An Entropy-Based Approach to Detecting Covert Timing Channels

Steven Gianvecchio and Haining Wang

Abstract—The detection of covert timing channels is of increasing interest in light of recent exploits of covert timing channels over the Internet. However, due to the high variation in legitimate network traffic, detecting covert timing channels is a challenging task. Existing detection schemes are ineffective at detecting most of the covert timing channels known to the security community. In this paper, we introduce a new entropy-based approach to detecting various covert timing channels. Our new approach is based on the observation that the creation of a covert timing channel has certain effects on the entropy of the original process, and hence, a change in the entropy of a process provides a critical clue for covert timing channel detection. Exploiting this observation, we investigate the use of entropy and conditional entropy in detecting covert timing channels. Our experimental results show that our entropy-based approach is sensitive to the current covert timing channels and is capable of detecting them in an accurate manner.

Index Terms—Network security, covert timing channels, entropy-based detection.

1 Introduction

As an effective way to exfiltrate data from a well-protected network, a covert timing channel manipulates the timing or ordering of network events (e.g., packet arrivals) for secret information transfer over the Internet, even without compromising an end-host inside the network. On the one hand, such information leakage caused by a covert timing channel poses a serious threat to Internet users. Their secret credentials like passwords and keys could be stolen through a covert timing channel without much difficulty. On the other hand, detecting covert timing channels is a well-known challenging task in the security community.

In general, the detection of covert timing channels uses statistical tests to differentiate covert traffic from legitimate traffic. However, due to the high variation in legitimate network traffic, detection methods based on standard statistical tests are not accurate and robust in capturing a covert timing channel. Although there have been recent research efforts on detecting covert timing channels over the Internet [1], [2], [3], [4], [5], some detection methods are designed to target one specific covert timing channel and therefore fail to detect other types of covert timing channels; the other detection methods are broader in detection but are over-sensitive to the high variation of network traffic. In short, none of the previous detection methods are effective at detecting a variety of covert timing channels.

In this paper, we propose a new entropy-based approach to detecting covert timing channels. The entropy of a process is a measure of uncertainty or information content, a concept that is of great importance in information and communication theory [6]. While entropy has been used in covert timing channel capacity analysis, it has never been used to detect covert timing channels. We observe that a covert timing channel cannot be created without causing some effects on the entropy of the original process[1]. Therefore, a change in the entropy of a process provides a critical clue for covert timing channel detection.

More specifically, we investigate the use of entropy and conditional entropy in detecting covert timing channels. For finite samples, the exact entropy rate of a process cannot be measured and must be estimated. Thus, we estimate the entropy rate with the corrected conditional entropy, a measure used on biological processes [8]. The corrected conditional entropy is designed to be accurate with limited data, which makes it excellent for small samples of network data. To evaluate our new entropy-based approach, we conduct a series of experiments to validate whether our approach is capable of differentiating covert traffic from legitimate traffic. We perform the fine-binned estimation of entropy and the coarse-binned estimation of corrected conditional entropy for both covert and legitimate samples. We then determine false positive and true positive rates for both types of estimations. Our experimental results show that the combination of entropy and corrected conditional entropy is very effective in detecting covert timing channels.

The remainder of this paper is structured as follows. Section 2 covers background and related work in covert timing channels and their detection schemes. Section 3 describes entropy measures. Section 4 validates the

S. Gianvecchio (srgian@cs.wm.edu) and H. Wang (hnw@cs.wm.edu) are with the Department of Computer Science, College of William and Mary, Williamsburg, VA 23188, US.

1. This observation applies to complex processes, like network traffic, but not to simple independent and identically distributed processes [7].

effectiveness of our approach through experiments with different covert timing channels. Section 5 describes potential countermeasures against our entropy-based detection scheme. Finally, Section 6 concludes the paper and discusses directions for our future work.

2 Background And Related Work

To defend against covert timing channels, researchers have proposed different solutions to detect, disrupt, and eliminate covert traffic. The disruption of covert timing channels adds random delays to traffic, which reduces the capacity of covert timing channels but degrades system performance as well. The detection of covert timing channels is accomplished using statistical tests to differentiate covert traffic from legitimate traffic. While the focus of earlier work is on disrupting covert timing channels [9], [10], [11], [12], [13] or on eliminating them in the design of systems [14], [15], [16], more recent research has begun to investigate the design and detection of covert timing channels [1], [2], [3], [4], [17], [5], [18]. In the following subsections, we give an overview of recent research on covert timing channels and detection tests.

