Exposing Invisible Timing-based Traffic Watermarks with BACKLIT

Xiapu Luo[‡], Peng Zhou[‡], Junjie Zhang[§],
Roberto Perdisci[†], Wenke Lee[§], and Rocky K. C. Chang[‡]
The Hong Kong Polytechnic University[‡] Georgia Institute of Technology[§] University of Georgia[†]
{csxluo|cspzhouroc|csrchang}@comp.polyu.edu.hk,
{jjzhang|wenke}@cc.gatech.edu, perdisci@cs.uga.edu

ABSTRACT

Traffic watermarking is an important element in many network security and privacy applications, such as tracing botnet C&C communications and deanonymizing peer-to-peer VoIP calls. The state-of-the-art traffic watermarking schemes are usually based on packet timing information and they are notoriously difficult to detect. In this paper, we show *for the first time* that even the most sophisticated timing-based watermarking schemes (e.g., RAINBOW and SWIRL) are not invisible by proposing a new detection system called BACKLIT. BACKLIT is designed according to the observation that any practical timing-based traffic watermark will cause noticeable alterations in the intrinsic timing features typical of TCP flows. We propose five metrics that are sufficient for detecting four state-of-the-art traffic watermarks for bulk transfer and interactive traffic. BACKLIT can be easily deployed in stepping stones and anonymity networks (e.g., Tor), because it does not rely on strong assumptions and can be realized in an active or passive mode. We have conducted extensive experiments to evaluate BACKLIT's detection performance using the PlanetLab platform. The results show that BACKLIT can detect watermarked network flows with high accuracy and few false positives.

1. INTRODUCTION

A traffic watermark is a piece of information embedded into a network flow. If the traffic watermark is retained in the network flow, it can be used to correlate network traffic observed at different network locations. Therefore, traffic watermarking has many important network security and privacy applications, including identifying stepping stones used by attackers to hide their true physical locations [1,2], tracking users visiting sensitive (e.g., pro-terrorism) Web sites [3], tracing communications among bot-compromised machines [4], and tracking anonymous peer-to-peer VoIP calls [5].

The state-of-the-art traffic watermarking schemes embed *timing-based watermarks* into a network flow by artificially manipulating packet timing information, such as inflating and deflating inter-packet delay [2, 3, 6, 7]. The timing-based watermarks have seen continuous improvement over the last few years, evolving from an initial sole focus on usability [1] to robustness [2, 3] and invisibility [6, 7]. The most recently proposed timing-based watermarks, such as RAINBOW [6] and SWIRL [7], are invisible as they can evade the existing detection schemes.

There are two general strategies for mitigating the traffic watermarking threat. The first strategy blindly removes the timing information brought by traffic watermarks through shaping and/or padding every flow no matter it is watermarked or not [8–10]. Although it is effective, this strategy introduces significant overhead such as high latency to all flows and useless packets consuming bandwidth, making them very difficult to scale. Another strategy is to first detect watermarked flows and then clean only the suspicious flows. Compared to the first strategy, the detection strategy introduces much less overhead for identifying watermarked flows and has the advantages of handling only the suspicious flows and discovering the existence of watermark stamping devices.

Detecting timing-based watermarks is a very challenging problem, because they do not have a fixed signature. So far, only a few detection mechanisms have been proposed [11, 12]. However, they suffer from two main drawbacks. First, they fail to detect advanced traffic watermarking schemes, such as RAINBOW and SWIRL, because these watermarking schemes may not cause anomalies in the features used by those detections mechanisms. Second, they rely on strong assumptions for the detection. For example, the detection system PNR (which is named by concatenating the first letters of the three authors' last names) [11] assumes that the traffic sender can inject a timestamp into each packet, but this is not feasible in many scenarios (e.g., a public Web server not controlled by the detection system). PNR's another assumption that one-way packet delay follows the Gaussian distribution also does not always hold [13]. In Multi-Flow Attack (MFA) [12], the assumptions of watermarking multiple flows with the same traffic watermark and modeling the flows as a Markov modulated Poisson process (MMPP) do not always hold either [14]. Kiyavash et al. [15] later showed how MFA can be easily evaded by randomizing the locations of watermarks or using different watermarks for different network flows.

In this paper, we demonstrate *for the first time* that even the most advanced timing-based traffic watermarking scheme can be detected by a practical system. We show this by proposing BACKLIT, a new system capable of detecting four advanced watermarking schemes, including the "invisible" RAINBOW and SWIRL, the interval based watermarking (IBW) scheme, and the interval centroid based watermarking (ICBW) scheme. We design BACKLIT based on the observation that any practical watermark that artificially perturbs packet timing information will cause noticeable alterations in the intrinsic *timing features* typical of TCP flows. We first select the TCP features that can be used for detection purposes, then design metrics to quantify these features, and finally employ the one-class

ACSAC '11 Dec. 5-9, 2011, Orlando, Florida USA

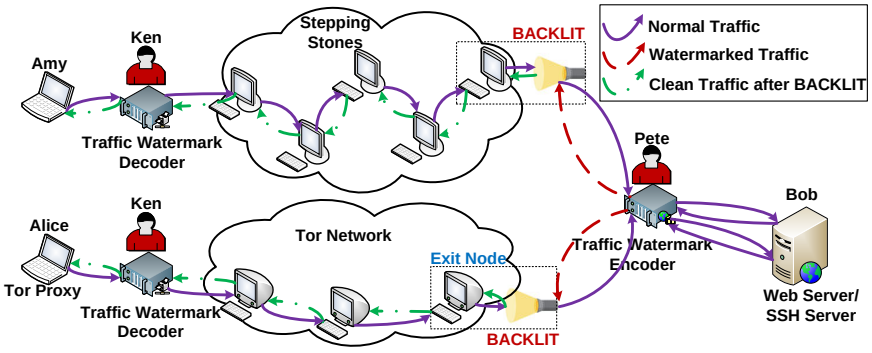


Figure 1: Typical scenarios of using traffic watermarks and BACKLIT.

classifier to detect watermarked flows based on the set of metrics. Moreover, unlike previous studies that use synthetic and replayed traffic for their evaluation [2,6,7,12], we evaluate BACKLIT using live HTTP traffic and SSH traffic through twelve PlanetLab nodes deployed around the world. The evaluation shows that BACKLIT can detect the four timing-based watermarking schemes with high accuracy, often achieving a 100% detection rate and low false positive rate.

Besides the high detection performance, BACKLIT can also be readily deployed in real network environments. BACKLIT is much more practically feasible than PNR and MFA, because it does not rely on strong assumptions required for PNR and MFA. BACKLIT is more flexible than PNR and MFA, because BACKLIT can be deployed as an active or passive system by either generating real requests or relaying requests from other users to induce response packets for detection. MFA is a passive-only system that makes a decision from observed traffic, and PNR is an active-only system that sends customized packets to detect the existence of traffic watermarks. Moreover, unlike PNR, BACKLIT does not need to modify packets and install special software at the remote server. In this paper, we focus on TCP flows, because TCP carries about 90% of Internet traffic [16], and popular anonymity networks (e.g., Tor [17]) currently only support TCP-based applications. However, the same idea could be applied to other end-to-end protocols.

The remainder of the paper is organized as follows. Section 2 describes the threat model and Section 3 introduces the state-of-the-art timing-based traffic watermarks. The design and implementation of BACKLIT are detailed in Section 4 and Section 5, respectively. We present evaluation results in Section 6. After introducing related work in Section 7, we conclude the paper with future work in Section 8.

2. THREAT MODEL

Figure 1 illustrates typical scenarios of using traffic watermarks. Bob runs a server that hosts *sensitive* content (e.g., WikiLeaks) that Alice and Amy visit through an anonymity network (e.g., Tor [17]) and a series of stepping stones [18], respectively. Pete, who has access to Bob's network traffic, embeds a sequence of traffic watermarks into the traffic originating from Bob in an attempt to trace back who is visiting Bob's server. Ken, Pete's associate, monitors the traffic directed towards Alice and Amy. If Ken can correctly decode the sequence of watermarks encoded by Pete, he can establish that Alice and Amy are visiting Bob's server.

