Adaptive Timing-based Active Watermarking for Attack Attribution through Stepping Stones

Young Hee Park   Douglas S. Reeves
Cyber Defense Laboratory
Department of Computer Science
North Carolina State University
Raleigh, North Carolina, USA

Abstract

The detection of the origin of a stepping stone attack has been difficult due to skillful evasion techniques such as encrypting attack traffic and introducing a random delay (i.e., timing perturbation). Timing-based active watermarking schemes, which manipulate packet timing in order to embed a watermark, can counteract these two evasion techniques. However, the schemes for a fixed set of watermark parameter values will not tolerate any timing perturbation which results in a distortion of the embedded watermark. In this paper, we propose an adaptive timing-based active watermarking scheme which can tolerate any timing perturbation due to an adaptive choice of the parameter values relative to the timing characteristics of each traced traffic. Our scheme consists of two different algorithms; one of them is based on packet timing, and the other algorithm employs packet size. We have evaluated our scheme using real SSH traffic. The results demonstrate that our scheme accomplishes almost 100% of the watermark detection rate up to average 8000 milliseconds of timing perturbation as well as almost 0% of the false positive rate. Moreover, even if significantly fewer packets and less timing change in packets in the past are applied, our scheme achieves a similar watermark detection rate as the previous scheme.

1 Introduction

The number of network-based attacks has been increasing following the rapid growth of high-speed networks. It is easy for attackers to hide their identity by a series of stepping stones (i.e., compromised intermediate hosts), and anonymous networks such as Tor [4] and FindNot [6] for the purpose of shifting responsibility to another user. As a result, the development of effective techniques which can detect the true source of the attack through a chain of stepping stones has been challenging. These techniques are needed to prevent malicious use of the network and to result in appropriate punishment.

Various connection correlation techniques have been proposed in the past decade to detect stepping stones. Among those, timing-based correlation techniques (e.g., [3, 16–18, 20, 22]), which use only timing information of packets, have been the most promising because they can be applied to encrypted connections such as SSH and IPsec. However, the attackers can also add a random delay, called timing perturbation which is a big obstacle to all the timing-based correlation techniques, on each connection between stepping stones to avoid the timing analysis.

Timing-based active watermarking techniques are the most desirable approaches to solve the problem of timing perturbation (e.g., [15–17]). In particular, even if a small amount of timing perturbation exists, the probabilistic watermarking scheme can uncover a series of stepping stones by actively changing packet timing of randomly selected packets to insert a watermark into suspicious traffic according to the fixed values of watermark parameters. It can also frustrate anonymous networks as shown in [15].

However, because the probabilistic watermarking scheme has utilized a fixed set of the parameter values, this scheme cannot effectively deal with any strong timing perturbation, which means that attackers insert a huge amount of delay or they do not introduce independently and identically distributed timing perturbation as assumed in [16, 17]. Strong timing perturbation can make the embedded watermark distorted; thus it decreases correlation effectiveness. Additionally, if the important parameter values are not carefully selected, attackers can discover watermark parameters by using attack schemes proposed in [9]. We define that a good watermark should be robust against any timing perturbation, unnoticeable by attackers, and survivable in case that the watermark is removed or damaged.

In this paper, we propose an adaptive timing-based active watermarking scheme which can create the good watermark by the careful selection of the parameter values through exploitation of timing characteristics of each traced traffic. In general, adaptive digital watermarking achieved through different classification algorithms according to various media types can be a unique solution to satisfy the basic properties of digital watermarking: robustness, invisibility, and security [8, 13, 14]. Similarly, in our scheme, the adaptation to individual traced traffic enables the probabilistic watermarking scheme to embed the good watermark to achieve the basic properties. Therefore, we adaptively make a decision on the parameter values: (1) Which randomly selected packets are more desirable for embedding? (2) How much delay in packet timing is needed? (3) How many watermark bits are embedded? (4) How many packets are needed for embedding?

The adaptive timing-based watermarking scheme is comprised of two different algorithms for the selection of the parameter values. One of them determines the parameter values on the basis of only packet timing between two packets in incoming traffic near a target. The other algorithm determines the values by using the relationship between packet timing and packet size as follows; packet timing between two small size packets is generally longer that packet timing between two large size packets in interactive connections. Our experimental results demonstrate that our scheme endures any strong timing perturbation with considerably high watermark detection rate. Moreover, even if a smaller delay and significantly fewer packets are used, our scheme can show a similar ability of watermark detection as the probabilistic watermarking scheme.

The main contribution of this paper is that we present worthwhile adaptive watermarking schemes to overwhelm any timing perturbation where all the timing-based correlation techniques have failed in detecting stepping stones. In addition, timing characteristics of interactive connections as demonstrated in this paper can be successfully used with other connection correlation techniques.

The remainder of this paper is organized as follows. We briefly describe the probabilistic watermarking scheme with related work in section 2. In section 3, we propose an adaptive timing-based active watermarking scheme. We also show one valuable packet timing characteristics in interactive connections in section 4. After that, we evaluate our scheme with real traffic in section 5. Lastly, we sum up this paper with conclusion in section 6.

2 Related Work

2.1 Connection Correlation Techniques

Connection correlation techniques used to identify the origin of attacks through stepping stones have been developed by using different features: host activity, packet content, and packet timing. Host activity based approaches such as CIS [7] and DIDS [10] can be easily destroyed because of the use of unreliable login information at each stepping stone. With packet content based approaches such as Thumprinting [12], these techniques cannot be applied to encrypted connections since they depends on only packet payload. Therefore, timing-based correlation techniques, which uses only packet timing for correlation, have been developed because they can be useful for the encrypted connections.

Among timing-based correlation approaches, ON/OFF-based [22], deviation-based [20], and IPD-based [18] approaches passively observe inter-packet timing characteristics of traffic which is preserved across all the stepping stone connections. In particular, the passive packet-counting approach [3], which uses the difference of packet number in certain time intervals, gives us theoretically polynomial upper bounds on the number of packets needed to confidently detect stepping stones. However, these approaches cannot tolerate timing perturbation although the ability of attackers' evasion is limited in theory [5]. On the other hand, timing-based active watermarking approaches can endure a certain degree of timing perturbation by actively delaying selected packets [16, 17]. Moreover, the probabilistic watermarking scheme can effectively identify encrypted peer-to-peer VoIP calls with low-latency anonymous networks [15]. Attackers can recover and duplicate a embedded watermark by inferring the values of watermark parameters for the purpose of defeating traceback systems [9].