2.1 Covert Timing Channels

There are two types of covert timing channels: active and passive. In terms of covert timing channels, active refers to covert timing channels that generate additional traffic to transmit information, while passive refers to covert timing channels that manipulate the timing of existing traffic. In general, active covert timing channels are faster, but passive covert timing channels are more difficult to detect. On the other hand, active covert timing channels often require a compromised machine, whereas passive covert timing channels, if creatively positioned, do not. The majority of the covert timing channels discussed in this section are active covert timing channels, except where stated otherwise.

2.1.1 IP Covert Timing Channel

Cabuk et al. [2] developed the first IP covert timing channel, which we refer to as IPCTC, and investigated a number of design issues. A scenario where IPCTC can be used is illustrated in Figure 1. In this scenario, a machine is compromised, and the defensive perimeter, represented as a perimeter firewall or intrusion detection system, monitors communication with the outside. Therefore, a covert timing channel can be used to pass through the defensive perimeter undetected. IPCTC uses a simple interval-based encoding scheme to transmit information. IPCTC transmits a 1-bit by sending a packet during an interval and transmits a 0-bit by not sending a packet during an interval. A major advantage to this scheme is that when a packet is lost, a bit is flipped but synchronization is not affected. The timing-interval t and the number of 0-bits between two 1-bits determines the distribution of IPCTC inter-packet delays. It is interesting to note that if the pattern of bits is uniform, the distribution of inter-packet delays is close to a Geometric distribution. To avoid creating a pattern of inter-packet delays at multiples of a single t, the timing-interval t is rotated among different values.

Fig. 1. IPCTC/TRCTC/MBCTC scenario

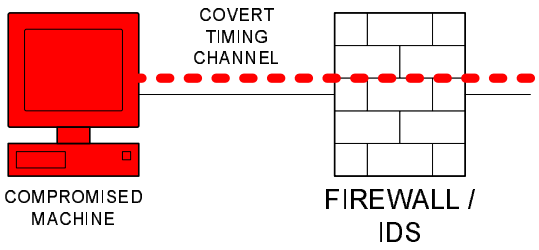
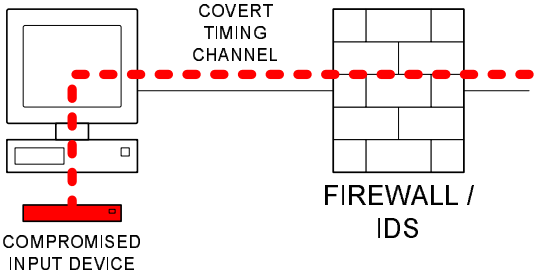


Fig. 2. JitterBug scenario



2.1.2 Time-Replay Covert Timing Channel

Cabuk [1] later designed a more advanced covert timing channel based on a replay attack, which we refer to as TRCTC. TRCTC uses a sample of legitimate traffic S_{in} as input and replays S_{in} to transmit information. S_{in} is partitioned into two equal bins S_0 and S_1 by a value t_{cutoff}. TRCTC transmits a 1-bit by randomly replaying an inter-packet delay from bin S_1 and transmits a 0-bit by randomly replaying an inter-packet delay from bin S_0. Thus, as S_{in} is made up of legitimate traffic, the distribution of TRCTC traffic is approximately equal to the distribution of legitimate traffic.

2.1.3 Model-Based Covert Timing Channel

Gianvecchio et al. [5] developed an automated framework for building model-based covert timing channels, which we refer to as MBCTC, to mimic legitimate traffic. MBCTC fits a sample of legitimate traffic to several models, such as Exponential or Weibull, and selects the model with the best fit. MBCTC then uses the inverse distribution function and cumulative distribution function for the selected model as encoding and decoding functions. MBCTC transmits by generating pseudo-random inter-packet delays using the inverse transform method of variate generation [19] to transmit hidden messages, i.e., messages are encoded using the inverse cumulative distribution function and decoded using the cumulative distribution function. Thus, as the distribution of the

pseudo-random inter-packet delays is determined by a model that approximates legitimate traffic, the distribution of MBCTC is close to that of legitimate traffic. To better model changes in the traffic, MBCTC refits the model in sets of 100 packets.

