Chapter 19

REAL-TIME COVERT TIMING CHANNEL DETECTION IN NETWORKED VIRTUAL ENVIRONMENTS

Anyi Liu, Jim Chen and Harry Wechsler

Abstract Despite extensive research on malware and Trojan horses, covert channels are still among the top computer security threats. These attacks, which are launched using specially-crafted content or by manipulating timing characteristics, transmit sensitive information to adversaries while remaining undetected. Current detection approaches typically analyze deviations from legitimate network traffic statistics. These approaches, however, are not applicable to highly dynamic, noisy environments, such as cloud computing environments, because they rely heavily on historical traffic and tedious model training. To address these challenges, we present a real-time, wavelet-based approach for detecting covert timing channels. The novelty of the approach comes from leveraging a secure virtual machine to mimic a vulnerable virtual machine. A key advantage is that the detection approach does not require historical traffic data. Experimental results demonstrate that the approach exhibits good overall performance, including a high detection rate and a low false positive rate.

Keywords: Covert timing channels, detection, virtual environments

1. Introduction

Covert timing channels are among the most prevalent computer security attacks. These attacks exfiltrate sensitive information and credentials, such as secret keys and passwords, by manipulating the timing or the ordering of network events [6]. A successful covert timing channel leaks sensitive information to external attackers without triggering alerts even in well-protected networks.

Most covert timing channel detection approaches use signatures to detect known channels [5] or consider anomalous deviations from legit-

G. Peterson and S. Shenoi (Eds.): Advances in Digital Forensics IX, IFIP AICT 410, pp. 273–288, 2013.

imate network traffic to detect unknown channels [3, 5, 6, 17]. These approaches, however, have at least two limitations. First, their effectiveness depends on the availability of a sufficient amount of legitimate (or attack-free) traffic to construct attack signatures and accurate models of legitimate traffic. Unfortunately, in a networked virtual environment, where virtual machines are interconnected, legitimate traffic is either hard to obtain or contains noise due to the imprecise timekeeping mechanism used in virtual machines [19]. This greatly reduces the effectiveness of signature-based and anomaly-based approaches for covert timing channel detection.

The second limitation is that most existing approaches were designed to detect covert timing channels that transmit information at a high rate to achieve high bandwidth. However, an attacker can deliberately extend the duration of the covert transmission process. This renders the timing patterns of covert timing channel traffic almost indistinguishable from those of legitimate network traffic. Thus, existing approaches are generally unable to detect slow covert timing channels (e.g., the channels described in [7, 13]).

This paper presents a new wavelet-based approach for detecting covert timing channels in networked virtual environments, where no historical data pertaining to legitimate traffic is available. The detection approach employs distance metrics between outbound traffic generated by two virtual machines, a suspicious virtual machine and a benign virtual machine. More specifically, the metrics use a discrete wavelet-based multiresolution transformation (DWMT) to measure the variability of timing differences at all decomposition scales. To our knowledge, this is the first online detection approach that operates by quantitatively measuring the distance between two network flows.

The wavelet-based approach is evaluated using a series of experiments focused on detecting several covert timing channels, including their slow variations. The robustness of the detection approach in the presence of noise is also investigated. Experimental results demonstrate that the wavelet-based approach is very effective at detecting covert timing channels in real time, despite efforts to make them slow and stealthy.

2. Background

Covert timing channels have been the subject of much research. Berk, *et al.* [3] implemented a simple binary covert timing channel based on the Arimoto-Blahut algorithm [4], which computes the input distribution that maximizes channel capacity. Cabuk [5] was the first to describe an IP covert timing channel (IPCTC) and a more advanced traffic replay

covert timing channel (TRCTC). Shah, *et al.* [17] developed JitterBug, a keyboard device that slowly leaks information typed by a user over the Internet. Giffin, *et al.* [8] showed that low-order bits of TCP timestamps can be exploited to create a covert timing channel due to the shared statistical properties of timestamps and packet timing.

Covert timing channels can also be used to trace suspicious traffic. For example, Wang, *et al.* [20] leveraged well-designed inter-packet delays to trace VoIP traffic. Their recent work [16] utilizes watermarked network traffic to trace covert timing channels back to botmasters. Gianvecchio, *et al.* [7] and Liu, *et al.* [13] designed model-based covert channel encoding schemes that seek to achieve both undetectability and robustness.