2.2 Probabilistic Watermarking Scheme

In brief, we describe the probabilistic watermarking scheme in [17] as it is related to our work, and the notations in this section will be also used in later sections.

In this scheme, watermark w, which preserves all the stepping stone connections, is embedded into outgoing traffic from a target by manipulating the packet timing of some selected packets. Each interactive stepping stone connection can be identified if a watermark is sufficiently unique to distinguish watermarked flows from unwatermarked flows.

First of all, similar to Figure 2, we independently and randomly select $2M$ distinct packets $< P_{k_1}, P_{k_2}, ..., P_{k_{2M}} >$ from given traffic $< P_1, P_2, ..., P_n >$ with time stamps of packets t_1, t_2, ... , t_n, respectively. The randomly selected packets are composed of $2M$ packet pairs such as $< P_{k_i}, P_{k_{i+d}} >$ $(d \geq 1, i = 1, .., 2M)$, and we define the *inter-packet delay*(IPD) as $ipd_{k_i} = t_{k_{i+d}} - t_{k_i} (i = 1, ..., 2M)$. We also randomly divide $2M$ IPDs into two distinct packet groups: packet group $1(pg1)$ with $ipd_{1,j}$ and packet group $2(pg2)$ with $ipd_{2,j} (1 \leq j \leq M)$. To embed a single watermark bit, we make use of the average difference of M IPDs from pg1 and pg2 as in the Equation (1), and M is called redundancy number. (Note that $E(ipd_{1,j}) = E(ipd_{2,j}) = 0$ because these IPDs is iid and $E(\overline{Y_M})$ should be also centered around 0)

$$\overline{Y_M} = \frac{1}{2M} \sum_{j=1}^{M} (ipd_{1,j} - ipd_{2,j}) \qquad (1)$$

As referred to Figure 2, to embed a watermark bit 1, we increase $\overline{Y_M}$ by the predefined timing adjustment A through increasing $ipd_{1,j}$ in $pg1$ by A and decreasing $ipd_{2,j}$ in $pg2$ by A. Reversely, embedding a watermark bit 0 is performed by decreasing $\overline{Y_M}$ by A. Consequently, the watermark bits can be decoded by checking the sign $\overline{Y_M}$. In other words, if the sign $\overline{Y_M}$ is positive(negative), the embedded watermark bit is decoded as 1(0). The above process is repeatedly performed as many times as the length of watermark bits L predetermines.

3 Adaptive Timing-based Active Watermarking Scheme

3.1 Watermark Parameters

In this section, we discuss the disadvantages of the probabilistic watermarking scheme compared to our adaptive timing-based active watermarking scheme.

The probabilistic watermarking scheme has many different watermark parameters as in Table 3.1 for embedding. The parameter values should be determined beforehand because embedders and detectors should confidentially share the values for embedding and decoding a unique watermark.

However, the predefined values of the parameters result in several troubles. First, the success of embedding a watermark cannot be guaranteed because this scheme has a slight probability (i.e., $Pr(|\overline{Y_M}| \geq A)$) that each watermark bit cannot be correctly embedded on grounds of IPD difference distribution as indicated in [17]. Second, it can fail to trace the origin of attacks if predefined watermark bits are not completely embedded due to the deficiency in the number of packets. To prevent this problem, the use of smaller values M makes this scheme intolerable against any timing perturbation. If higher values A are applied, it may give attackers a higher chance to defeat this scheme [9]. The reason is that it makes attackers aware that they are being traced due to the backward domino effect which happens in order to maintain the original order of packets. In contrast, the use of lower values A can make embedding success rate diminish; hence watermark detection rate is also reduced. Above all, this scheme with the fixed values of the parameters cannot be completely robust against timing perturbation by attackers. Therefore, we should scrupulously choose these parameter values which directly affect the performance of watermark detection.

We propose an adaptive timing-based active watermarking scheme which can effectively deal with the various issues as discussed earlier. Our scheme determines the parameter values according to the IPD characteristics of each traced traffic. Therefore, compared with the probabilistic watermarking scheme, our scheme has various advantages as follows:

- More robustness against any timing perturbation
- Higher watermarking detection rate due to the increase in embedding success rate
- Lower false positive rate due to high tolerance of embedded watermark
- More efficiency due to the use of fewer packets
- More stealthiness due to the use of smaller timing adjustment

Term	Description
L	Watermark length in bits ($L \geq 1$)
w_p	Each value of watermark w in binary ($1 \leq p \leq L$)
A	Maximum amount by which any packet is delayed
M	The number of packets needed to embed one bit
Packet Group(PG)	Randomly selected packets P_{k_i} for $pg1$ and $pg2$

Table 1. Watermark parameters

3.2 Adaptive Watermarking Schemes

In this section, we present our tracing model for adaptive timing-based active watermarking scheme as shown in Figure 1, and we introduce our adaptive watermarking scheme.

Generally, to launch an attack through several stepping stones, attackers first establish a series of connections by using SSH and Telnet protocol, called interactive connections, as shown in Figure 1. The traffic in interactive connections consists of forward flows $\{f_1, f_2, ..., f_n\}$, which are incoming traffic into the target and backward flows $\{b_1, b_2, ..., b_n\}$, which are outgoing traffic from the target. The traceback of the attack origin is performed in only the backward flows.