2.1.4 JitterBug

Shah et al. [3] developed a keyboard device called JitterBug that slowly leaks typed information over the network. JitterBug is a passive covert timing channel, so new traffic is not created to transmit information. JitterBug demonstrates how a passive covert timing channel can be positioned so that the target machine does not need to be compromised. A scenario where JitterBug can be used is illustrated in Figure 2. In this scenario, an input device is compromised, and the attacker is able to leak typed information over the network. JitterBug operates by creating small delays in keypresses to affect the inter-packet delays of a networked application. JitterBug transmits a 1-bit by increasing an inter-packet delay to a value modulo w milliseconds and transmits a 0-bit by increasing an inter-packet delay to a value modulo $\lceil \frac{w}{2} \rceil$ milliseconds. The timing-window w determines the maximum delay that JitterBug adds to an inter-packet delay. For small values of w, the distribution of JitterBug traffic is very similar to that of the original legitimate traffic. To avoid creating a pattern of inter-packet delays at multiples of w and $\lceil \frac{w}{2} \rceil$, a random sequence s_i is subtracted from the original inter-packet delay before the modulo operation.

2.1.5 Other Covert Timing Channels

Berk et al. [20] implemented a simple binary covert timing channel based on the Arimoto-Blahut algorithm, which computes the input distribution that maximizes the channel capacity [21], [22]. Luo et al. [4] designed a combinatorics-based scheme, called Cloak, to transmit information in the ordering of packets within different flows. Cloak can be considered as a storage and timing channel, as the encoding methods require packets and/or flows to be distinguishable by their contents. The same authors also proposed a covert timing channel based on the timing of TCP bursts [17]. Similar to Cloak, El-Atawy et al. [23] built a covert timing channel based on packet ordering and showed how code selection can make this technique effective at evading packet order metrics. Sellke et. al [18] showed that with independent and identically distributed (i.i.d.) traffic as cover, it is theoretically possible to create "provably secure" covert timing channels, i.e., covert timing channels that are computationally non-detectable. The same basic proof as [18] can be used to show that TRCTC is computationally non-detectable for i.i.d. cover traffic when its input messages are XOR'd with cryptographically-secure random numbers. Although not a covert timing channel, Giffin et al. [24] showed that low-order bits of the TCP timestamp can be exploited to create a covert storage channel, which is related to timing channels due to the shared statistical properties of timestamps and packet timing.

2.1.6 Timing-Based Watermarks

A number of efforts have investigated timing-based watermarking systems [25], [26], [27], [28], [29], [30], which are related to covert timing channels. A timing-based watermarking system is basically a side-channel that is augmented by a low-capacity covert timing channel. Wang et al. [26] proposed a method for watermarking inter-packet delays to track anonymous peer-to-peer voice-over-IP (VoIP) calls. More recently, Houmansadr et. al [30] proposed a subtle watermark called RAINBOW that is non-blind, i.e., it records both incoming and outgoing flows, allowing it to differentiate flows by adding only small delays. By doing so, RAINBOW is able to evade several detection tests, including entropy-based methods. However, the assumptions of timing-based watermarking systems, like RAINBOW, are quite different than those of covert timing channels. The entropy, if any, that is added by a watermarking system can be very small. For example, if a set of flows are naturally differentiable, a watermarking system need not add any delays to differentiate them. Generally, timing-based watermarking systems are passive timing channels in that new traffic is not created. Such systems again demonstrate how a passive timing channel can be positioned so that the target, i.e., the anonymizing network, does not need to be compromised.

2.2 Detection Tests

There are two broad classes of detection tests: shape tests and regularity tests. The shape of traffic is described by first-order statistics, e.g., mean, variance, and distribution. The regularity of traffic is described by second or higher-order statistics, e.g., correlations in the data. Note that in previous research the term regularity is sometimes used to refer to frequency-domain regularity [2], [3], whereas here we use this term exclusively to refer to time-domain regularity, i.e., the regularity of a process over time.