A number of covert timing channel detection methods have been developed. Peng, *et al.* [14] showed that the Kolmogorov-Smirnov test is effective at detecting covert timing channels that manipulate inter-packet delays. Cabuk [5] investigated a regularity-based approach for detecting covert timing channels. In particular, he proposed an ϵ-similarity metric to measure the proportion of similar inter-packet delays. The limitation of the ϵ-similarity metric is that it only targets IPCTCs; it is not general enough to detect other covert timing channels. Berk, *et al.* [3] employed a simple mean-max ratio test to detect binary and multi-symbol covert timing channels. However, the mean-max ratio test assumes that legitimate inter-packet delays follow a normal distribution, which is often not true for real-world traffic.

Gianvecchio and Wang [6] proposed an effective entropy-based method for detecting covert timing channels. However, our approach has three advantages over this method. First, as an essential step for computing patterns of length m, the corrected conditional entropy metric used by Gianvecchio and Wang has quadratic time complexity, while our approach has linear time complexity. Second, the corrected conditional entropy metric cannot detect stealthy covert timing channels, while our wavelet-based distance metric can detect a variety of covert timing channels, including their stealthy versions. Third, our approach can be used in an online configuration and is well suited to virtual environments.

Jing and Wang [10] developed a wavelet-based method for measuring time distortion in low latency anonymous networks. This method is closely related to our approach. However, our approach has two advantages. First, unlike the time distortion metric, our wavelet-based distance metric can clearly differentiate between slow versions of covert timing channels. Second, our approach is specifically designed to detect covert timing channels, which have more stealthy timing characteris-

tics than the timing distortion of packet transformation introduced by anonymous networks.

Zhang *et al.* [22] have proposed online covert timing channel prevention mechanisms that mitigate information leakage. Although their mechanisms are efficient at identifying the upper and lower bounds of information leakage as a function of elapsed time, they are unable to detect covert timing channels.

3. Wavelet-Based Detection Approach

This section describes a wavelet-based metric for quantitatively measuring the distance between the inter-packet delays (IPDs) of a legitimate flow and a covert timing channel flow. The section begins by describing the covert timing channel detection approach. Next, it discusses how the timing distance between two outbound network flows can be measured using the variables derived from a wavelet-based multi-resolution transformation (DWMT). Finally, it formulates the metric for measuring the timing distance between network flows.

Given an inbound network flow I, the problem of measuring the timing distance between two outbound flows, O_1 and O_2, can be formulated as follows. The inbound flow I contains $K > 0$ packets $\langle p_{i,1}, \ldots, p_{i,K-1} \rangle$. In response to I, a virtual machine VM1 generates O_1 containing $M > 0$ packets $\langle p_{o_1,1}, \ldots, p_{o_1,M-1} \rangle$ and a virtual machine VM2 generates O_2 containing $N > 0$ packets $\langle p_{o_2,1}, \ldots, p_{o_2,N-1} \rangle$. Since the packets in O_i were generated by a virtual machine in response to I, the packets in I and O_i are segmented based on their request/response relationship. Specifically, for the j^{th} inbound segment $I^j = \langle p^j_{i,1}, \ldots, p^j_{i,m} \rangle$, the responding outbound segment in O_i is defined as $O^j_i = \langle p^j_{o_i,1}, \ldots, p^j_{o_i,o} \rangle$. The notation $t^j_{(o_i,k)}$ is used to represent the timestamp of the k^{th} packet in the j^{th} segment of O_i. Since the length of I is much greater than that of O_i, O_i can be further aggregated into w aggregated segments.

3.1 Covert Timing Channel Detection

Figure 1 shows example inbound and outbound flows. O^{i-2}_2 is the responding outbound segment of I^{i-2} in flow O_2. For $O_{1,i}$, the packets in segment t_i are denoted as $\langle p_{o_1,0}, \ldots, p_{o_1,m-1} \rangle (m > 0)$. Similarly, for $O_{2,i}$, the packets in the segment are denoted as $\langle p_{o_2,0}, \ldots, p_{o_2,n-1} \rangle (n > 0)$, where $n \approx m$. The timestamps of the j^{th} packet in $O_{1,i}$ and $O_{2,i}$ are denoted as $t^j_{(O_1,i)}$ and $t^j_{(O_2,i)}$, respectively.

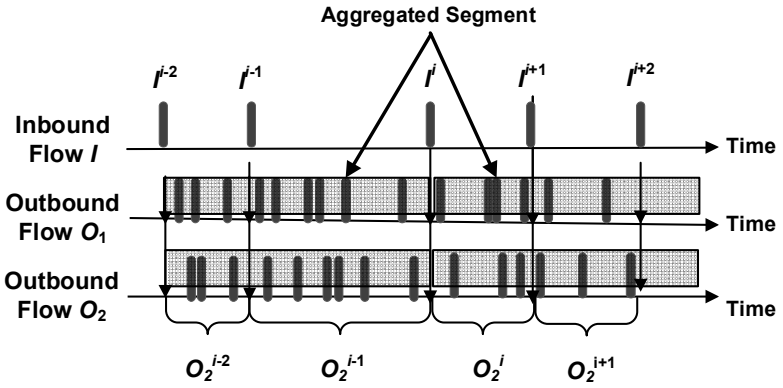

Figure 1. Inbound and outbound flows.

