R., Anderson, T.: Practical network support for IP traceback. In: Proceedings of the ACM SIGCOMM. (2000) 295–306
7. Snoeren, A.C., Partridge, C., Sanchez, L.A., Jones, C.E., Tchakountio, F., Kent, S.T., Strayer, W.T.: Hash-based IP traceback. In: Proceedings of the ACM SIGCOMM. (2001) 3–14
8. Yu, S., Zhou, W., Doss, R., Jia, W.: Traceback of DDoS attacks using entropy variations. IEEE Transactions on Parallel and Distributed Systems **22**(3) (March 2011) 412–425
9. Mitzenmacher, M., Upfal, E.: Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press (2005)
10. CAIDA: Skitter project. http://www.caida.org/tools/measurement/skitter/