2.2.1 Kolmogorov-Smirnov Test

Peng et al. [27] showed that the Kolmogorov-Smirnov test is effective to detect watermarked inter-packet delays, a form of timing channel [25]. The watermarked inter-packet delays are shown to have a distribution that is the sum of a normal and a uniform distribution. Thus, the Kolmogorov-Smirnov test can be used to determine if a sample comes from the appropriate distribution. The Kolmogorov-Smirnov test determines whether or not two samples (or a sample and a distribution) differ. The use of the Kolmogorov-Smirnov test to detect covert timing channels is described in more detail in Section 4.1.2. The Kolmogorov-Smirnov test is distribution free, i.e., the test is not dependent on a specific distribution. Thus, the Kolmogorov-Smirnov test is applicable

to different types of traffic with different distributions. The Kolmogorov-Smirnov test statistic measures the maximum distance between two empirical distribution functions:

$$KSTEST = \max \mid S_1(x) - S_2(x) \mid, \quad (1)$$

where S_1 and S_2 are the empirical distribution functions of the two samples.

2.2.2 Regularity Test

Cabuk et al. [2] investigated a method of detecting covert timing channels based on regularity. This detection method, referred to as the regularity test, determines whether or not the variance of the inter-packet delays is relatively constant. This detection test is based on the fact that for most network traffic, the variance of the inter-packet delays changes over time, whereas with covert timing channels, if the encoding scheme does not change over time, then the variance of the inter-packet delays remains relatively constant. The use of the regularity test to detect covert timing channels is discussed in more detail in Section 4.1.2. For the regularity test, a sample is separated into sets of w inter-packet delays. Then, for each set, the standard deviation of the set σ_i is computed. The regularity is the standard deviation of the pairwise differences between each σ_i and σ_j for all sets $i < j$.

$$regularity = STDEV\left(\frac{\mid \sigma_i - \sigma_j \mid}{\sigma_i}, i < j, \forall i, j\right) \quad (2)$$

2.2.3 Other Detection Tests

Cabuk et al. [2] investigated a second method of detecting covert timing channels, referred to as ϵ-similarity, based on measuring the proportion of similar inter-packet delays. The ϵ-similarity test is based on the fact that IPCTC creates clusters of similar inter-packet delays at multiples of the timing-interval. Luo et al. [4] developed a detection method that targets the Cloak channel by measuring the intervals between acknowledgment and data packets. While both detection methods are effective at detecting the specific covert timing channels for which they are designed, namely IPCTC and Cloak, their respective scopes of detection are very limited. In comparison with more generic detection methods, they are less effective at detecting other types of covert timing channels. Berk et al. [20] used a simple mean-max ratio to test for bimodal or multimodal distributions that could be induced by binary or multi-symbol covert timing channels. Giani [31], continuing this research, explored the tradeoff between detection and covert timing channel capacity. Although entropy estimation has not been used previously to detect covert timing channels, it has been employed in related research on timing-based traffic analysis [32], [33] and network anomaly detection [34].

3 Entropy Measures

In this section, we first describe entropy, conditional entropy, and corrected conditional entropy, and then explain how these measures relate to first-order statistics, second or higher-order statistics, and the regularity or complexity of a process. Finally, we present the design and implementation of the proposed scheme to detect covert timing channels, based on the concept of entropy.

3.1 Entropy and Conditional Entropy

The entropy rate, which is the average entropy per random variable, can be used as a measure of complexity or regularity [8], [35], [33]. The entropy rate is defined as the conditional entropy of a sequence of infinite length. The entropy rate is upper-bounded by the entropy of the first-order probability density function or first-order entropy. A simple i.i.d. process has an entropy rate equal to the first-order entropy. A highly complex process has a high entropy rate, but less than the first-order entropy. A highly regular process has a low entropy rate, zero for a rigid periodic process, i.e., a repeated pattern.

A random process $X = \{X_i\}$ is defined as an indexed sequence of random variables. To give the definition of the entropy rate of a random process, we first define the entropy of a sequence of random variables as:

$$H(X_1, ..., X_m) = \\ - \sum_{X_1,...,X_m} P(x_1, ..., x_m) \log P(x_1, ..., x_m), \quad (3)$$

where $P(x_1, ..., x_m)$ is the joint probability $P(X_1 = x_1, ..., X_m = x_m)$.

Then, from the entropy of a sequence of random variables, we define the conditional entropy of a random variable given a previous sequence of random variables as:

$$H(X_m \mid X_1, ..., X_{m-1}) = \\ H(X_1, ..., X_m) - H(X_1, ..., X_{m-1}). \quad (4)$$

Lastly, the entropy rate of a random process is defined as:

$$\overline{H}(X) = \lim_{m \to \infty} H(X_m \mid X_1, ..., X_{m-1}). \quad (5)$$

The entropy rate is the conditional entropy of a sequence of infinite length and, therefore, cannot be measured for finite samples. Thus, we estimate the entropy rate with the conditional entropy of finite samples. It is also important to note that the definition of entropy rate is for stationary stochastic processes [36] and the extent to which measured data is non-stationary could affect the accuracy of entropy rate estimates.