The owner of the anonymity network or stepping stones can install BACKLIT to determine whether Bob's traffic is being watermarked. If that is the case, she can employ various packet padding and buffering strategies to remove traffic watermarks without affecting all incoming flows [8, 9]. BACKLIT either sends real requests (i.e., active mode) or employs relayed requests from other users (i.e., passive mode) to induce packets from Bob's server for detection. Therefore, Pete cannot distinguish whether the host running BACKLIT is actually relaying requests on behalf of a real user (e.g., Alice) or is attemping to discover Pete's presence. Thus, Pete cannot selectively avoid watermarking traffic directed from Bob to BACKLIT. We assume that Pete does not collude with Bob.

As shown in Figure 1, BACKLIT can be strategically deployed at an exit node of the Tor network or a stepping stone connecting directly to Bob's server to facilitate the detection. The deployment flexibility provides the advantage that the traffic observed by BACKLIT is less affected by noise in comparison with the traffic monitored by Ken, because the traffic observed by Ken has to pass through *more* hops within the anonymity network or stepping stones. Such deployment flexibility has also been used in PNR's [11] and MFA's [12] experiments where there is no relay node between the traffic watermark encoder and the detection system.

3. STATE-OF-THE-ART TIMING-BASED TRAFFIC WATERMARKS

This paper considers four state-of-the-art traffic watermarking schemes listed in Table 1. The interval based watermarking (IBW) scheme [2] and the interval centroid based watermarking (ICBW) scheme [3] do not consider invisibility in their design, but the more recently proposed RAINBOW and SWIRL do. In particular, RAINBOW and SWIRL introduce much shorter delay to the packets than IBW and ICBW. Although MFA can detect IBW and ICBW [12], they can evade the detection by stamping different watermarks to network flows and randomizing the location of watermarks [15]. As shown in subsequent sections, BACKLIT can detect all of these watermarking schemes. PNR [11] is not listed in Table 1 because there are no experiment results showing whether PNR can detect these traffic watermarks.

Table 1: The timing-based traffic watermarking schemes considered in this paper.

Traffic watermarking scheme	Designed for invisibility?	Inserted delay	Evades MFA?	Evades BACKLIT?
IBW [2]	No	Long	No	No
ICBW [3]	No	Long	No	No
SWIRL [7]	Yes	Short	Yes	No
RAINBOW [6]	Yes	Short	Yes	No

3.1 IBW

IBW adjusts the number of packets in selected time intervals to embed watermarks [2]. Starting from a random offset, the flow to be watermarked is divided into a series of intervals I_i of the same length T_{IBW}. Pairs of consecutive intervals are randomly chosen to encode a bit. As shown in Figures 2(a) and 2(b), to embed a 0, IBW forces the number of packets in interval I_k to be larger than that in interval I_{k+1} by delaying all packets within I_{k-1} to I_k and those within I_{k+1} to I_{k+2}. To embed an 1, IBW delays packets in I_k to I_{k+1}, so that I_k has fewer packets than I_{k+1}.

3.2 ICBW

ICBW varies the centroid of the packets within selected time intervals to encode watermarks [3]. Starting from a random offset, the target flow is partitioned into a series of intervals of the same length T_{ICBW}. As shown in Figure 2(c), to alter the centroid of the packets in a given interval, ICBW changes each packet's relative time to the start of its interval from ΔT to $\Delta T^{'} = \alpha_{ICBW} + \frac{(T_{ICBW} - \alpha_{ICBW})\Delta T}{T_{ICBW}}$, where α_{ICBW} determines the maximum delay applied to the packets. Before encoding a bit, ICBW randomly selects two sets of intervals, say I_A and I_B, from a series of consecutive intervals. To encode a 0 (or 1), ICBW forces the centroid of the packets in I_B (or I_A) to be larger than that in I_A (or I_B) by delaying packets in I_B (or I_A).

3.3 SWIRL

SWIRL changes the locations of packets within selected time slots to encode watermarks [7]. Starting from a random offset, SWIRL divides a flow into a series of intervals of the same length T_{SWIRL}. It further defines two intervals: the mark interval and the basic interval. As shown in Figure 2(d), SWIRL partitions a mark interval T_{SWIRL} into r subintervals of length T_{SWIRL}/r and then splits each subinterval into m slots. To embed traffic watermarks, SWIRL first selects a slot i in each subinterval j according to a permutation $\pi^{(j)}(s) = i$, where s is a variable determined by the basic interval's centroid, and then delays each packet from the original slot to the selected slot. If the original slot's index is larger than that of the selected slot, packets will be postponed to the selected slot in the next subinterval. The delay added to each packet is in the range of $[0, \frac{2(m-1)T_{SWIRL}}{mr}]$ [7].

3.4 RAINBOW

RAINBOW inflates or deflates an inter-packet delay (IPD) by T_{RB} with equal probability [6]. Let T^i_{RB} ($i = 1, \ldots$) denote the adjustment made to the ith IPD. If $T^i_{RB} > 0$, the IPD will be increased; otherwise, it will be shortened. Since a packet cannot be delayed for a negative amount of time, RAINBOW sets the delay of the jth ($j = 1, \ldots$) packet to $T_j = T_0 + \sum_{i=1}^{j-1} T^i_{RB}$, where T_0 is a parameter large enough to ensure $T_j \geq 0$. Figure 2(e) gives an example of RAINBOW.

4. BACKLIT

Table 2 summarizes the differences among PNR [11], MFA [12], and BACKLIT. BACKLIT is more practically feasible, because unlike PNR and MFA, BACKLIT neither needs a timestamp in each packet and the cooperation of the remote server, nor relies on traffic distribution assumptions. Instead, BACKLIT exploits TCP's intrinsic features to expose various traffic watermarks, which are elaborated in Section 4.1.

4.1 Feature selection

Since all timing-based traffic watermarking schemes delay packets, BACKLIT leverages three TCP features that will be noticeably altered by traffic watermarks for detection: (1) *request-response time* (RRT), the time elapsed between sending the last packet of a request message from BACKLIT and receiving the first response packet from the remote server, (2) *inter-packet delay* (IPD), the time between two consecutive packets from the remote server, and (3) *burst size*, the number of TCP packets sent back-to-back [19]. We calculate the burst size using the method in [19].

Figure 3(a) shows an example of *normal* IPD, RRT, and burst size in the absence of traffic watermarks. Upon receiving a request packet, the server sends back a burst of four packets. If a traffic watermarking scheme delays the fourth response packet, a disturbed IPD (i.e., denoted by IPD') will be observed, as shown in Figure 3(b). As a side effect of inflating the IPDs, the burst size will also be changed. Particularly, if the delay induced by a watermark is large enough, say not less than the round-trip time (RTT), we can discern two bursts of packets: one has three packets and the other has one. Similarly, if a traffic watermarking scheme delays the first response packet (e.g., the second delay in Figure 3(b)), we could observe an increased RRT (i.e., denoted by RRT').