3.2 Timing Distance

The assumption underlying the timing distance metric is that the timing patterns of outbound legitimate traffic are similar. Therefore, by characterizing the timing patterns of network flows, it is possible to efficiently measure the timing distance between a legitimate flow and a covert timing channel flow. The measurement is performed using a discrete wavelet-based multi-resolution transformation (DWMT) [1]. The DWMT, which has been widely used in signal processing and anomaly detection [10], has at least three prominent features. First, DWMT provides multi-resolution analysis, which facilitates the examination of a sequence of data at different scales. Second, DWMT allows feature localization in that it provides information about the characteristics of a signal and "approximately" where they occur in time. Third, DWMT supports online analysis, i.e., online comparisons of the differences between two flows.

The DWMT takes a sequence of data as input and transforms the sequence into a number of wavelet coefficient sequences. Specifically, the l-level DWMT takes a sequence of IPDs and transforms the sequence into: (i) l-wavelet detailed coefficient vectors at different scales (CD_i, where $1 \leq i \leq l$); and (ii) a low-resolution approximate vector (CA_l, where $1 \leq i \leq l$). For the j^{th} segment of O_i, the wavelet detailed coefficients vector at scale l can represented as:

$$V(i,j,l) = \langle CD_l^j(o_i,1), ..., CD_l^j(o_i,N_j)\rangle$$

where $N_j = n_j \times 2^{-j}$ is the number of wavelet coefficients at scale j, and $c_{l,k} = c_{l-1,2k} + c_{l-1,2k+1}$ for $l \geq 0$.

Our wavelet-based distance (WBD) must satisfy three design goals. First, the WBD between two legitimate flows should be small. Second, the WBD between a legitimate flow and a covert timing channel flow should be different enough to be detectable. Third, the WBD should be able to differentiate between a regular covert timing channel and a stealthy covert timing channel.

To achieve these goals, we define three derived vectors based on the coefficient vector $V(i,j,l)$ at scale l ($l \geq 0$):(i) intra-flow vector ($intraFV$); (ii) inter-flow vector ($interFV$); and (iii) Kullback-Leibler divergence (KLD) vector.

We define:

$$\begin{aligned} intra(i,j,l) = \langle CD_l^j(O_i,1) &- CD_l^j(O_i,0),...,CD_l^j(O_i,N_j) \\ &- CD_l^j(O_i,N_{j-1})\rangle \end{aligned}$$

which reflects the fluctuating characteristics between adjacent coefficients in a coefficient vector at scale j of one flow O_i. $intraFV(j,l)$ is defined as the Euclidean distance between $intra(1,j,l)$ and $intra(2,j,l)$:

$$intraFV(j,l) = dist(intra(1,j,l), intra(2,j,l)).$$

Similarly, we define:

$$interFV(j,l) = dist(V(1,j,l), V(2,j,l))$$

as the Euclidean distance between the two coefficient vectors $V(1,j,l)$ and $V(2,j,l)$, which characterizes the deviation between two wavelet coefficients for the same segment j at the same scale j.

The Kullback-Leibler divergence (KLD) has been used to measure the distance between two probability distributions $p_1(x)$ and $p_2(x)$ [11]. From an information theory perspective, KLD measures the expected number of extra bits required to code samples from $p_1(x)$ when using a code based on $p_2(x)$. The KLD for the probability distributions $p_1(x)$ and $p_2(x)$ is given by:

$$KLD(p_1(x), p_2(x)) = \sum_{i=0}^{|x|} p_1(x) \log \frac{p_1(x)}{p_2(x)}. \quad (1)$$

In order to calculate the KLD between two wavelet coefficient vectors at scale j, it is necessary to obtain the probability distribution of $V(i,j,l)$, i.e., $p(V(1,j,l))$. To obtain $p(V(1,j,l))$, the term $V(i,j,l) = \langle CD_l^j(o_i,1),\ldots,CD_l^j(o_i,N_j)\rangle$ with numeric coefficients is converted to a vector of symbols $\tilde{S}_i = \langle \alpha_1,\ldots,\alpha_l\rangle$. Then, $p(V(i,j,l))$ is calculated

based on $\tilde{S}_i$. The effectiveness of converting $V(i,j,l)$ to $\tilde{S}_i$ depends on a function F that maps $CD_l^j(o_i,m)$ to an alphabet $\mathcal{A} = \alpha_1, \ldots, \alpha_m$, where the soundness of translation from $CD_l^j(o_i,m)$ to α_m is given by:

$$F(CD_l^j(o_i,m)) = \alpha_m$$

if and only if $\beta_{j-1} \leq \alpha_m \leq \beta_j (1 \leq i \leq l; 1 \leq j \leq m)$.