3.2 Corrected Conditional Entropy

The exact entropy rate cannot be measured for finite samples and must be estimated. In practice, we replace probability density functions with empirical probability

density functions based on the method of histograms. The data is binned in Q bins. The specific binning strategy being used is important to the overall effectiveness of the test and is discussed in Section 3.3. The empirical probability density functions are determined by the proportions of patterns in the data, i.e., the proportion of a pattern is the probability of that pattern. Here a pattern is defined as a sequence of bin numbers. The estimates of the entropy or conditional entropy, based on the empirical probability density functions, are represented as: EN and CE, respectively.

There is a problem with the estimation of $CE(X_m \mid X_1, ..., X_{m-1})$ for some values of m. The conditional entropy tends to zero as m increases, due to limited data. If a specific pattern of length $m-1$ is found only once in the data, then the extension of this pattern to length m will also be found only once. Therefore, the length m pattern can be predicted by the length $m-1$ pattern, and the length m and $m-1$ patterns cancel out. If no pattern of length m is repeated in the data, then $CE(X_m \mid X_{m-1})$ is zero, even for i.i.d. processes.

To solve the problem of limited data, without fixing the length of m, we use the corrected conditional entropy (CCE) [8]. The corrected conditional entropy is defined as:

$$CCE(X_m \mid X_1, ..., X_{m-1}) = CE(X_m \mid X_1, ..., X_{m-1}) + perc(X_m) \cdot EN(X_1), \quad (6)$$

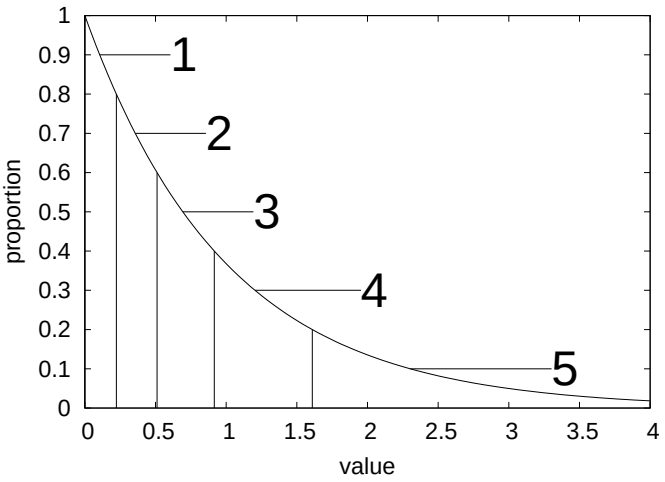
where $perc(X_m)$ is the percentage of unique patterns of length m and $EN(X_1)$ is the entropy with m fixed at one, i.e., only the first-order entropy.

The estimate of the entropy rate is the minimum of the corrected conditional entropy over different values of m. The minimum of the corrected conditional entropy is considered to be the best estimate of the entropy rate with the available data. The corrected conditional entropy has a minimum, because the conditional entropy decreases while the corrective term increases. The corrected conditional entropy has been mainly used on biological data, such as electrocardiogram [8] and electroencephalogram data [35]. Although not related to our work, it is interesting to see how such a measure can differentiate the states of complex biological processes. For example, with the electroencephalogram, an increase in the entropy rate indicates a decrease in the depth of anesthesia, i.e., the subject is becoming more conscious.

3.3 Binning Strategies

The strategy of binning the data is critical to the overall effectiveness of the test. The binning strategy mainly decides: (1) how the data is partitioned and (2) the bin granularity or the number of bins Q. In previous work, partitioning data into equiprobable bins seems to be most effective [8], [35]. The use of equiprobable bins is illustrated in Figure 3, showing the partitioning of Exponential data into bins of equal area. The bins, numbered 1 through 5, are small in width when the proportion of values is high and large in width when the proportion of values is low. Thus, while the bins have different widths, the total area of each bin is equal. The bin number for a value can then be determined based on the cumulative distribution function:

Fig. 3. The equiprobable binning of Exponential data in $Q = 5$ bins



$$bin = \lfloor F(x) * Q \rfloor, \quad (7)$$

where F is the cumulative distribution function and x is the value to be binned.