As shown in Figure 1, an adaptive watermarking system has a tuple $< O, W, WP, E_{WP}, D_{WP}, C_H >$, where O is the set of all original flows, W the set of all watermarks w, and WP the set of all parameters(i.e., $WP = \{w,\ L,\ A,\ M, PG\}$). Embedders form the watermarked carrier C_w into the closest backward flow b_n to the target with the several input parameters such as the original carrier object C_o with b_n, the watermark w to be embedded, and secret information WP as follows (Note that each values in WP is adaptively selected for embedding a good watermark before creating C_w):

$$E_{WP}(C_o, w) = C_w \tag{2}$$

Detectors extract w' from the possibly manipulated carrier object C'_w in each backward flow between stepping stones with the adaptively selected values in WP by embedders as follows:

$$D_{WP}(C'_w) = w' \tag{3}$$

Detectors utilize neither the original carrier object C_o nor the watermark w in the decoding process. The adaptively selected values in WP are a sort of secret keys which only embedders and detectors share for embedding and decoding. Lastly, the extracted watermark w' is compared with the original embedded watermark w by a threshold H:

$$Diff(w,\ w') \leq H \tag{4}$$

If the difference between originally embedded watermark w and decoded watermark w' is less than the predefined threshold H as in (4), detectors notify a target or traceback systems about the detection of stepping stones.

We present two different adaptive watermarking schemes, which means that embedders can determine the parameter values for themselves according to IPD characteristics of individual traced traffic for embedding a good watermark. The two schemes focus on the packet classification for two distinct packet groups: $pg1$ and $pg2$. The other parameter values except for the packet groups can be also adaptively determined by embedders without the degradation of performance. We call these values adaptive factors.

The first adaptive watermarking scheme, called *adaptation-I*, as in Algorithm 1, directly uses IPDs between two packets in b_n. First of all, embedders set another parameter μ, a timing value as a standard for packet classification for $pg1$ and $pg2$, by monitoring IPDs in b_n. To satisfy Equation (5), embedders carefully single out embedded packets $P_{k_i}(i = 1, ..., M)$

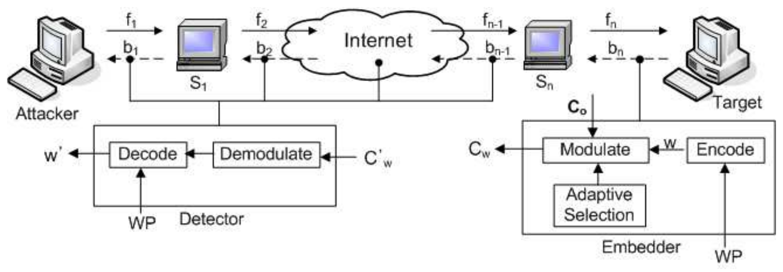


Figure 1. Tracing model for adaptive timing-based active watermarking scheme

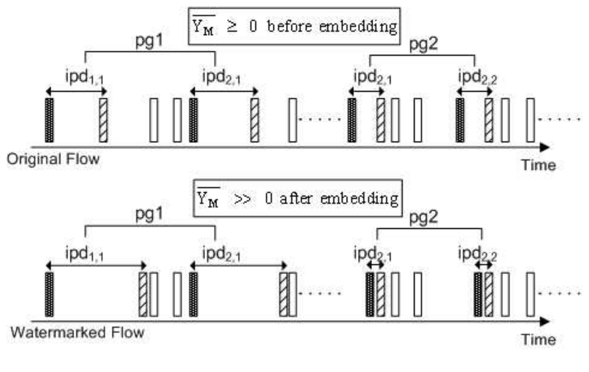


Figure 2. Embedding single watermark bit '1'

from packets $P_i(i = 1, ..., n)$ in b_n by comparing ipd_{k_i} with the standard μ according to each single watermark bit w_p. Specifically, as shown in Figure 2, to embed a watermark bit 1, $pg1$ consists of large IPDs and $pg2$ is made up of small IPDs. By doing so, because the sign $\overline{Y_M}$ becomes positive before embedding, we can initially have an embedding effect. Furthermore, by delaying the selected packets by A, we can make $\overline{Y_M}$ have a bigger average difference as in Figure 2. Likewise, to embed a watemark bit 0, $pg1$ is a set of small IPDs and $pg2$ is composed of large IPDs.

$$\begin{cases} \overline{Y_M} \geq 0 & \text{if } w_p = 1\ (1 \leq p \leq L) \\ \overline{Y_M} < 0 & \text{if } w_p = 0 \end{cases} \tag{5}$$

The second adaptive watermarking scheme, called *adaptation-II*, as in Algorithm 2 is based on IPD distribution as demonstrated in the section 4. In other words, it makes use of the relationship between IPD and packet size; IPDs between two small size command packets is greater than IPDs between two large size result packets. Above all, embedders should determine another parameter λ, which is the size of command packets, to distinguish between command packets and results packets from randomly selected packets

Algorithm 1 Adaptation-I based on IPD

```
procedure ENCODE($w$, $A$, $M$, $L$, $\mu$, $PG$)
▷ Adaptive Factors: $A$, $M$, $L$, $\mu$, $PG$
    for $1 \leq p \leq L$ do
        for $1 \leq x \leq M$ do
            $i \leftarrow random()$                    ▷ Compose $< P_{k_i}, P_{k_{i+d}} >$
            $ipd_{k_i} \leftarrow t_{k_{i+d}} - t_{k_i}$
            $res \leftarrow compare(idp_{k_i}, \mu)$
            if $w_p = 0$ then
                $(res \leq 0)?PG.pg1[(p-1)*M+x] \leftarrow P_{k_i} : PG.pg2[(p-1)*M+x] \leftarrow P_{k_i}$
            else                                       ▷ $w_p = 1$
                $(res > 0)?PG.pg1[(p-1)*M+x] \leftarrow P_{k_i} : PG.pg2[(p-1)*M+x] \leftarrow P_{k_i}$
            end if
            $t_{k_{i+d}} \leftarrow t_{k_{i+d}} + A$
        end for
    end for
end procedure
```

$P_{k_i}(i = 1, ..., M)$ for each watermark bit. To satisfy Equation (5), embedders classify the given packets P_{k_i} into two distinct packet groups. More specifically, as shown in Figure 2, $pg1$ is a set of command packets with large IPDs and $pg2$ is a set of result packets with small IPDs for a watermark bit 1. Accordingly, embedders can make $\overline{Y_M}$ positive before embedding. After embedding, the average difference of $\overline{Y_M}$ becomes greater as shown in Figure 2. Encoding a watermark bit 0 is the reverse case of a watermark bit 1.