To facilitate effective conversion, the data discretization scheme of SAX [12] is used to assign wavelet coefficients to k equiprobable regions. Each continuous coefficient value that falls in a region maps to a unique symbol α_i $(1 \leq i \leq \kappa)$. The KLD defined in Equation (1) for $V(1,j,l)$ and $V(2,j,l)$ now becomes:

$$KLD(j,l) = KLD(p(V(1,j,l)), p(V(2,j,l))) = \sum_{i=0}^{k} p(\tilde{S}_1) \log \frac{p(\tilde{S}_1)}{p(\tilde{S}_2)}.$$

A tradeoff is involved when choosing the optimal size κ of alphabet $\mathcal{A}$. A large value κ keeps more information about the distribution of continuous data, but it may generate too many false alarms. In contrast, a small value of κ keeps less information about the distribution, but it may be difficult to detect slow and stealthy attacks. We choose $\kappa = 10$ to retain the ability to measure the deviation of malicious traffic. All the WBD values calculated in this paper use $\kappa = 10$, except when stated otherwise.

Given $intraFV$, $interFV$ and KDL at scale l, the wavelet-based distance (WBD) between O_1 and O_2 for all scales is defined by:

$$WBD(O_1,O_2) = \sum_{j=1}^{m} \left(\sum_{l=0}^{m} intraFV(j,l) \times \sum_{l=0}^{m} interFV(j,l) \times \sum_{l=0}^{m} KLD(j,l) \right)^2.$$

WBD essentially summarizes the divergence of intra-flow, inter-flow and KLD between two network flows. A large value of WBD indicates a significant difference between two flows while a small value of WBD implies that the two flows are similar.

4. Implementation

Figure 2 shows the architecture of the detection system. It has three main components: (i) a traffic filter, which identifies the packets that are

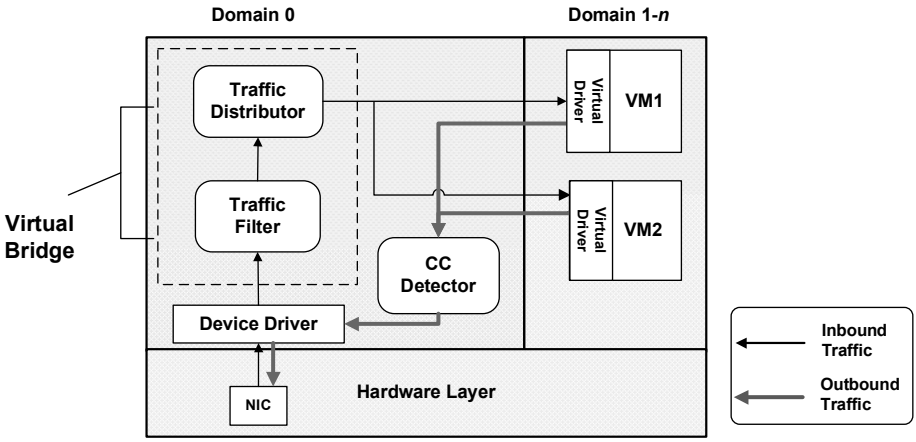

Figure 2. Detection system architecture.

sent to the suspicious virtual machine; (ii) a traffic distributor, which forwards inbound packets to the suspicious virtual machine and the benign virtual machine; and (iii) a covert channel (CC) detector, which uses the timing distance metric to calculate the timing distance between the suspicious virtual machine and the benign virtual machine.

Our covert channel detection system is implemented in C on top of Xen [2]. The traffic filter and traffic distributor are implemented as a virtual bridge. To emulate a malicious program that exfiltrates insider information, we modified the source code of `vsftpd v2.3.4` running on a Real Time Application Interface (RTAI) for Linux. The modified `vsftpd` code incorporates a covert timing channel encoder that generates a timing delay before sending a packet to a client. The covert timing channel encoder is implemented in C and inline assembly code that invokes instructions to the read timestamp counter (RDTSC) of a CPU. The RDTSC instruction was chosen because it has excellent resolution and requires low overhead for keeping time information [7].