The bin numbers can also be determined based on ranges, e.g., $0.0 < bin_1 \leq 0.22$, $0.22 < bin_2 \leq 0.51$, $0.51 < bin_3 \leq 0.91$, and so on, which requires a search of the ranges to determine the correct bin number for a value. Meanwhile, the cumulative distribution function can determine the correct bin in constant time, which is important for performance when the number of bins is large.

The choice of the number of bins offers a tradeoff. While a larger number of bins retains more information about the distribution of the data, it increases the number of possible patterns Q^m and, thus, limits the ability of the test to recognize longer patterns due to the limited data. In contrast, a small number of bins captures less information about the distribution, but is better able to measure the regularity of the data. Therefore, as both strategies have advantages and disadvantages, we use both coarse-grain and fine-grain binning.

To determine the best choice of Q for coarse-grain binning, we run tests on correlated and uncorrelated samples for $Q = 2$ through 10. The correlated samples are 100 traces of 2,000 HTTP inter-packet delays. The uncorrelated samples are random permutations of the correlated samples. We then count the number of uncorrelated samples with scores that overlap with the scores of correlated samples. There is no overlap for the values of $Q = 5$ to 8. Therefore, to retain the ability of the test

to recognize longer patterns and measure regularity, we use $Q = 5$ for coarse-grain binning.

It is much simpler to determine the best choice of Q for fine-grain binning. With increasing values of Q, the number of possible patterns Q^m becomes much larger than the size of the sample being tested. At this point, the test scores are dominated by the estimate of the entropy for length one. Then, as we increase the value of Q, the bins continue to become more precise, leading to a better estimate of the entropy for length one than that for smaller values of Q. Therefore, as Q can be made arbitrarily precise, we use $Q = 2^{16} = 65{,}536$ for fine-grain binning.

3.4 Implementation Details

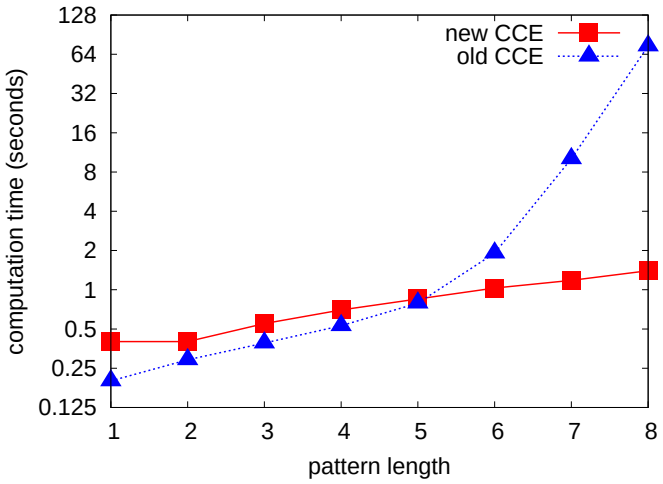
Our design goal is to be effective in detection and efficient in terms of run-time and storage. The efficiency of tests is particularly important if tests are conducted in real-time for online processing of data. Thus, we are careful to optimize our implementation for performance. We implement the corrected conditional entropy in the C programming language. The patterns are represented as nodes in a Q-ary tree of height m. The nodes of the tree include pattern counts and links to the nodes with longer patterns. The level of the tree corresponds to the length of patterns. The children of the root are the patterns of length 1. The leaf nodes are the patterns of length m.

To add a new pattern of length m to the tree, we move down the tree towards the leaves, updating the counts of the intermediate nodes and creating new nodes. Thus, when we reach the bottom of the tree, we have counted both the new pattern and all of its sub-patterns. After all patterns of length m are added, we perform a breadth-first traversal. The breadth-first traversal computes the corrected conditional entropy at each level and terminates when the minimum is obtained. If the breadth-first traversal reaches the bottom of the tree without having the minimum, then we must increase m and continue.