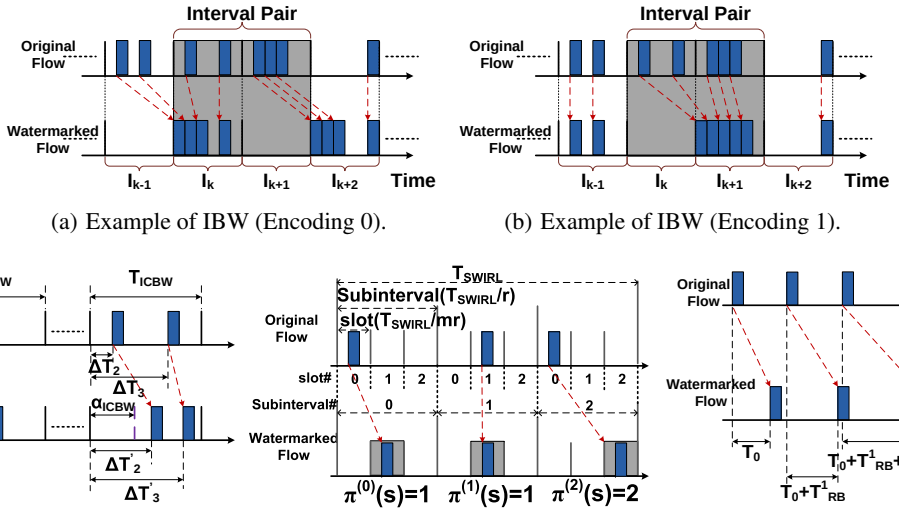
We roughly divide the network traffic into two types: (1) bulk transfer traffic that contains much more response packets than request packets. For example, an HTTP request that initiates a large file download will cause bulk transfer traffic. (2) interactive traffic that has similar number of request packets and response packets. For instance, an SSH session where the client sends a number of shell commands and receives relatively short responses (in terms of content) from the server will result in interactive traffic. Although, to our best knowledge, the types of traffic to which the existing traffic watermarking schemes can be applied are not explicitly documented, we assume that they can be employed to watermark both bulk transfer traffic and interactive traffic.

We discuss the selected features in bulk transfer traffic and interactive traffic individually for the ease of explaining the anomalies caused by different traffic watermarks. In bulk transfer traffic, we generally observe a large number of IPD samples and bursts of packets. Hence, BACKLIT employs them to perform the detection. In interactive traffic, we generally obtain many RRT samples triggered by requests. If the response to a request does not have many packets, the number of IPD samples may be limited and the burst size will not be determined by the TCP congestion control mechanism. In this case, BACKLIT relies only on RRT samples to carry out the detection. Since a network flow can be either bulk transfer traffic or interactive traffic or mixed traffic, BACKLIT selects suitable features for detection. For example, if an HTTP client sends many requests, each of which induces a short response, BACKLIT can obtain many RRT samples and then use them to carry out the detection. As another example, issuing a command like `ls -R /`, which asks a server to respond with a list of all files and directories on disk, will trigger many back-to-back packets from the server. In this case, BACKLIT can exploit the IPD samples and the burst sizes to perform the detection.

In the next three subsections, we use data from watermarked HTTP flows for bulk transfer traffic and watermarked SSH flows for interactive traffic to elaborate on the anomalies caused by different traffic watermarks on selected features. We adopt the traffic watermarking parameters given in the first row of Table 6 in Section 6. The experiments were conducted between two hosts with a minimum RTT of 0.184 seconds if there is no other specification.

4.1.1 RRT

RAINBOW and SWIRL cause two types of anomalies in RRTs.



(a) Example of IBW (Encoding 0). (b) Example of IBW (Encoding 1).

(c) Example of ICBW. (d) Example of SWIRL. (e) Example of RAINBOW.

Figure 2: Examples of IBW, ICBW, SWIRL and RAINBOW.

Table 2: Comparison of PNR, MFA, and BACKLIT.

Detection method	Timestamp required?	Assumes an ideal distribution?	Features	Mode	Requires cooperation from remote server ?
PNR [11]	Yes	Yes (Gaussian)	One-way packet delay	Active	Yes
MFA [13]	No	Yes (MMPP)	IPD	Passive	No
BACKLIT	No	No	IPD, RRT, burst size	Hybrid	No

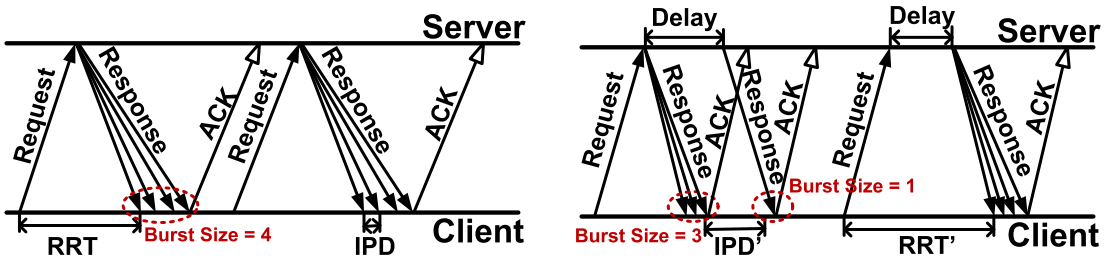
First, since most response packets will be delayed for some periods, the mean and the variance of disturbed RRTs will be larger than those of normal RRTs. Second, the absolute difference of consecutive RRTs, denoted as $|\Delta\text{RRT}|$, is equal to one or multiple T_{RB}s in traffic watermarked by RAINBOW and zero or multiple $\frac{T_{SWIRL}}{mr}$ in traffic watermarked by SWIRL if there is no noise. It is worth noting that the case of $|\Delta\text{RRT}| = 0$ in traffic watermarked by SWIRL appears only when consecutive responses are delayed for the same period. The probability of observing such case is extremely small because each response packet is delayed for a random period and the time when SWIRL observes a response is determined by the sending time of a request and the processing time of the server, both of which are random to SWIRL. BACKLIT leverages the second observation to detect RAINBOW and SWIRL because $|\Delta\text{RRT}|$ could be used to estimate the parameters employed in RAINBOW and SWIRL. Figure 4(a) illustrates the distribution of ΔRRT in normal SSH traffic and that in traffic watermarked by RAINBOW and SWIRL. The bin width is 0.005 seconds. These two distributions are quite different as most ΔRRTs in normal traffic are in the range of $[-0.005, 0.005]$, whereas the majority of ΔRRTs in watermarked traffic fall in the range of $[-0.05, 0.05]$.

IBW and ICBW will inflate RRTs. To clean packets in an interval, IBW delays them to the next interval. Depending on their locations in an interval, response packets will be delayed for a period in $(0, T_{IBW}]$. Similarly, ICBW delays response packets in selected intervals for a period in $(0, \alpha_{ICBW}]$. Figure 4(b) shows the distributions of RRTs from normal SSH traffic, traffic watermarked by IBW, and traffic watermarked by ICBW. It is clear that IBW and ICBW cause a set of abnormal RRTs that are larger than the normal RRTs.

4.1.2 IPD

All timing-based watermarking schemes change IPDs. Let ξ_{RTT} be the minimum RTT between BACKLIT and a remote server. BACKLIT employs IPDs larger than ξ_{RTT} to detect IBW and ICBW, because they introduce large delays to packets. In normal bulk transfer traffic, such large IPD samples are usually found from the intervals between two bursts of packets and such observation has been used to estimate a TCP flow's RTT [20]. When IBW and ICBW delay a batch of packets, abnormal IPDs will be observed between the first packet in the current burst and the last packet in the previous burst. Figure 4(c) illustrates IPDs that are larger than ξ_{RTT} in normal HTTP traffic and compares them with IPDs that are larger than ξ_{RTT} in traffic watermarked by ICBW and IBW. We can see that in normal traffic such IPDs are close to ξ_{RTT}, whereas in traffic watermarked by IBW and ICBW the IPDs are in the range of $(0.5, 0.7)$ seconds and $(0.35, 0.54)$ seconds, respectively. Note that the IPDs close to the lower bound (i.e., 0.5 and 0.35) are due to the delaying of packets within a burst, and those close to the upper bound (i.e., 0.7 and 0.54) originate from the delaying of the first packet in a burst.

BACKLIT uses IPDs lower than ξ_{RTT} to detect RAINBOW and SWIRL, because they introduce smaller delays. In normal traffic, such IPD samples usually come from packets sent back-to-back. It is worth noting that if two packets are sent back-to-back, then the normal IPD is close to ξ_{CAP}, the time required by the bottleneck on the network path to transmit a packet [21]. This observation has been widely used to estimate bottleneck capacity [21]. ξ_{CAP} is generally very small, for example, a 10Mbps link needs only 1.2 ms to transmit a 1500-byte packet. Since RAINBOW and SWIRL



(a) Normal IPD, RRT, and burst size.