By utilizing large IPDs to make $\overline{Y_M}$ be a large average difference, the two adaptive watermarking schemes can achieve more robustness against any timing perturbation and increase embedding success rate(i.e, $p = Pr(\overline{Y_M} < A) \approx \Phi(\frac{A\sqrt{M}}{\sigma_{Y_M}})$, where Φ is the cumulative distribution function of normal distribution). In fact, in the probabilistic watermarking scheme, the large IPDs should be filtered out to increase the embedding success rate. As a result, due to the increase of embedding success rate, our scheme

Algorithm 2 Adaptation-II based on packet size

```
procedure ENCODE($w$, $A$, $M$, $L$, $\lambda$, $PG$)
  ▷ Adaptive Factors: A, L, λ, PG
  for $1 \leq p \leq L$ do
    for $1 \leq x \leq M$ do
      $i \leftarrow nextsequence()$        ▷ Compose $< P_{k_i}, P_{k_{i+d}} >$
      $res1 \leftarrow compare(|P_{k_i}|, \lambda)$
      $res2 \leftarrow compare(|P_{k_{i+d}}|, \lambda)$
      if $w_p = 0$ then
        if $(res1 > 0 \; and \; res2 > 0)$ then
          $PG.pg1[(p-1)*M+x] \leftarrow P_{k_i}$
        else if $(res1 \leq 0 \; and \; res2 \leq 0)$ then
          $PG.pg2[(p-1)*M+x] \leftarrow P_{k_i}$
        end if
      else
        if $(res1 \leq 0 \; and \; res2 \leq 0)$ then
          $PG.pg1[(p-1)*M+x] \leftarrow P_{k_i}$
        else if $(res1 > 0 \; and \; res2 > 0)$ then
          $PG.pg2[(p-1)*M+x] \leftarrow P_{k_i}$
        end if
      end if
      $t_{k_{i+d}} \leftarrow t_{k_{i+d}} + A$
    end for
  end for
end procedure
```

can expect higher watermark detection rate as in Equation (6) in [17]. Our scheme can obtain a similar watermark detection rate even with the use of small values in A and M. Moreover, embedders can adaptively produce a long watermark w instead of the predefined L because this process can be repeatedly performed as long as P_i exist in each traced traffic.

$$D_t = \sum_{i=1}^{H} \binom{L}{i} p^{(L-i)}(1-p)^i \qquad (6)$$

3.3 Watermark Detection

For the accurate detection of the embedded watermark w, detectors should know the parameter values used by embedders. In particular, embedded packets P_{k_i} for $pg1$ and $pg2$ are the most important factors in WP in terms of security.

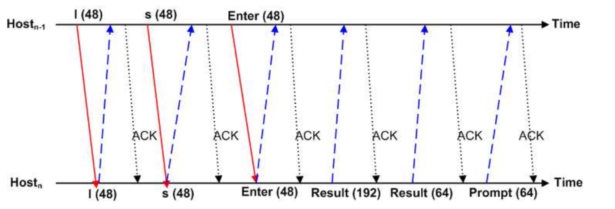
In *adaptation-I*, embedders must send the used parameter values to detectors because all the values in WP are adaptively determined by embedders. After receiving the values, detectors can report the detection of stepping stones through extracting w in off-line analysis. This off-line analysis can be realized by using SPIE(Source Path Isolation Engine) based on Blooming filter [2, 11]. Specifically, for watermark detection, detectos record only timing information of packets $P_i(i = 1, ..., n)$ with one source IP address and one destination IP address for limited time. The timing information can be either arrival time or departure time of P_i, and around 1,500 packets are sufficient to detect stepping stones efficiently as presented in section 5. Therefore, detectors can significantly reduce the storage space required for watermark detection because they do not need to save any portions of IP packets.

In *adaptation-II*, because embedders and detectors securely share the parameter values in WP beforehand as assumed in [17], detectors can decode the exact w in real time without receiving any information from embedders. In particular, from the given P_{k_i}, detectors accurately classify them into $pg1$ and $pg2$ based on packet size. It is achieved by easy prediction based on the IPD distribution as demonstrated in section 4. However, the process of the packet classification can not be performed by attackers because they do not have any knowledge in regard to P_{k_i} for two distinct packet groups.

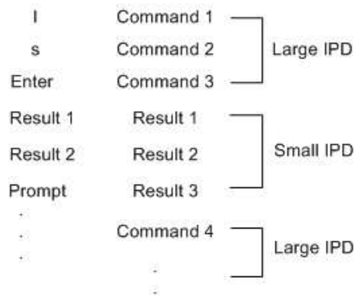
4 IPD distribution of Interactive Traffic

In this section, we demonstrate that traffic has a property related to inter-packet delay(IPD) by the keystroke of users in interactive connections.

We assume that attackers type shell commands on a keyboard in person for their attacks, called attack constraint. In an example of 'ls' shell command as shown in Figure 3(b), when attackers type the shell command on the keyboard, their keystroke is immediately transmitted as each individual IP packet into a forward flow (i.e., $Host_{n-1} \rightarrow Host_n$) as soon as they press a key on the keyboard (Note that the order of ACKs can be different.). After that, each keystroke of the shell command echo back into a backward flow (i.e., $Host_{n-1} \leftarrow Host_n$) with the corresponding results. In particular, each packet by keystrokes of shell commands is the smallest size, and the size of result packets is larger than command packets. For example, each keystroke in Telnet is 1 byte except for IP header size. In case of SSH, the payload size of each keystroke packet is generated by the form $8k$ or $8k+4$ according to SSH specification [19, 21]. Therefore, we can easily distinguish between command packets and results packets with packet size in both of Telnet and SSH connections. By analyzing IPDs in the backward flow, we found one IPD characteristics; IPDs between two small sized command packets are largely greater than IPDs between two large sized result packets as shown in Figure 3(b).