5. Evaluation

The effectiveness of our approach was evaluated using the WBD metric to detect a number of covert timing channels. The covert timing channels, which are shown in Table 1, include both active and passive channels. The following sections describe the detection methods and the detection results for each type of covert timing channel.

Table 1. Covert timing channels used in the evaluation.

Name	**Attack Description**	**Active/ Passive**	**Detectable?**
IP Channel (IPCTC) [5]	Transmit `1`-bit or `0`-bit by choosing to send or not send a packet during time interval w	Passive	Yes
Time-Replay Channel (TRCTC) [5]	Transmit `1`-bit or `0`-bit by choosing historical legitimate IPDs as additional input from different baskets	Active	Yes
Botnet Traceback Watermark (BTW) [16]	Inject modified control text	Passive	Yes
JitterBug [17]	Operate small delays in keystrokes to affect the original IPDs	Passive	Yes

5.1 Detection Methods

Four broad classes of measurements were used to detect covert timing channels: (i) statistical tests; (ii) timing distortion metric for measuring low latency anonymous networks [10]; (iii) corrected conditional entropy metric [6]; and (iv) our wavelet-based distance (WBD) metric.

Four statistical tests were employed: (i) shape tests; (ii) Kolmogorov-Smirnov test (KS-Test) [9]; (iii) Welch's t-test (WT-Test) [21]; and (iv) regularity test (RT-Test) [5].

The shape tests employ first-order statistics such as mean, standard deviation and empirical cumulative distribution.

The KS-Test measures the distance between a legitimate sample and a test sample. A small test score implies that the test sample is close to the legitimate sample while a large test score indicates the possible presence of a covert timing channel.

The WT-Test determines if the means of two samples with different sizes and variances are different. The test statistic for the WT-Test is computed as:

$$t = \frac{\overline{X_1} - \overline{X_2}}{\sqrt{\frac{{s_1}^2}{N_1} + \frac{{s_2}^2}{N_2}}}$$

where $\overline{X_i}$, ${s_i}^2$ and N_i are the i^{th} sample mean, variance and sample size, respectively.

The RT-Test determines if the variance of inter-packet delays is relatively constant. In our evaluation, the sequence of outbound IPDs is divided into k segments, following which the standard deviation of segment i (σ_i) is computed. The regularity score is computed as:

$$regularity = STDEV(\frac{|\sigma_i - \sigma_j|}{\sigma_i}\ i < j, \forall i, j).$$

Jin and Wang [10] proposed a wavelet-based metric (MLAN) that quantitatively measures the practical effectiveness of anonymous networks in the presence of timing attacks. This metric can be used to determine the likelihood that a network flow has been distorted by a covert timing channel. MLAN is computed as:

$$e_j = \frac{\sum_{p=0}^{N_j-1}[CD(X,Y,j)(p)]^2}{N_j}$$

where $CD(X,Y,j)(p)$ is the p^{th} $(p = 0, \ldots, N_j - 1)$ wavelet detail coefficient at scale j for the j^{th} vector $D(X,Y,j)$ and $N_j = 2^{-j}n_j$ is the number of wavelet detail coefficients at scale j.

Gianvecchio and Wang [6] have shown that the corrected conditional entropy (CCE) is effective at detecting different types of covert timing channels. The CCE is computed as:

$$CCE(x_m|x_1, \ldots, x_{m-1}) = CE(x_m|x_1, \ldots, x_{m-1}) + perc(x_m)EN(x_1)$$

where $CE(x_m|x_1, \ldots, x_{m-1})$ is the estimated conditional entropy of the pattern $x_1, \ldots, x_m$; $perc(x_m)$ is the percentage of unique patterns of length m; and $EN(x_1)$ is the entropy with m fixed at one (i.e., first-order entropy only).

5.2 IPCTC Detection

The first set of experiments was designed to test the ability of our approach to detect the presence of IPCTCs [5] (see Table 1 for the mechanism). An IPCTC encodes a `1`-bit by transmitting a packet during a timing interval w, and encodes a `0`-bit by not transmitting a packet during w. In our experiments, the covert timing channel encoder read the `/etc/passwd` file and encoded its binary data into an IPCTC.

Four encoding schemes were used. Given the observation that the average IPD of traffic was 0.147s, we designed the first three encoding schemes by choosing timing intervals w of 0.1s (IPCTC1), 0.12s (IPCTC2) and 0.14s (IPCTC3). The fourth encoding scheme (IPCTC4) rotated between the three w timing intervals after every 100 packets to

Table 2. Test scores of legitimate and IPCTC IPDs.