(b) Disturbed IPD (IPD'), RRT (RRT'), and burst size.

Figure 3: Normal and disturbed inter-packet delay, request-response time, and burst size.

do not know the path from the encoder to the decoder in advance (i.e., the path is what they want to trace), they cannot set T_{RB} and $\frac{T_{SWIRL}}{mr}$ according to ξ_{CAP}. Moreover, if two packets are not sent back-to-back, say the interval between these two packets being larger than ξ_{CAP}, then the IPD will be equal to the original interval if there is no cross traffic [21]. Therefore, although RAINBOW and SWIRL introduce a small delay to the packets, they disturb the distribution of IPDs less than ξ_{RTT}. Figure 4(d) plots the distribution of such IPDs in normal HTTP traffic along with that in traffic watermarked by RAINBOW and SWIRL. In normal traffic most IPDs approximate ξ_{CAP}, whereas in the traffic watermarked by RAINBOW or SWIRL the majority of IPDs scatter to 0.01-0.06 seconds.

Though RAINBOW may let T_{RB} be larger than or equal to RTT, BACKLIT can still detect it, because in the former case it causes similar anomalies as IBW and ICBW do, and in the latter case RAINBOW leads to abnormal burst size, as will be discussed next.

4.1.3 *Burst size*

Although RAINBOW may decrease the number of disturbed IPDs that are lower than ξ_{RTT} by setting T_{RB} to ξ_{RTT}, doing so will cause suspicious burst sizes. In bulk transfer traffic, the size of a burst is approximately equal to the TCP sender's congestion window. Since RAINBOW divides the packets in each burst into several small groups, many one-packet bursts can be observed. If there is enough data to send in bulk transfer traffic, a TCP sender seldom dispatches one packet in an RTT, except for the case in which the sender is in the timeout state (i.e., retransmitting lost packets) or in the fast retransmit state with a very small congestion window. We propose methods to filter out biased samples due to packet losses in Section 4.3.

To facilitate the explanation, we assume that there are only two kinds of IPDs: one is equal to ξ_{RTT}, and the other is equal to ξ_{CAP}. Since a one-packet burst occurs if two consecutive IPDs are equal to ξ_{RTT} according to the algorithm for computing burst size in [19], the probability of observing a one-packet burst is equal to $1 - P_{\text{tw}}$ where P_{tw} is the probability that no consecutive IPDs are equal to ξ_{RTT}. Since RAINBOW will increase IPDs to ξ_{RTT} with probability 0.5, according to the Lemma 1 in the Appendix, we can get the probability of observing one-packet bursts as $1 - \frac{L}{2}/\binom{L}{\frac{L}{2}}$ from a sequence of L IPDs. This probability increases quickly with the length of L. In other words, when RAINBOW uses $T_{RB} = \xi_{RTT}$ to adjust the intervals between packets, we can observe many abnormal bursts that have only one packet.

Figure 4(e) shows the rate of one-packet bursts in normal HTTP flows and that in flows watermarked by RAINBOW with $T_{RB} \approx \xi_{RTT}$. The experiments were carried out between a host in Hong Kong and three PlanetLab nodes in Japan. It is obvious that the rate of one-packet bursts in normal flows is very small, whereas RAINBOW significantly increases it by letting $T_{RB} \approx \xi_{RTT}$.

4.2 Detection metrics and algorithm

BACKLIT adopts the anomaly detection approach and it marks a flow as a watermarked one if anomalies are found in any metric listed in Table 3. We elaborate on the metrics and the detection algorithm in Sections 4.2.1 and 4.2.2, respectively.

4.2.1 *Metrics*

Of the five metrics in Table 3, BACKLIT uses $\mathbb{R}$ and $\mathbb{T}$ to detect IBW and ICBW, and employs $\mathbb{Q}$, $\mathbb{D}$ and $\mathbb{Z}$ to discover RAINBOW and SWIRL. Let $\mathcal{F}$ represent a flow under inspection. We first describe how to calculate $\mathbb{R}$ and $\mathbb{T}$, because their computation procedures are similar, and then present how to compute $\mathbb{Q}$ and $\mathbb{D}$ for the same reason, and subsequently explain the calculation of $\mathbb{Z}$.

Let L denote a set of IPDs that are larger than ξ_{RTT} and $||L||$ be the number of such IPDs. We define a threshold $TH_{IPD}(k) = \mu_{IPD} + k\sigma_{IPD}$, where μ_{IPD} and σ^2_{IPD} are the mean and the variance of L in normal flows, respectively. Let $\mathbb{M}_{IPD}(k)$ denote the number of IPDs that are larger than $TH_{IPD}(k)$ in $\mathcal{F}$. We define $\mathbb{R}$ as the accumulative rate of IPDs larger than TH_{IPD}:

$$\mathbb{R} = \sum_{k=3}^{\varpi} \frac{\mathbb{M}_{IPD}(k)}{||L||}, \tag{1}$$

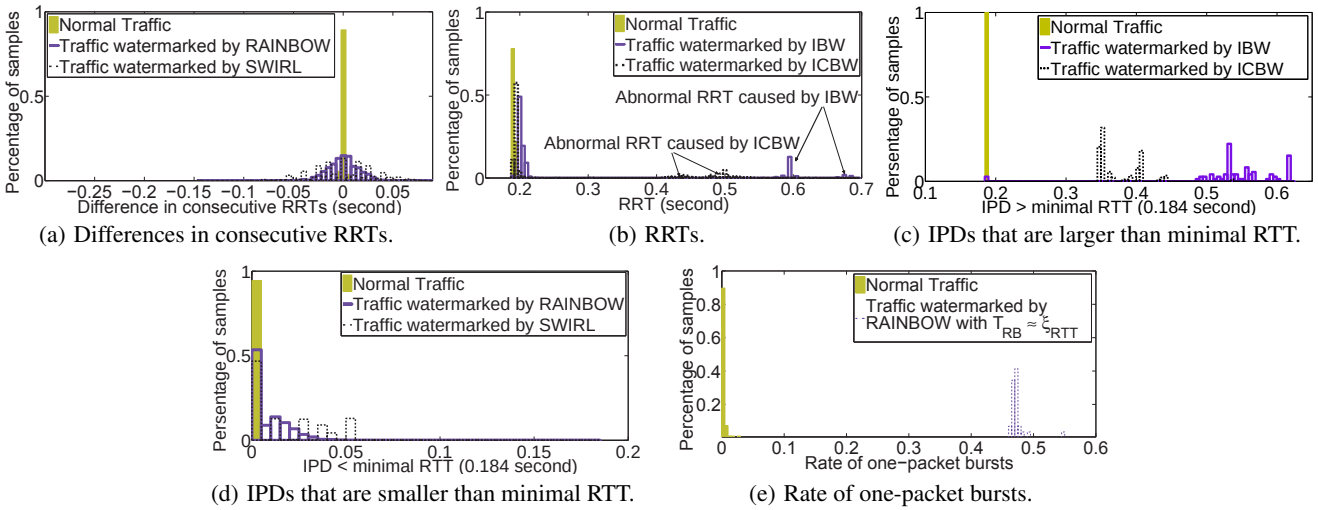
where ϖ is the minimum k that causes $\mathbb{M}_{IPD}(k) = 0$. More precisely, when k increases, the threshold TH_{IPD} becomes larger and $\mathbb{M}_{IPD}$ decreases. Since IPD is a limited value, there exists a k that leads to $\mathbb{M}_{IPD}(k) = 0$, assuming that $\sigma_{IPD} > 0$. We let k start from 3, because, according to the one-sided Chebyshev inequality [22], the probability of having a sample larger than such threshold is not greater than $\frac{1}{1+k^2} = 0.1$.