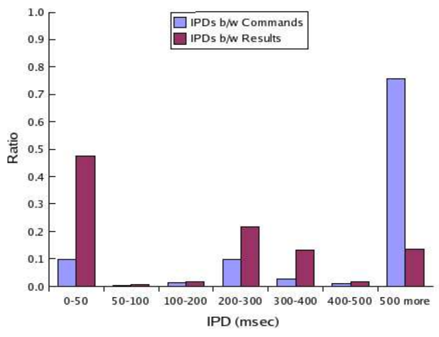
(a) 'ls' shell command in SSH



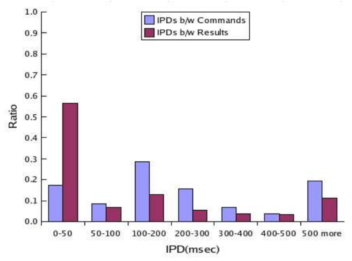
(b) IPD characteristics

Figure 3. Traffic characteristics in interactive connections

To validate the IPD characteristics in the interactive connections, we investigated IPDs (i.e.,$ipd_{k_i} = t_{k_{i+d}} - t_{k_i}(d = 1)$) with real Telnet traffic and SSH traffic which are extracted as only backward flows from the 2004 AUCK directory at NLANR [1]. For instance, we extracted the backward flows for SSH (Telnet) from source port 22 (23) for the same IP address. However, because the extracted flows do not have a tendency to follow the attack constraint, we should classify all packets (i.e., $< P_{k_i}, P_{k_{i+d}} > (d = 1)$) into a pair of command packets or a pair of result packets for the exact analysis. In other words, in case of Telnet traffic, if each packet size of a pair is 1 byte except for IP header size, we regard the pair as a set of command packets C. If the packet size is bigger than 10 bytes, the pair is categorized as a set of result packets R. Similarly, with SSH traffic, if each payload size of a pair is 48 bytes, the pair is classified as C. If not, we consider the pair as R. The reason is that the smallest payload size is 48 bytes in most cases of the extracted SSH flows. As a result, among 103,259 packets in the Telnet traffic, 44,939 packets are classified as C and 58,320 packets are considered as R. In case of the SSH traffic, among 3,000,872 packets, the number of C is 570,511 and the number of R is 2,430,361.



(a) Telnet



(b) SSH

Figure 4. IPD Distribution between shell command packets or between result packets

As shown in the Figure 4, we show a histogram of IPD distribution to prove the IPD characteristics. First, we divide x-axis into seven possible bins, and each bar represents a ratio of the number of the pairs in the associated bin over the total number of C or R. With Telnet flows as in Figure 4(a), nearly 75% of the IPDs in C are more than 500ms; on the other hand, nearly 48% of IPDs in R fall within less than 50ms. In addition, IPDs with more than 100ms in C are close to 90% in C, and almost 55% of the IPDs in R are less than 100ms. In case of SSH flows as shown in Figure 4(b), IPDs with more than 100ms in C are almost 75%, and

64% of the IPDs in R are within 100ms. Moreover, around 58% of the IPDs in a set R fall within 50ms. Therefore, we make sure that most of the IPDs in C are greater than IPDs in R in interactive connections. In particular, we could prove that the IPD distribution in Telnet flows is more significantly obvious than SSH flows because the payload size of result packets with a few bytes could also be 48 bytes in the SSH flows.

Parameters	Values
L	24 bits
w_p	100110001100111101110000
A	from 50ms to 600ms
M	from 1 to 5
Packet Group	P_{k_i}(i=1, 5, 9, ...)

Table 2. Watermark parameters used for experiments

5 Performance Evaluation

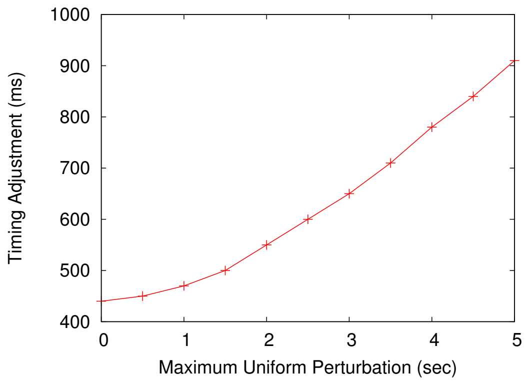
We evaluate the performance of our two adaptive watermarking schemes, compared with the probabilistic watermarking scheme. These experiments show the importance of watermark parameters in different views: timing adjustment (A), redundancy number (M), threshold (H) and the number of packets needed for successful correlation.

In this experiment, we use real 112 SSH flows from the AUCK directory at NLANR [1] as discussed in section 4. Each flow has more than 1,000 packets. All the flows are backward flows, which means that they consist of command packets and result packets.

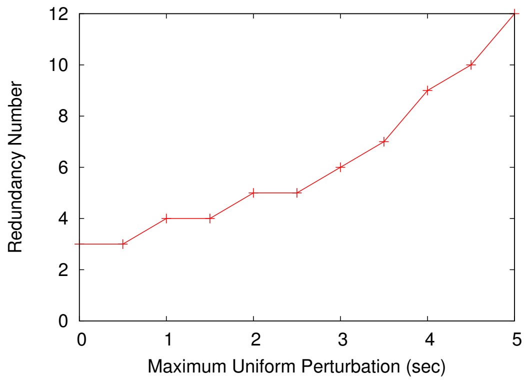
We estimate watermark detection rate (D_t) with the various values of watermark parameters as shown in Table 2. In particular, μ is used as median of IPDs in each flow. The value μ is changed according to each SSH flow used in our experiments. λ is set at 48 because the smallest payload size of packets in most of the extracted SSH flows is 48 bytes. We use two types of timing perturbation: uniform perturbation (iid) and batch-releasing perturbation (non-iid) as in [17]. In other words, uniform perturbation means that attackers introduce uniformly distributed random delay from 0 to the maximum delay. On the other hand, batch-releasing perturbation, attackers periodically release several packets in a burst after holding the packets arriving at stepping stones for a certain time. Specially, this batch-release perturbation have been a hard case for any timing-based correlation techniques, and it can make possible completely to defeat all the traceback systems using packet timing.