	Legit.	**IPCTC1**	**IPCTC2**	**IPCTC3**	**IPCTC4**
Average #IPDs	0.1472	0.1560	0.1870	0.2180	0.1880
Mean	0.1472	0.1560	0.1870	0.2180	0.1880
Std Dev	0.041	0.089	0.106	0.124	0.111
Regularity	0.9723	0.9723	0.9723	0.9723	0.9723
WT-Test	0	1	1	1	1
KS-Test	0	1	1	1	1
KS-Test (p value)	0.9670	0	0	0	0
MLAN	0.0433	1.3616	1.5072	1.7034	1.353
CCE	1.1874	0.6055	0.6095	0.6099	0.7614
WBD	0.6295	15,982.68	37,092.88	76,188.46	50,241.94

avoid creating a regular pattern of inter-packet delays. This design ensured that both legitimate and covert timing channel flows have almost the same duration and send almost the same number of packets at run time.

The test was run 100 times for durations of 50 seconds. Around 400 packets were collected in each flow. For the regularity test, the sequence of inter-packet delays was divided into 20 segments. According to Gianvecchio and Wang [6], an IPCTC is the easiest covert timing channel to detect because its abnormality shows up in simple statistical tests.

Table 2 shows the detailed results for all the test flows. Although all the IPCTCs were detected in the tests, the WBD metric produced the best results. Note that the WBD of legitimate flows is very small (0.6295), which is in stark contrast to the WBDs of IPCTC flows, all of which are 15,982.68 or higher. Although MLAN and CCE are able to differentiate all four IPCTCs from legitimate flows, it is almost impossible to differentiate between the various IPCTCs. For example, the CCE values for all four IPCTCs range from 0.6055 to 0.7614, with three of them close to 0.6.

5.3 TRCTC Detection

The second set of experiments investigated the ability of our approach to detect TRCTCs [5]. Compared with an IPCTC, a TRCTC is a more advanced covert timing channel that replays a set of legitimate inter-packet delays to mimic the behavior of legitimate traffic. A TRCTC transmits a `0`-bit by randomly replaying an inter-packet delay from Bin_0 and transmits a `1`-bit by randomly replaying an inter-packet delay from Bin_1. Since Bin_i ($i = 0$ or 1) contains legitimate traffic, the distri-

Table 3. Test scores of legitimate and TRCTC IPDs.

	Legit.	**TRCTC1**	**TRCTC2**	**TRCTC3**	**TRCTC4**
Mean	0.1472	0.1546	0.1471	0.1466	0.1467
Std Dev	0.0406	0.0411	0.0409	0.0410	0.0411
Regularity (size = 20)	0.2453	0.2958	0.1991	0.3009	0.1955
WT-Test	0	0	0	0	0
KS-Test	0	0	0	0	0
KS-Test (p value)	0.9670	0.6451	0.9982	1	1
MLAN	0.0433	0.8700	1.0074	0.8995	0.8561
CCE	1.1833	1.1589	1.1749	1.1813	1.1829
WBD	0.4669	859.58	651.92	455.50	318.91

bution of TRCTC traffic is approximately equal to the distribution of legitimate traffic. The TRCTC was encoded and transmitted bit by bit to the receiver. Then, the message was rebuilt by the decoder bit by bit as in the IPCTC experiments.

Four versions of TRCTCi ($i = 1, \ldots, 4$) were created by injecting one additional bogus packet in the flow after every a packets ($a = 5$, 10, 15 and 20). Note that a larger value of a indicates a slower attack.

The TRCTC results shown in Table 3 are similar to those obtained in the IPCTC experiments. Since TRCTCs have the same distribution as legitimate traffic, the WT-Test and the KS-Test failed to detect the TRCTCs – the tests accept the null hypothesis that the IPDs of TRCTC traffic and legitimate traffic have the same distribution. Although MLAN and CCE tests can differentiate TRCTCs from legitimate traffic, they are incapable of identifying slow TRCTCs – the aggressive TRCTC (TRCTC1) and the stealthy TRCTC (TRCTC4) yield similar results for the MLAN and CCE tests. On the other hand, the WBD test detects all the TRCTCs, including the stealthy TRCTC that has the lowest score.