The time and space complexities are $O(n \cdot m)$, where n is the size of the sample, if we assume a priori knowledge of the distribution and use the cumulative distribution function to determine the correct bin for each value in constant time. Otherwise, the time complexity increases to $O(n \cdot m \cdot \log(Q))$. In practice, running our program on a sample of size 2,000 with $Q = 5$ and a pattern of length 10 on our test machine, an Intel Pentium D 3.4Ghz, takes 16 milliseconds. However, small changes in the implementation can have significant impact on performance.

To demonstrate this, we evaluate the computation overhead of our implementation and that of a previous implementation [35]. The computation time of both implementations with increasing pattern length is shown in Figure 4. For small values of m, our computation time is slightly longer, because of the overhead of creating our data structure. However, as m increases, the previous implementation increases quadratically, whereas

Fig. 4. CCE performance



our implementation increases linearly. The quadratic growth is caused by the separate processing of patterns of different lengths, i.e., the patterns of length 1, then the patterns of length 2, and so on, which introduces a quadratic term due to the summation of the pattern lengths: $\sum_{i=1}^{m} i = \frac{m^2+m}{2}$.

4 Experimental Evaluation

In this section, we validate the effectiveness of our proposed approach through a series of experiments. The focus of these experiments is to determine if our entropy-based methods (entropy and corrected conditional entropy) are able to detect covert timing channels. We test our entropy-based methods against four covert timing channels: IPCTC [2], TRCTC [1], MBCTC [5] and JitterBug [3]. Furthermore, we compare our entropy-based methods to two other detection tests: the Kolmogorov-Smirnov test and the regularity test [2].

The purpose of a detection test is to differentiate covert traffic from legitimate traffic. The performance of a detection test can be measured based on false positive and true positive rates, with low false positive rate and high true positive rate being desirable. In practice, because of the large variation in legitimate network traffic, it is important that tests work well for typical traffic and occasional outliers. If a detection test gives test scores with significant overlap between legitimate and covert samples, then it fails on detection. Therefore, the mean, variance, and distribution of test scores are critical metrics to the performance of a detection test.

4.1 Experimental Setup

The defensive perimeter of a network, made up of firewalls and intrusion detection systems, is designed to protect the network from malicious traffic. Typically, only a few specific application protocols, such as HTTP and SMTP, although heavily monitored, are allowed to

pass through the defensive perimeter. In addition, other protocols, such as SSH, might be permitted to cross the perimeter but only to specific trusted destinations.

We now consider the scenarios discussed in Section 2. In the first scenario, which relates to IPCTC, TRCTC and MBCTC, a compromised machine uses a covert timing channel to communicate with a machine outside the network. For IPCTC, TRCTC and MBCTC, we utilize outgoing HTTP inter-packet delays as the medium, due to the wide acceptance of HTTP for crossing the network perimeter and the high volume of HTTP traffic. In the second scenario, which relates to JitterBug, a compromised input device uses a covert timing channel to leak typed information over the traffic of a networked application. For JitterBug, we utilize outgoing SSH inter-packet delays as the medium, based on the original design [3] and the high volume of keystrokes in interactive network applications.

4.1.1 Dataset

The covert and legitimate samples that we use for our experiments are from two datasets: (1) HTTP traces we collected on a medium-size campus network and (2) a dataset obtained from the University of North Carolina at Chapel Hill (UNC). In total, we have 12GB of tcpdump packet header traces (HTTP protocol) that we collected and 79GB of tcpdump packet header traces (all protocols) from the UNC dataset [37]. In our experiments, we use several subsets of the two datasets, including:

- HTTP training set: 200,000 HTTP packets
- HTTP test set: 200,000 HTTP packets
- TRCTC test set: 200,000 HTTP packets
- MBCTC test set: 200,000 HTTP packets
- SSH training set: 200,000 SSH packets
- SSH test set: 200,000 HTTP packets
- JitterBug test set: 200,000 SSH packets

The packets in each dataset are grouped into flows. The flows represent outgoing traffic from a host to a specific port, e.g., port 80 for HTTP or port 22 for SSH. The flows are based on a 3-tuple of source host, destination port, and protocol, rather than a 5-tuple of source address, source port, destination address, destination port, and protocol. The subsets contain 100 samples and each sample has 2,000 packets from a flow.