Similarly, let S represent a set of RRTs that are larger than ξ_{RTT} and $||S||$ be the number of such RRTs. We define a threshold $TH_{RRT}(k) = \mu_{RRT} + k\sigma_{RRT}$, where μ_{RRT} and σ^2_{RRT} are the mean and the variance of S in normal flows. Let $\mathbb{M}_{RRT}(k)$ be the number of RRTs that are larger than $TH_{RRT}(k)$ in $\mathcal{F}$. We define $\mathbb{T}$ as the accumulative rate of RRTs larger than TH_{RRT}:

$$\mathbb{T} = \sum_{k=3}^{\omega} \frac{\mathbb{M}_{RRT}(k)}{||S||}, \tag{2}$$

where ω is the minimum k that causes $\mathbb{M}_{RRT}(k) = 0$.

$\mathbb{Q}$ refers to the similarity between the distribution of IPDs that are lower than ξ_{RTT} in $\mathcal{F}$ and that in normal flows. $\mathbb{D}$ represents the similarity between the distribution of absolute ΔRRT (i.e., $|\Delta\text{RRT}|$) in $\mathcal{F}$ and that in normal flows. BACKLIT uses $\mathbb{Q}$ and $\mathbb{D}$ to detect RAINBOW and SWIRL, because they cause significant changes in the distribution of IPDs that are smaller than ξ_{RTT} and in the distribution of $|\Delta\text{RRT}|$.



(a) Differences in consecutive RRTs. (b) RRTs. (c) IPDs that are larger than minimal RTT.

(d) IPDs that are smaller than minimal RTT. (e) Rate of one-packet bursts.

Figure 4: Features in normal traffic and watermarked traffic.

Table 3: Metrics used by BACKLIT to detect timing-based traffic watermarks.

Feature	Bulk transfer traffic (e.g., HTTP)				Interactive traffic (e.g., SSH)				Notation
	IBW	ICBW	RAINBOW	SWIRL	IBW	ICBW	RAINBOW	SWIRL	
Accumulative rate of IPDs larger than TH_{IPD}	√	√							$\mathbb{R}$
Accumulative rate of RRTs larger than TH_{RRT}					√	√			$\mathbb{T}$
Distribution similarity of IPDs smaller than ξ_{RTT}			√	√					$\mathbb{Q}$
Distribution similarity of $\lvert\Delta\text{RRT}\rvert$s							√	√	$\mathbb{D}$
Rate of one-packet bursts			√ if $T_{RB} \approx \xi_{RTT}$						$\mathbb{Z}$

For $\mathbb{Q}$ and $\mathbb{D}$, we calculate the difference between histograms to quantify the similarity between the distribution in normal traffic and that in a flow to be inspected. $\mathbb{D}$ is used as an example to illustrate the computation of this similarity. The method comprises two steps:

1. Construct a histogram, denoted as $H_{\Delta RTT}$, for a sequence of $|\Delta\text{RRT}|$s. Let ΔRRT_{max} be the maximum value of $|\Delta\text{RRT}|$. Divide the range $[0, \Delta\text{RRT}_{max}]$ into M disjoint subregions of equal size, called *histogram bins*. Let $H_{\Delta RTT}(i)$ $(i = 1, \ldots, M)$ be the percentage of $|\Delta\text{RRT}|$s that fall into the ith histogram bin.
2. Compute the similarity between $H^{Normal}_{\Delta RTT}$ from normal flows and $H^{\mathcal{F}}_{\Delta RTT}$ from $\mathcal{F}$ as

$$\mathbb{D} = \sum_{i=1}^{\max\{M^{Normal}, M^{\mathcal{F}}\}} (H^{\mathcal{F}}_{\Delta RTT}(i) - H^{Normal}_{\Delta RTT}(i))^2, \quad (3)$$

where M^{Normal} and $M^{\mathcal{F}}$ are the number of bins in $H^{Normal}_{\Delta RTT}$ and $H^{\mathcal{F}}_{\Delta RTT}$, respectively. We set $H^{Normal}_{\Delta RTT}(j) = 0$ (or $H^{\mathcal{F}}_{\Delta RTT}(j) = 0$) if $j > M^{Normal}$ (or $j > M^{\mathcal{F}}$).

$\mathbb{Z}$ represents the ratio of the number of one-packet bursts to the total number of bursts. It is applied to bulk transfer traffic, because as explained in Section 4.1.3 RAINBOW will cause many one-packet bursts in bulk transfer traffic when $T_{RB} = \xi_{RTT}$.

Figures 5(a), 5(b), 5(c), and 5(d) show the distributions of $\mathbb{R}$ and $\mathbb{Q}$ in normal HTTP flows and those in HTTP flows watermarked by IBW, ICBW, RAINBOW and SWIRL using different parameter settings. Figures 6(a), 6(b), 6(c), and 6(d) show the CDFs of $\mathbb{T}$ and $\mathbb{D}$ obtained from normal SSH flows and those in watermarked SSH flows using different parameter settings. It can be seen that watermarked flows cause significant changes to $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{T}$, and $\mathbb{D}$, indicating that these metrics can be used to effectively detect watermarked flows.

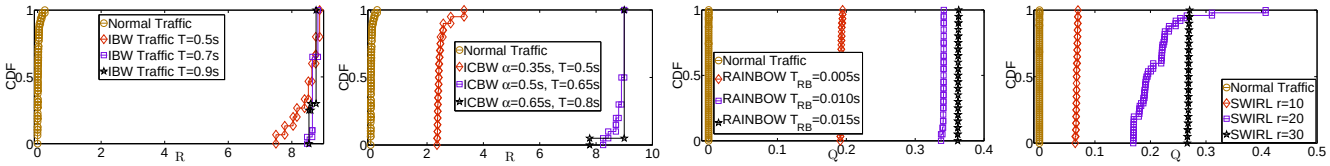
4.2.2 *Detection algorithm*

BACKLIT employs the one-class classifier [23] to capture anomalies in the five metrics, because it does not have watermarked flows when training the classifier. The one-class (OC) classifier learns from a single class of samples, which is labeled as the *target class*. The OC classifier identifies a boundary around the available data from the target class such that it includes as much data from the target class as possible, at the same time minimizing the chance of accepting *outliers* [23]. In other words, each datum is classified as a member of either the target class or the *outlier class*. The boundary is determined during the training with a given pre-reject ratio (P_R) which is the fraction of rejected training data for the sake of excluding possible noise in the training data [23]. If all the training samples belong to the target class, P_R can be regarded as a tolerable false positive rate [24].

When using the one-class classifier to detect traffic watermarks, the target (outlier) class corresponds to the class of non-watermarked (watermarked) network flows. We use the false positive rate and the detection rate to evaluate BACKLIT. The former is ratio of the number of non-watermarked flows that are mistakenly regarded as watermarked flows to the total number of non-watermarked flows. The latter is the ratio of the number of watermarked flows that are detected by BACKLIT to the total number of watermarked flows.

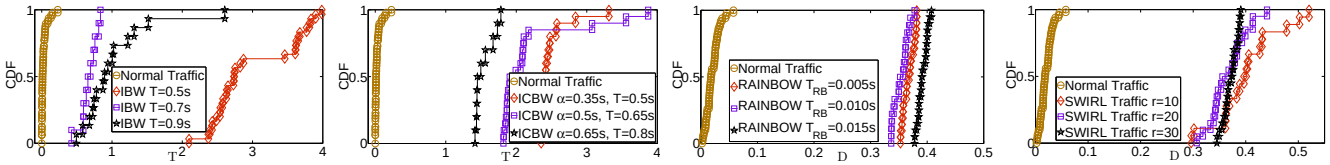
4.3 Normal profile creation

Before carrying out the detection, we collect data for constructing the normal profiles of RRT, IPD, and burst size before the de-



(a) Detect IBW through $\mathbb{R}$. (b) Detect ICBW through $\mathbb{R}$. (c) Detect RAINBOW through $\mathbb{Q}$. (d) Detect SWIRL through $\mathbb{Q}$.

Figure 5: The distributions of metrics $\mathbb{R}$ and $\mathbb{Q}$ in normal HTTP flows and those in HTTP flows watermarked by IBW, ICBW, RAINBOW and SWIRL using different settings.