5.4 BTW Detection

BackTrack watermark (BTW) is a passive covert timing channel that is embedded in normal network traffic to track communications from a bot back to its botmaster [16]. We extended the covert timing channel design to generate slow attacks. In particular, we used $2a$ ($a \geq 1$) packets to encode a bit sequence S, where the parameter a is a constant or variable generated by a pseudorandom number generator. P_{r_i} and P_{e_i} were chosen from $2a$ packets. The parameter a serves as an "amplifier," specifying the speed at which data is transmitted via the covert channel. The larger the value of a, the slower the attack. Our experiments used

Table 4. Test scores of legitimate and BTW IPDs.

	Legit.	**BTW1 (a=5)**	**BTW2 (a=10)**	**BTW3 (a=20)**	**BTW4 (a=30)**
Mean	0.1474	0.2207	0.1814	0.1627	0.1587
Std Dev	0.0399	0.1775	0.1304	0.0918	0.0828
Regularity (size = 20)	0.2453	1.3504	0.7595	0.9206	0.9723
WT-Test	0	1	1	1	0
KS-Test	0	1	0	0	0
KS-Test (p value)	0.9670	0.0004	0.2808	0.9862	0.9999
MLAN	0.0433	1.8296	1.2081	1.4790	1.2394
CCE	1.1848	1.0440	1.0842	1.1079	1.1312
WBD	2.6757	1,489.90	828.76	250.86	124.97

four versions of BTW, each with a different value of a. A total of 60 seconds of live traffic was generated by the virtual machines.

The experimental results in Table 4 show that legitimate and BTW IPDs have similar means and standard deviations, especially for higher values of a. The WT-Test and KS-Test scores can be used to detect aggressive BTWs (BTW1), but they fail to detect BTW4, the stealthiest covert timing channel in the set of experiments. Note also that the MLAN test produces a large percentage of false positive errors because it is not very effective at discriminating between legitimate and BTW flows. In general, the CCE and WBD tests yield good results – they are able to detect all the BTWs while yielding a 100% true positive rate and a zero false positive rate. However, the WBD metric yields better results than CCE because, unlike CCE, it is able to quantify the stealth of BTWs – the larger the WBD score, the more aggressive the covert timing channel.

5.5 JitterBug Detection

JitterBug is a passive covert timing channel [17] that manipulates network traffic. It operates by creating small delays in key-presses that affect the inter-packet delays of an application. It transmits a `1`-bit by increasing an IPD to a value modulo w milliseconds and transmits a `0`-bit by increasing an IPD to a value modulo $\lfloor \frac{w}{2} \rfloor$ milliseconds. For small values of w, the distribution of JitterBug traffic is very similar to that of the original legitimate traffic. However, because of the small value of w, it can cause the covert timing channel to be indistinguishable from legitimate traffic containing noise.

A value of $w = 100$ milliseconds was chosen in our experiments. Four versions of JitterBug were employed, each with a different amplifier

Table 5. Test scores of legitimate and JitterBug IPDs.

	Legit.	**JB1**	**JB2**	**JB3**	**JB4**
Mean	0.1472	0.1626	0.1549	0.1510	0.1497
Std Dev	0.0406	0.0502	0.0482	0.0432	0.0432
Regularity (size = 20)	0.2452	0.2125	0.1848	0.2453	0.2969
WT-Test	0	1	0	0	0
KS-Test	0	1	0	0	0
KS-Test (p value)	1-E7.5	1-E7.5	1-E7.5	1-E7.5	1-E7.5
MLAN	0.0433	0.6380	0.6957	0.8286	0.0710
CCE	1.1834	1.2012	1.2015	1.1933	1.1944
WBD	7.5E-05	0.0218	0.0341	0.0349	0.0645

value: JitterBug1 ($a = 5$), JitterBug2 ($a = 10$), JitterBug3 ($a = 20$) and JitterBug4 ($a = 30$). This design ensured that legitimate and covert timing channel flows have almost the same duration and send about the same number of packets at run time. A total of 30,000×10 packets were collected in real time to evaluate covert timing channel detection and the true/false positive rates.

Table 5 shows the resulting test scores. The mean and standard deviation of legitimate traffic (0.1472 ± 0.0406) and stealthy covert timing traffic are very similar, especially for JitterBug4 (0.1497 ± 0.0432). The WT-Test and the KS-Test failed to detect JitterBugs, except for the most aggressive one (JitterBug1). In the case of the regularity test, the smaller scores indicate the presence of covert timing channels for the aggressive JitterBug1 and JitterBug2, but the scores are unable to discern the stealthy JitterBug3 and JitterBug4 channels. The CCE test is unable to differentiate between the different versions of Jitterbugs because most of the scores are between 1.18 and 1.20. However, the WBD test is able to detect the stealthiest JitterBug channel (JitterBug4) because its score is 0.0645, which is much larger than the score for legitimate traffic.