Watermark detection(D_t) is evaluated if the watermarked flow can be detected from its perturbed flow after embedding a 24-bit watermark w into each flow. As in Figure 6, our scheme with *Adaptation-I* and *Adaptation-II* always surpasses the probabilistic watermarking scheme under both timing perturbation in the same conditions such as L=24, M=5, A=600 and H=5. In these experiments, we use only different embedded packets P_{k_i} for two distinct packet groups. More specifically, *Adaptation-I* accomplishes almost 100% D_t up to a maximum of 10000ms, where it extremely distorts the emebedded watermark w, in uniform and batch-releasing perturbation. *Adaptation-II* achieves $D_t \geq 95\%$ up to a maximum of 7500ms of uniform perturbation or up to 6000ms of batch-releasing perturbation. However, the probabilistic watermarking scheme shows $D_t \geq 95\%$ only up to a maximum of 3500ms of uniform perturbation or only up to 3000ms of batch-releasing perturbation. As a result, we can assure that *Adaptation-I* can completely overwhelm any timing perturbation by carefully selecting P_{k_i} for $pg1$ and $pg2$. Additionally, compared with the probabilistic watermarking scheme, *Adaptation-II* can achieve D_t twice as well as the probabilistic watermarking scheme only by differently classifying the same P_{k_i} into two distinct packet groups. Therefore, the packet classification for two distinct packet groups is an influential parameter for effectiveness of watermark detection.

Timing adjustment(A) determines how much delay is needed to achieve $D_t \geq 95\%$ under a certain timing perturbation. First, as shown in Figure 5(a), the expected A is theoretically derived to achieve $D_t \geq 95\%$ under uniform perturbation from Equation (6). To $D_t \geq 95\%$ with $L = 24$, $M = 5$ and $H = 5$ up to a maximum of 2500ms of uniform perturbation, $A = 600$ is required. In real experiments as in Figure 7, the probabilistic watermarking scheme can show $D_t \geq 95\%$ along with the same parameter values up to a maximum of 3500ms of uniform perturbation. However, as discussed earlier, our scheme with the same values can show substantially higher D_t than the probabilistic watermarking scheme. Furthermore, as

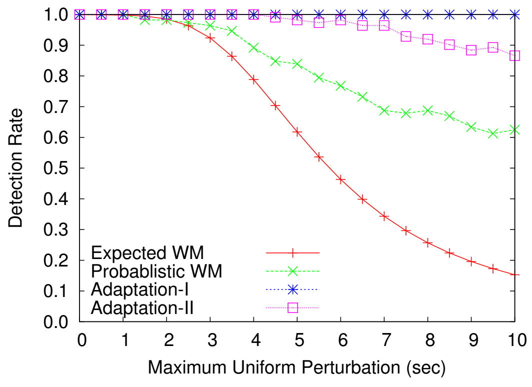


(a) $L = 24, M = 5, H = 5, D_t \geq 95$

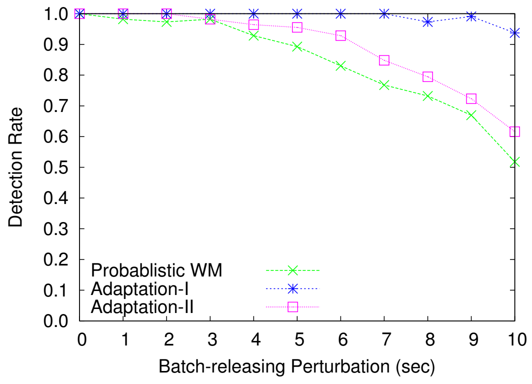


(b) $L = 24, A = 600, H = 5, D_t \geq 95$

Figure 5. Expected timing adjustment A and expected redundancy number M to achieve $D_t \geq 95\%$ under uniform perturbation



(a) $L = 24, M = 5, A = 600, H = 5$

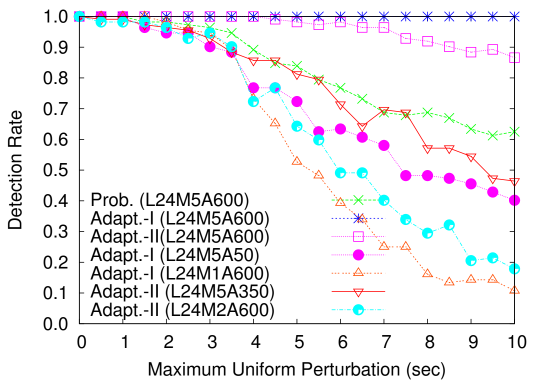


(b) $L = 24, M = 5, A = 600, H = 5$

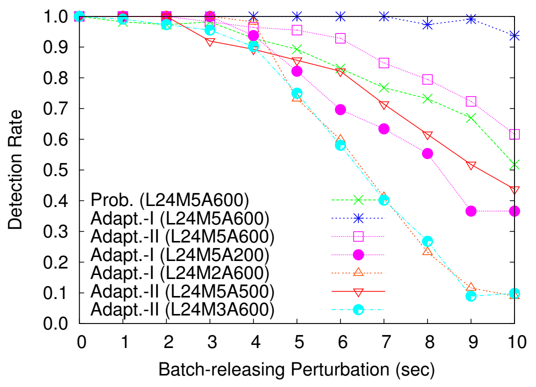
Figure 6. Watermark detection rate comparisons

shown Figure 7, our scheme can accomplish a similar D_t as in the probabilistic watermarking scheme even if lower values A than the expected values are used. Out of many experiments which we have performed, in Figure 7, we show only a comparable D_t as in the probabilistic watermarking scheme. In other words, we guarantee that the larger values than the used values in Figure 7 achieve higher D_t than the presented results. In detail, *Adaptation-I* with only $A = 50$ and *Adaptation-II* with only $A = 350$ can achieve $D_t \geq 95\%$ up to almost a maximum of 3000ms of uniform timing peturbations along with $L = 24$, $M = 5$ and $H = 5$. In case of batch-releasing perturbation, *Adaptation-I* with only $A = 200$ can show $D_t \geq 95\%$ up to 4000ms. *Adaptation-II* with $A = 500$ can obtain $D_t \geq 95\%$ up to 3000ms of batch-releasing perturbation. Therefore, if A is less than 500 in our scheme, our scheme can still accomplish a similar or higher D_t, compared with the probabilistic watermarking scheme.