(a) Detect IBW through $\mathbb{T}$. (b) Detect ICBW through $\mathbb{T}$. (c) Detect RAINBOW through $\mathbb{D}$. (d) Detect SWIRL through $\mathbb{D}$.

Figure 6: The distributions of metrics $\mathbb{T}$ and $\mathbb{D}$ in normal SSH flows and those in SSH flows watermarked by IBW, ICBW, RAINBOW and SWIRL using different settings.

ployment of traffic watermarking schemes. RRT samples are obtained by sending requests to the server and then recording the timestamp of a request's last packet (if a request consists of more than one packet) and the timestamp of the response's first packet. The response packet's acknowledgement number should be equal to the summation of the request packet's sequence number and its payload length. We compute IPDs from the response packets triggered by each single request. We calculate burst size in a flow using the method outlined in [19].

Since packet loss may lead to biased RRT, IPD and burst size samples, we exploit TCP's basic mechanism to filter out potentially biased samples that match one of the following criteria:

1. Samples containing out-of-order data packets. Nowadays packet reordering due to network elements is not prevalent, and most reordered packets are caused by packet retransmission [25].
2. Samples including data packets that follow three duplicate ACK packets. These ACK packets suggest that a TCP sender will use the fast retransmit/fast recovery mechanism to retransmit lost packets [26].
3. Samples comprising duplicate packets. This may happen when ACK packets are lost and then the server consequently retransmits the unacknowledged data packets.
4. IPDs that are close to or larger than the server's retransmission timeout (RTO) value and are followed by retransmitted packets. This rule is also applied to RRTs that are close to or larger than the summation of RTO and ξ_{RTT}. Even if there are not retransmitted packets, we can still infer whether a packet is retransmitted through the identification field in the IP header (IPID) or the estimated congestion window (cwnd) [27]. These methods are also employed to determine whether a one-packet burst is due to packet loss. The server's features, such as RTO, IPID, and cwnd can be measured using `TBIT`'s method [28].

5. IMPLEMENTATION

We implement IBW, ICBW, RAINBOW and SWIRL on Ubuntu Linux (kernel 2.6.35-22) using `iptables` (version 1.4.4) and `libnetfilter_queue` library (version 1.0.0). We add rules into `iptables`'s OUTPUT chains to hook outgoing TCP packets from the service under surveillance (e.g., HTTP, SSH). These packets are queued in the kernel, and our program acquires them by invoking `libnetfilter_queue`. By exploiting the head-of-blocking feature of the system's output queue, our program can delay a batch of packets by holding the first packet for a designated time period.

For IBW, we implement the interval selection function that chooses every second and third intervals to embed information [2]. For an ICBW interval, our algorithm randomly determines, with a probability of 0.5, whether packets within that interval will be delayed or not, because ICBW randomly selects intervals and then delays packets in selected intervals [3]. For SWIRL, we generate the permutation randomly, because Houmansadr et al. do not describe how to construct π in [7]. For RAINBOW, we follow [6] to prepare the delay added to each packet in advance and set the upper bound of T_0 to 50ms. Since in practice RAINBOW cannot predict when the next packet will arrive, we add pre-calculated delay to each packet independently. It is worth noting that our implementation does not allow reordered packets (i.e., if the adjusted IPD is less than zero, it will be set to zero), because frequent packet reordering is suspicious. Moreover, reordered packets cannot carry watermarks through stepping stones and anonymity networks, because the TCP/IP stack in a relay node will rearrange reordered packets and send *in-order* packets to the next node.

BACKLIT comprises a set of Python scripts and Matlab scripts that carry out the entire detection procedure. It instructs hosts to visit the target server through HTTP or SSH, extracts metrics from the observed traffic, and performs the detection. The detection module is based on the DD_tools Matlab toolbox 1.7.3 [29] and the PRTools Matlab toolbox 4.1.4 [30]. We employ the OC classifier based on support vector data description (SVDD) and detailed information about OC classifiers can be found in [23].

6. EVALUATION

6.1 Experiment settings

As shown in Table 4, most of the previous research on traffic watermarking used replayed or synthetic traffic for evaluation purposes [2, 6, 7, 12], with the exceptions of ICBW [3] and PNR [11]

which employed live traffic. The use of replayed or synthetic traffic is inadequate because they cannot legitimately represent TCP's behaviors, such as being responsive to network conditions and adapting its parameters to the changes. Moreover, replaying packets in either one or two directions of interactive traffic cannot mimic the real environment, because the RTT and network conditions on the path between two hosts in the trace may not be the same as those on the path between the two hosts replaying the traffic.

We used live HTTP and SSH flows to evaluate BACKLIT. The four watermarking schemes were deployed on a host in Hong Kong to embed watermarks into HTTP and SSH connections between the server and 12 PlanetLab nodes around the world, which are listed in Table 5.

For experiments using HTTP traffic, `curl` was run with the option `-max-time` on each PlanetLab node to download a large file from the server for 90 seconds. For experiments using SSH traffic, we implemented an SSH client based on `libssh2` [31] and ran it on the PlanetLab nodes. Besides using the default settings suggested in [2, 3, 6, 7], we also adopted parameters in line with the rules suggested by those traffic watermarks. Table 6 summarizes the parameter settings used in the experiments.

6.2 Evaluating BACKLIT's false positive rate

To evaluate BACKLIT using HTTP traffic, each PlanetLab node first downloaded a large file 60 times. Half of the traces were used to estimate ξ_{RTT} and train the OC classifier. The remaining traces were employed to evaluate BACKLIT's false positive rate. When evaluating BACKLIT using SSH, we executed a sequence of non-existent commands (i.e., input "a" and return). This is a simple way to test the presence of traffic watermarks. We observed that sending "a" and the return character triggers four packets from the server. The first two packets echoed "a" and the return character. The other two packets may contain the error message (e.g., `-bash: a: command not found`). Before executing a new command, BACKLIT slept for 100ms. We executed the commands for 60 times in each PlanetLab node. Half of the traces were used to train the OC classifier and the remaining traces were employed to evaluate BACKLIT's false positive rate.

We first set P_R to 0 and then 0.03. The former indicates that all normal samples are used to build the detection metrics profiles, and the latter means that all except one normal sample are employed, because the training data set consists of 30 samples. Table 7 shows the false positive rates for detection metrics $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{T}$, and $\mathbb{D}$. We did not observe any false positive in $\mathbb{Z}$.

For $P_R = 0$, none of the metrics produces false positives. For $P_R = 0.03$, the metrics may incorrectly identify at most one out of 30 normal samples as a watermarked one. One possible reason for the false positive is that the sample excluded from the training data due to $P_R = 0.03$ is not an outlier and should be kept. Another possible reason is that the boundary determined by the SVDD OC classifier is too conservative. This is supported by the fact that some misclassified samples from the testing data set were close to the boundary, whereas samples from watermarked flows were far away from the boundary. To remedy this problem, we added artificially generated outlier data to the training data set, labeled them as normal data and then re-trained the OC classifier. This approach helped remove the false positives. The DD_tools toolbox provides functions, such as `gendatout`, to generate such outliers based on existing data.

6.3 Evaluating BACKLIT's detection rate

To evaluate BACKLIT's detection rate, we activated each traffic watermarking scheme one by one and then asked each PlanetLab node to download the big file 20 times for acquiring watermarked HTTP traces. To collect watermarked SSH traces, BACKLIT instructed each PlanetLab node to enter the SSH commands 20 times. Setting $P_R = 0.03$ to exclude potential outliers in the training data set, BACKLIT discovered *all* the watermarked HTTP flows and only missed two watermarked SSH flows. One was an SSH flow watermarked by SWIRL using parameter setting 1 on the path from a PlanetLab node in France and the other was an SSH flow watermarked by IBW using parameter setting 1 on the path from a PlanetLab node in Japan (i.e., JP3).