6. Conclusions

The wavelet-based approach described in this paper is well suited to detecting covert timing channels in real time. A key advantage is that the detection approach does not require historical traffic data. Experimental results demonstrate that the approach exhibits good performance for a variety of covert timing channels in networked virtual environments, including slow and stealthy channels. Furthermore, the detection approach is robust in the presence of the inaccurate timekeeping mechanisms used by virtual machines.

References

[1] A. Akansu and P. Haddad, *Multiresolution Signal Decomposition: Transforms, Subbands and Wavelets*, Academic Press, San Diego, California, 2001.

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt and A. Warfield, Xen and the art of virtualization, *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, pp. 164–177, 2003.

[3] V. Berk, A. Giani and G. Cybenko, Covert channel detection using process query systems, *Proceedings of the Second Annual Workshop on Flow Analysis*, 2005.

[4] R. Blahut, Computation of channel capacity and rate-distortion functions, *IEEE Transactions on Information Theory*, vol. 18(4), pp. 460–473, 1972.

[5] S. Cabuk, Network Covert Channels: Design, Analysis, Detection and Elimination, Ph.D. Dissertation, Department of Computer Science, Purdue University, West Lafayette, Indiana, 2006.

[6] S. Gianvecchio and H. Wang, Detecting covert timing channels: An entropy-based approach, *Proceedings of the Fourteenth ACM Conference on Computer and Communications Security*, pp. 211–230, 2007.

[7] S. Gianvecchio, H. Wang, D. Wijesekera and S. Jajodia, Model-based covert timing channels: Automated modeling and evasion, *Proceedings of the Eleventh International Symposium on Recent Advances in Intrusion Detection*, pp. 211–230, 2008.

[8] J. Giffin, R. Greenstadt, P. Litwack and R. Tibbetts, Covert messaging through TCP timestamps, *Proceedings of the Second International Conference on Privacy Enhancing Technologies*, pp. 194–208, 2002.

[9] M. Hollander and D. Wolfe, *Nonparametric Statistical Methods*, John Wiley, New York, 1999.

[10] J. Jin and X. Wang, On the effectiveness of low-latency anonymous network in the presence of timing attack, *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 429–438, 2009.

[11] S. Kullback and R. Leibler, On information and sufficiency, *Annals of Mathematical Statistics*, vol. 22(1), pp. 79–86, 1951.

[12] J. Lin, E. Keogh, L. Wei and S. Lonardi, Experiencing SAX: A novel symbolic representation of time series, *Data Mining and Knowledge Discovery*, vol. 15(2), pp. 107–144, 2007.

[13] Y. Liu, D. Ghosal, F. Armknecht, A. Sadeghi, S. Schulz and S. Katzenbeisser, Hide and seek in time: Robust covert timing channels, *Proceedings of the Fourteenth European Conference on Research in Computer Security*, pp. 120–135, 2009.

[14] P. Peng, P. Ning and D. Reeves, On the secrecy of timing-based active watermarking trace-back techniques, *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 335–349, 2006.

[15] M. Pereyra and L. Ward, *Harmonic Analysis: From Fourier to Wavelets*, American Mathematical Society, Providence, Rhode Island, 2012.

[16] D. Ramsbrock, X. Wang and X. Jiang, A first step towards live botmaster traceback, *Proceedings of the Eleventh International Symposium on Recent Advances in Intrusion Detection*, pp. 59–77, 2008.

[17] G. Shah, A. Molina and M. Blaze, Keyboards and covert channels, *Proceedings of the Fifteenth USENIX Security Symposium*, 2006.

[18] United States Government Accountability Office, Cyberspace: United States Faces Challenges in Addressing Global Cybersecurity and Governance, Report to Congressional Requesters, GAO-10-606, Washington, DC, 2010.

[19] VMWare, Timekeeping in VMWare virtual machines, Palo Alto, California (`www.vmware.com/files/pdf/Timekeeping-In-VirtualMachines.pdf`), 2011.

[20] X. Wang, S. Chen and S. Jajodia, Tracking anonymous peer-to-peer VoIP calls on the Internet, *Proceedings of the Twelfth ACM Conference on Computer and Communications Security*, pp. 81–91, 2005.

[21] B. Welch, The generalization of student's problem when several different population variances are involved, *Biometrika*, vol. 34(1-2), pp. 28–35, 1947.

[22] D. Zhang, A. Askarov and A. Myers, Predictive mitigation of timing channels in interactive systems, *Proceedings of the Eighteenth ACM Conference on Computer and Communications Security*, pp. 563–574, 2011.