Redundancy number(M) determines how many packets are required to produce $D_t \geq 95\%$ under a certain timing perturbation. Similar to timing adjustment, the expected M is derived to make $D_t \geq 95\%$ under unifrom perturbation as indicated in Figure 5(b). As discussed earlier, $M = 5$ is needed to achieve $D_t \geq 95\%$ up to a maximum of 2500ms of uniform perturbation with $L = 24$, $A = 600$ and $H = 5$. However, our scheme with the same values can achieve sig-

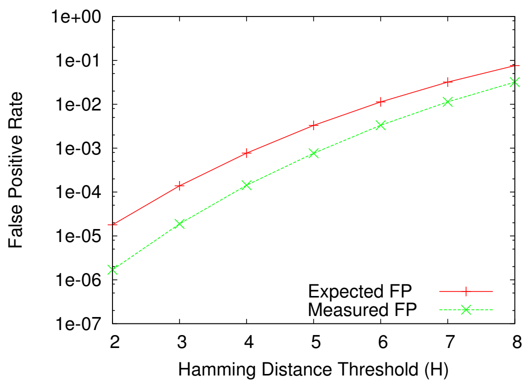


(a)



(b)

Figure 7. Watermark detection rate according to various values of A and M



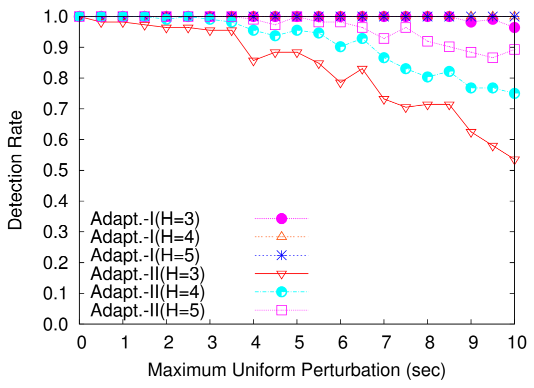
(a)

Figure 8. False positive rate

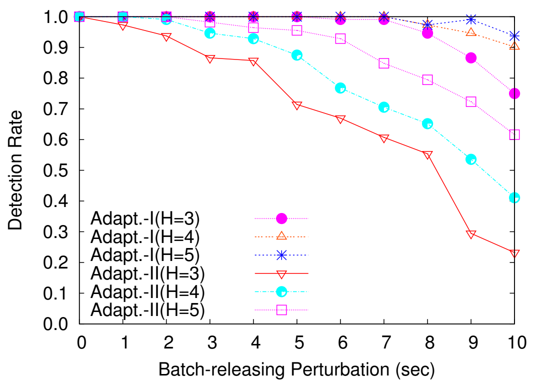
nificanlty higher D_t than the probabilistic watermarking scheme as shown in Figure 6. Moreover, even if smaller values M than the expected values are applied, our scheme can produce an analogous D_t to the probabilistic watermarking scheme as in Figure 7. More specifically, only if $M = 1$ with $L = 24$, $A = 600$ and $H = 5$ is used, *Adaptation-I* can accomplish $D_t \geq 95\%$ up to a maximum of 3000ms of uniform perturbation, and so does *Adaptation-II* with $M = 2$. In addition, up to 4000ms of batch-releasing perturbation, *Adaptation-I* with $M = 2$ can produce the same D_t with $L = 24$, $A = 600$ and $H = 5$. *Adaptation-II* with $M = 3$ can obtain $D_t \geq 95\%$ up to 3000ms of batch-releasing perturbation along with the same parameter values. Therefore, if M is less than 3 in our scheme, we make sure that our scheme can get similar or higher D_t, compared with the probabilistic watermarking scheme.

False positive rate(F_p) is estimated if unwatermarked flows are erroneously reported as watermarked flows. As in Figure 8, we experimentally evaluate a false positive rate F_p between a specific watermark w in a given flow from the 112 SSH flows and 100,000 randomly generated 24-bit watermarks under various threshold values H according to Equation (7) as shown in [17]. Our scheme still maintains $F_p \leq 1\%$ for $H \leq 5$. In fact, in the probabilistic watermarking scheme, the threshold H is used for the guarantee of high watermark detection rate because w can be distorted by timing perturbation. The higher the H is, the higher the false positive rate F_p is as in Equation (7).

However, as shown in Figure 9 which shows D_t with the several values H under both timing perturbation, even if $H = 3(F_p \leq 0.01\%)$ is used, *Adaptation-I* achieves almost 100% D_t under any strong timing perturbation with $L = 24$, $M = 5$ and $A = 600$. Additionally, *Adaptation-II* with $H = 4$ can produce $D_t \geq 95\%$ up to a maximum of 5500ms of uniform perturbation or up to 4000ms of batch-releasing perturbation along with the same parameter values. As a result, our scheme can show significantly high D_t while the lowest possible F_p with the use of small H because our scheme can prevent w from tampering by timing perturbation. In other words, it means that the adaptation to each traced traffic makes the em-



(a) $L = 24, M = 5, A = 600$



(b) $L = 24, M = 5, A = 600$

Figure 9. Watermark detection rate according to threshold H

bedded watermark more robust against any timing perturbation. It also decreases F_p by using a long watermark w because we need significantly fewer packets to achieve the analogous D_t to the previous scheme. Consequently, detectors can confidently detect a series of stepping stones with higher D_t as well as lower F_p (amost 0% F_p) than the probabilistic watermarking scheme.

$$F_p = \binom{L}{H} 2^L \qquad (7)$$

The number of packets, which are needed to guarantee the same level of successful correlation under a certain timing perturbation, is calculated with the probabilistic watermarking scheme and the passive packet-counting scheme [3]. Our scheme requires significantly fewer packets than the two other schemes. In other words, in our experiments, our scheme needs between 48 IPDs and 144 IPDs ($1 \leq M \leq 3$) to achieve almost 100% D_t with around 0.3% F_p up to a maximum of 1000ms of both timing perturbations. However, the probabilistic watermarking scheme requires 240 IPDs with $M = 5$ under the same conditions as our scheme does. From 112 SSH flows in our experiments, the passive-counting scheme needs over 15,000 packets which is the upper bound to achieve 100% detection rate and 1% F_p up to 1000ms of timing perturbation. As a result, our scheme uses at most half of the number of packet needed to accomplish the same D_t as the probabilistic watermarking scheme. Compared with the passive-counting scheme, our scheme requires considerably fewer packets to guarantee a similar performance for correlation.