The detection rates for IBW and ICBW are high, because they delay a set of packets for a long period (i.e., several hundred milliseconds) to facilitate the decoding of traffic watermarks [2,3]. By exploiting TCP's basic mechanism, we can filter out long IPDs resulting from packet loss. Such IPDs are similar to the delay introduced by IBW or ICBW and may cause false positives. Although MFA also exploits the long delay caused by IBW and ICBW, it did not mention whether such noises were removed and did not report the false positive rate [12]. We attribute the high detection rates for RAINBOW and SWIRL to the fact that they aggressively affect contiguous packets, although the introduced delay is small in comparison with IBW and ICBW. In particular, RAINBOW affects almost every packet, while SWIRL influences packets in consecutive short intervals.

After investigating the undetected watermarked flows, we found that both SSH connections were broken up after BACKLIT dispatched a few "a" commands. It may be caused by the instability of the PlanetLab nodes or network congestions. In the flow watermarked by IBW only part of the response packets were delayed and the period was short. The reason is that IBW only delays packets in selected intervals and if a packet is close to the end of a selected interval it will be delayed for a short period. Similarly, SWIRL only postponed some response packets and the $|\Delta\text{RRT}|$s affected by SWIRL were not obvious. It may be due to network congestions or cross traffic that could have delayed other response packets. Since existing timing-based traffic watermarking schemes usually require several hundreds or thousands of packets to embed traffic watermarks that can be successfully decoded [2, 3, 6, 7], it is difficult for them to trace short flows. On the other hand, BACKLIT can increase the sending rate of requests (i.e., in the active mode) to gain more prolonged RRT samples.

7. RELATED WORK

7.1 Timing-based traffic watermarks

Tracing network communications is motivated by a well-known security problem, namely detecting stepping-stone attacks. The pioneering work from Staniford-Chen and Heberlein used the thumbprint of a packets' content to correlate flows [32]. With the prevalence of encrypted payloads, content-based approaches have become less effective [32]. Zhang and Paxson exploited *invariant* traffic features, instead of the packet content [18], to detect stepping-stone attacks. Their method belongs to a class of passive approaches that do not perturb the traffic under surveillance. Besides the ON/OFF patterns employed in [18], many other features have been examined in recent papers [33,34], for example, packet count, inter-packet delay and the increasing pattern of TCP sequence number, to name a few. However, these approaches are vulnerable to normal time perturbations and *chaff* packets, and usually require thousands of packets to be effective [6].

Wang et al. proposed an original method that actively embeds watermarks into a flow by adjusting the timing of randomly selected packets [1]. Recently, more advanced flow watermarks have been proposed, which used sophisticated approaches to manipu-

Table 4: Experiment traffic used in this paper, IBW [2], ICBW [3], RAINBOW [6], SWIRL [7], PNR [11], and MFA [12].

Traffic	Synthetic SSH	Replayed SSH	Replayed HTTP	Live SSH	Live HTTP
Watermarking Schemes	IBW, ICBW	RAINBOW, SWIRL			ICBW
Detection Mechanism		MFA	MFA	PNR†, **BACKLIT**	**BACKLIT**

† PNR does not indicate the traffic type, which might be live SSH traffic as the experiment was conducted between stepping stones.

Table 5: The 12 PlanetLab nodes used in the experiments.

Country	IP	Country	IP	Country	IP	Country	IP	Country	IP	Country	IP
FR	132.227.62.25	CH	130.92.70.254	SG 1	137.132.80.106	SG 2	203.30.39.238	FI	193.166.167.5	JP 1	133.68.253.243
JP 2	202.23.159.52	JP 3	150.65.32.68	NZ	132.181.10.57	DE	141.20.103.211	BR	200.17.202.195	US	208.94.63.193

Table 6: Parameters used for each type of traffic watermark.

	IBW	ICBW	SWIRL	RAINBOW
Setting 1	$T_{IBW} = 0.5$s	$\alpha_{ICBW} = 0.35$s, $T_{ICBW} = 0.5$s†	$T_{SWIRL} = 2$s, $m = 5, r = 30$	$T_{RB} = 0.005$s
Setting 2	$T_{IBW} = 0.7$s	$\alpha_{ICBW} = 0.5$s, $T_{ICBW} = 0.65$s	$T_{SWIRL} = 2$s, $m = 5, r = 20$	$T_{RB} = 0.01$s
Setting 3	$T_{IBW} = 0.9$s†	$\alpha_{ICBW} = 0.65$s, $T_{ICBW} = 0.8$s	$T_{SWIRL} = 2$s, $m = 5, r = 10$	$T_{RB} = 0.015$s

† The experiment settings used for evaluating MFA [12].

Table 7: False positive rates obtained from the PlanetLab nodes for HTTP and SSH traffic.

Country	FR		CH		SG 1		SG 2		FI		JP 1		JP 2		JP 3		NZ		DE		BR		US	
P_R	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03	0	0.03
$\mathbb{R}$ (HTTP)	0	0.03	0	0.03	0	0.03	0	0	0	0	0	0.03	0	0.03	0	0	0	0	0	0	0	0	0	0
$\mathbb{Q}$ (HTTP)	0	0.03	0	0	0	0.03	0	0.03	0	0	0	0	0	0.03	0	0	0	0.03	0	0	0	0.03	0	0.03
$\mathbb{T}$ (SSH)	0	0	0	0.03	0	0	0	0.03	0	0	0	0.03	0	0	0	0	0	0	0	0.03	0	0	0	0
$\mathbb{D}$ (SSH)	0	0	0	0	0	0.03	0	0.03	0	0.03	0	0	0	0.03	0	0	0	0.03	0	0.03	0	0	0	0

late packet timing information [2, 3, 6, 7] or the traffic rate [35]. Houmansadr et al. [6] classified watermarking schemes into *blind* and *non-blind*, depending on whether the decoder has *a priori* information about the flow that is being watermarked. Most existing watermarking techniques belong to the blind scheme category [1–3, 5, 35], whereas RAINBOW [6] is a non-blind scheme.

7.2 Countermeasures

Although various traffic watermarks have been designed, only a few approaches have been proposed to detect them. PNR [11], a pioneering work from Peng et al., employs inflated one-way packet delays to detect the traffic watermarks proposed in [1] and estimates its parameters. However, PNR adds a timestamp to each packet and has to handle the synchronization issues. BACKLIT does not insert information into the packets or involve the cooperation of a remote server, although this can help detection. Instead, BACKLIT exploits TCP's timing features for the detection.

Kiyavash et al. proposed an interesting detection scheme, MFA [12], which exploits the observation that when the same watermarks are embedded to multiple flows simultaneously, there will be abnormally long idle periods in the aggregated traffic. However, MFA can be evaded if the encoder randomizes the location of the watermarks or uses different watermarks on different flows, as reported in their follow-on paper [15]. Moreover, MFA requires multiple flows for successful detection and assumes that normal traffic follows the Markov-modulated Poisson process model. BLACKLIT can detect watermarks in a single flow and does not make any assumption about the distribution of normal traffic. Moreover, BLACKLIT is not affected by the random location of watermarks.

Yu et al. devised an invisible throughput-based traffic watermarking scheme that uses the direct-sequence spread spectrum theory to hide the watermarks [35]. We proposed an approach exploiting the features of Pseudo-Noise (PN) codes to detect such watermarks [36].

8. CONCLUSION

Traffic watermarking is important to many network security and privacy applications, because it can correlate network flows observed at different locations quickly and accurately. The state-of-the-art timing-based traffic watermarks can not only survive adversarial network conditions but also evade existing detection methods. However, we show in this paper for the first time that all of them will disturb the intrinsic timing features in TCP flows and propose a novel system, BACKLIT, to detect them. Our extensive empirical evaluation has shown that BACKLIT can detect the state-of-the-art traffic watermarks with high detection rate and low false positive rate. By exploiting these TCP features, BACKLIT can potentially be extended to detect other timing-based traffic watermarks. In future work, we will differentiate these traffic watermarks based on different anomalies and exploit the disturbed features to estimate their parameters.