6 Conclusion

The probabilistic watermarking scheme can effectively detect stepping stones and also trace back anonymous networks. However, this scheme for a fixed set of watermark parameter values cannot tolerate any timing perturbation. In this paper, we propose an adaptive timing-based watermarking scheme consisting of two different adaptive schemes by using IPD characteristics in interactive connections. One of them is based on inter-packet delay(IPD) itself, and the other scheme utilizes the relationship between the IPD and the packet size. By using the adaptive watermarking schemes, we can completely counteract timing perturbation which have been challenging in most of the timing-based correlation techniques. Compared with the probablistic watermarking scheme, the results of our experiments have shown that our scheme can achieve a considerably high watermark detection rate while keeping the lowest false positive rate at the same time. These schemes can also accomplish a similar watermark detection rate even if the smaller values of timing adjustment A and redundancy number M are used. Furthermore, the small values A makes the watermarked flow less noticeable by attackers. The use of small values M can enable our scheme to trace the attack origin successfully with significantly fewer packets even if attack traffic is short. Furthermore, the concept of adaptation in our scheme can be extended to other timing-based correlation techniques.

References

[1] NLANR Trace Archive. http://pma.nlanr.net/Traces/Traces/long//.

[2] B. H. Bloom. Space/Time Trade-offs in Hash Coding with Allowable Errors. *Communication of ACM*, 13(7):422–426, 1970.

[3] A. Blum, D. X. Song, and S. Venkataraman. Detection of Interactive Stepping Stones: Algorithms and Confidence Bounds. In *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004)*, pages 258–277, October 2004.

[4] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. *Proceedings of the 13th USENIX Security Symposium*, pages 303–320, August 2004.

[5] D. L. Donoho, A. G. Flesia, U. Shankar, V. Paxson, J. Coit, and S. Staniford. Multiscale Stepping-Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detectionn (RAID 2002)*, pages 17–35, October 2002.

[6] FindNot.com. FindNot-Anonymous Surfing, Anonymous Email & Anomymous Internet. http://www.findnot.com/.

[7] H. T. Jung, H. L. Kim, Y. M. Seo, G. Choe, S. L. Min, C. S. Kim, and K. Koh. Caller Identification System in the Internet Environment. In *Proceedings of the 4th USENIX Security Symposium*, 1993.

[8] K. MS, Rajmohan, and R. KR. Adaptive visible watermarking of images. *IEEE international conference on multimedia computing and systems*, June 1999.

[9] P. Peng, P. Ning, and D. S. Reeves. On the Secrecy of Timing-Based Active Watermarking Trace-Back Techniques. *Proceedings of the 2006 IEEE Symposium on Security and Privacy (S&P)*, May 2006.

[10] S. R. Snapp, J. Brentano, G. V. Dias, T. L. Goan, L. T. Heberlein, C.-L. Ho, K. N. Levitt, B. Mukherjee, S. E. Smaha, T. Grance, D. M. Teal, and D. Mansur. DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and Early Prototype. In *Proceedings of the 14th National Computer Security Conference*, pages 167–176, 1991.

[11] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent, and W. T. Strayer. Single-Pakcet IP Traceback. *ACM SIGCOMM'01*, August 2001.

[12] S. Staniford-Chen and L. T. Heberlein. Holding intruders accountable on the Internet. In *Proceedings of the 1995 IEEE Symposium on Security and Privacy*, pages 39–49, 1995.

[13] B. Tao and B. Dickinson. Adaptive Watermarking in the DCT Domain. *IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP'97)*, 1997.

[14] B. Tao and B. Dickinson. Adaptive model-driven bit allocation for MPEG video coding. *IEEE Circuits and Systems for Video Technology*, pages Vol 10 pp.147–157, February 2000.

[15] X. Wang, S. Chen, and S. Jajodia. Tracking Anonymous Peer-to-Peer VoIP Calls on the Internet. *Proceedings of the 10th ACM conference on Computer and Communications Security (CCS)*, 42:81–91, November 2005.

[16] X. Wang and D. S. Reeves. Robust Correlation of Encrypted Attack Traffic through Stepping Stones by Manipulation of Interpacket Delays. In *Proceedings of the 10th ACM conference on Computer and Communications Security (CCS 2003)*, pages 20–29, October 2003.

[17] X. Wang, D. S. Reeves, P. Ning, and F. Feng. Robust Network-Based Attack Attribution through Probabilistic Watermarking of Packet Flows. Technical Report TR-2005-10, Department of Computer Science, NC State University, February 2005.

[18] X. Wang, D. S. Reeves, and S. F. Wu. Inter-Packet Delay Based Correlation for Tracing Encrypted Connections through Stepping Stones. In *Proceedings of the 7th European Symposium on Research in Computer Security (ESORICS 2002)*, pages 244–263, October 2002.

[19] T. Ylonen. The Secure Shell (SSH) Protocol Architecture. *IETF RFC:4251*, January 2006.

[20] K. Yoda and H. Etoh. Finding a Connection Chain for Tracing Intruders. In *Proceedings of the 6th European Symposium on Research in Computer Security (ESORICS 2000)*, pages 191–205, October 2000.

[21] Y. Zhang and V. Paxson. Detecting Backdoors. In *Proceedings of the 9th USENIX Security Symposium*, August 2000.

[22] Y. Zhang and V. Paxson. Detecting Stepping Stones. In *Proceedings of the 9th USENIX Security Symposium*, August 2000.