Acknowledgments

We thank the anonymous reviewers and Dr. John McDermott for their very useful suggestions and Michael Cumming for editing the paper. This work is partially supported by a grant (G-U386) from The Hong Kong Polytechnic University. This material is based upon work supported in part by the National Science Foundation under grant no. 0831300, the Department of Homeland Security under contract no. FA8750-08-2-0141, the Office of Naval Research under grants no. N000140710907 and no. N000140911042. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Department of Homeland Security, or the Office of Naval Research.

9. REFERENCES

[1] X. Wang and D. Reeves, "Robust correlation of encrypted attack traffic through stepping stones by manipulation of interpacket delays," in *Proc. ACM CCS*, 2003.

[2] Y. Pyun, Y. Park, X. Wang, D. Reeves, and P. Ning, "Tracing traffic through intermediate hosts that repacketize flows," in *Proc. IEEE INFOCOM*, 2007.

[3] X. Wang, S. Chen, and S. Jajodia, "Network flow watermarking attack on low-latency anonymous communication systems," in *Proc. IEEE Symp. Security and Privacy*, 2007.

[4] D. Ramsbrock, X. Wang, and X. Jiang, "A first step towards live botmaster traceback," in *Proc. RAID*, 2008.

[5] X. Wang, S. Chen, and S. Jajodia, "Tracking anonymous peer-to-peer voip calls on the Internet," in *Proc. ACM CCS*, 2005.

[6] A. Houmansadr, N. Kiyavash, and N. Borisov, "RAINBOW: A robust and invisible non-blind watermark for network flows," in *Proc. NDSS*, 2009.

[7] A. Houmansadr and N. Borisov, "SWIRL: A scalable watermark to detect correlated network flows," in *Proc. NDSS*, 2011.

[8] V. Shmatikov and M. Wang, "Timing analysis in low-latency mix networks: attacks and defenses?" in *Proc. ESORICS*, 2006.

[9] H. Daginawala and M. Wright, "Studying timing analysis on the Internet with SubRosa," in *Proc. PET*, 2008.

[10] W. Wang, M. Motani, and V. Srinivasan, "Dependent link padding algorithms for low latency anonymity systems," in *Proc. ACM CCS*, 2008.

[11] P. Peng, P. Ning, and D. Reeves, "On the secrecy of timing-based active watermarking trace-back techniques," in *Proc. IEEE Symp. Security and Privacy*, 2006.

[12] N. Kiyavash, A. Houmansadr, and N. Borisov, "Multi-flow attacks against network flow watermarking schemes," in *Proc. USENIX Security*, 2008.

[13] A. Hernandez and E. Magana, "One-way delay measurement and characterization," in *Proc. IEEE ICNS*, 2007.

[14] N. Macfadyen, "Traffic characterisation and modelling," *BT Technology Journal*, vol. 20, no. 3, 2002.

[15] A. Houmansadr, N. Kiyavash, and N. Borisov., "Multi-flow attack resistant watermarks for network flows," in *Proc. IEEE ICASSP*, 2009.

[16] W. John and S. Tafvelin, "Analysis of Internet backbone traffic and header anomalies observed," in *Proc. ACM/USENIX IMC*, 2007.

[17] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proc. USENIX SECURITY*, 2004.

[18] Y. Zhang and V. Paxson, "Detecting stepping stones," in *Proc. USENIX Security*, 2000.

[19] S. Shakkottai, N. Brownlee, and k. claffy, "A study of burstiness in TCP flows," in *Proc. Passive and Active Measurement Conf.*, 2005.

[20] H. Jiang and C. Dovrolis, "Passive estimation of TCP round-trip times," *ACM Computer Commun. Review*, 2002.

[21] C. Dovrolis, P. Ramanathan, and D. Moore, "Packet dispersion techniques and a capacity-estimation methodology," *IEEE/ACM Trans. Networking*, vol. 12, no. 6, 2004.

[22] C. Therrien and M. Tummala, *Probability for Electrical and Computer Engineers*. CRC, 2004.

[23] D. Tax, "One-class classification; concept-learning in the absence of counter-examples," Ph.D. dissertation, Delft University of Technology, 2001.

[24] R. Perdisci, G. Gu, and W. Lee., "Using an ensemble of one-class SVM classifiers to harden payload-based anomaly detection systems," in *Proc. IEEE ICDM*, 2006.

[25] L. Gharai, C. Perkins, and T. Lehman, "Packet reordering, high speed networks and transport protocol performance," in *Proc. IEEE ICCCN*, 2004.

[26] M. Allman, V. Paxson, and W. Stevens, "TCP congestion control," RFC 2581, April 1999.

[27] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, "Measurement and classification of out-of-sequence packets in a tier-1 IP backbone," *IEEE/ACM Trans. Networking*, vol. 15, no. 1, 2007.

[28] A. Medina, M. Allman, and S. Floyd, "Measuring the evolution of transport protocols in the Internet," *ACM Computer Communication Review*, 2005.

[29] D. Tax, "DDtools, the data description toolbox for Matlab (version 1.5.3)," June 2009.

[30] "PRTools: The Matlab toolbox for pattern recognition," http://www.prtools.org/, 2008.

[31] "libssh2," http://www.libssh2.org/, 2011.

[32] L. Heberlein and S. Staniford-Chen, "Holding intruders accountable on the Internet," in *Proc. IEEE Symp. Security and Privacy*, 1995.

[33] A. Blum, D. Song, and S. Venkataraman, "Detection of interactive stepping stones: Algorithms and confidence bounds," in *Proc. RAID*, 2004.

[34] B. Coskun and N. Memon, "Efficient detection of delay-constrained relay nodes," in *Proc. ACSAC*, 2007.

[35] W. Yu, X. Fu, S. Graham, D. Xuan, and W. Zhao, "DSSS-based flow marking technique for invisible traceback," in *Proc. IEEE Symp. Security and Privacy*, 2007.

[36] X. Luo, J. Zhang, R. Perdisci, and W. Lee, "On the secrecy of spread-spectrum flow watermarks," in *Proc. ESORICS*, 2010.

APPENDIX

LEMMA 1. *Given a sequence of L observations of IPDs, where L_r IPDs are equal to ξ_{RTT} and the remaining IPDs are equal to ξ_{CAP}, the probability that there will be no consecutive $IPDs = \xi_{RTT}$s is:* $P_{\mathfrak{w}} = \frac{\binom{L-L_r+1}{L_r}}{\binom{L}{L_r}}$.

PROOF. We define a run of $IPDs = \xi_{RTT}$ (or $IPDs = \xi_{CAP}$) as a sequence of length $l \geq 1$ of consecutive IPDs that have a value equal to ξ_{RTT} (or ξ_{CAP}). If there are no consecutive $IPDs = \xi_{RTT}$, then there are L_r runs of $IPDs = \xi_{RTT}$ having length $l = 1$, and the number of runs of $IPDs = \xi_{CAP}$ is equal to $L_r - 1$, $L_r + 1$, or L_r, because the runs of $IPDs = \xi_{RTT}$ and the runs of $IPDs = \xi_{CAP}$ must alternate.

If the value is equal to $L_r - 1$, then the first observation is $IPD = \xi_{RTT}$. If it is equal to $L_r + 1$, then the first observation is $IPD = \xi_{CAP}$. If the value is equal to L_r, either $IPD = \xi_{RTT}$ or $IPD = \xi_{CAP}$ could be the first observation. Since the number of combinations of putting $(L - L_r)$ $IPDs = \xi_{CAP}$ into L_r runs is $\binom{L-L_r-1}{L_r-1}$, the probability that there are L_r runs of $IPDs = \xi_{RTT}$ is equal to $\frac{\binom{L-L_r-1}{L_r}+\binom{L-L_r-1}{L_r-2}+2*\binom{L-L_r-1}{L_r-1}}{\binom{L}{L_r}} = \frac{\binom{L-L_r+1}{L_r}}{\binom{L}{L_r}}$. □