In our experiments, we test a number of covert samples, which are generated from these subsets and from the encoding methods for IPCTC, TRCTC, MBCTC, and JitterBug. The covert timing channels are configured with the recommended settings from their original works, and we use the most advanced version if multiple versions of a covert timing channel are available. Specifically, IPCTC rotates the timing-interval t amongst 40ms, 60ms and 80ms; TRCTC is the BMC type; and JitterBug subtracts the random sequence s_i before the modulo operation. The input messages transmitted in our tests are random bits generated by a pseudo-random number generator, which avoids creating patterns in the output due to repeated bit sequences, such as 01100101 for the letter 'e' in ASCII-encoded text. For TRCTC, we generate the covert samples from a set of 200,000 legitimate HTTP inter-packet delays. For MBCTC, we generate the covert samples from a model that is selected by fitting multiple models to a set of 200,000 legitimate HTTP inter-packet delays. For JitterBug, we generate the covert samples from a set of 200,000 legitimate SSH inter-packet delays. A test machine replays the set of 200,000 SSH inter-packet delays and adds JitterBug delays. Note that our version of JitterBug is implemented in software. A monitoring machine on the campus backbone then collects a trace of the JitterBug traffic, which adds network delays after the addition of JitterBug delays. Since the monitoring machine is only four hops away from the test machine, with a RTT of 0.3ms, the added network delays are small. This JitterBug scenario is illustrated in Figure 2, where a defensive perimeter monitors outgoing traffic.

The training sets of legitimate traffic are useful for some of the detection tests. The Kolmogorov-Smirnov test uses the training sets to represent the behavior of legitimate traffic. The Kolmogorov-Smirnov test then measures the distance between the test sample and the training set. The entropy and corrected conditional entropy tests use the training sets to determine the range of each bin, based on equiprobable binning. These tests do not require a priori binning, but doing so improves performance, as the data does not need to be partitioned online.

4.1.2 Detection Methodology

In our experiments, we run detection tests on samples of covert and legitimate traffic. We use the resulting test scores to determine if a sample is covert or legitimate as follows. First, we set the targeted false positive rate at 0.01. To achieve this false positive rate, the cutoff scores—the scores that decide whether a sample is legitimate or covert—are set at the 99th or 1st percentile (high scores or low scores for different tests) of legitimate sample scores from the HTTP or SSH training set. Then, samples with scores worse than the cutoff are identified as covert, while samples with scores better than the cutoff are identified as legitimate. The false positive rate is the proportion of legitimate samples in the test set that are wrongly identified as covert, while the true positive rate is the proportion of covert samples in the test set that are correctly identified as covert.

Considering the properties of the detection tests, we can classify them as tests of shape or regularity. The shape of traffic is described by first-order statistics, and the regularity of traffic is described by second or higher-order statistics. The Kolmogorov-Smirnov test and entropy test are tests of shape, while the regularity test and corrected conditional entropy test are tests of regularity. The test scores are interpreted as follows.

In the Kolmogorov-Smirnov test, we measure the distance between the test sample and the training set that

represents legitimate behavior. Thus, if the test score is small, it implies that the sample is close to the normal behavior. However, if the sample does not fit the normal behavior well, the test score will be large, indicating the possible occurrence of a covert timing channel. By contrast, in the regularity test, we measure the standard deviation of the normalized standard deviations of sets of 100 packets. If the regularity score is low, then the sample is highly regular, indicating the possible existence of a covert timing channel.

The entropy test estimates the first-order entropy, whereas the corrected conditional entropy test estimates the higher-order entropy. The entropy test is based on the same algorithm as the corrected conditional entropy test, except that the corrective term is not added. The corrected conditional entropy test uses $Q = 5$, whereas the entropy test uses $Q = 65,536$ and m fixed at one. If the entropy test score is low, it suggests a possible covert timing channel, because the sample does not fit the appropriate distribution. If the conditional entropy test score is lower or higher than the cutoff scores, it suggests a possible covert timing channel. When the conditional entropy test score is low, the sample is highly regular. When the conditional entropy test score is high, near the first-order entropy, the sample shows a lack of correlations.

4.2 Experimental Results

In the following, we present our experimental results in detail. The four detection tests are: the Kolmogorov-Smirnov test, regularity test, entropy test, and corrected conditional entropy test. The four covert timing channels are: IPCTC, TRCTC, MBCTC, and JitterBug. The experiments are organized by covert timing channels, which are ordered in terms of increasing detection difficulty.

4.2.1 IPCTC

Our first set of experiments investigates how the detection tests perform against IPCTC [2]. IPCTC is the simplest among the three covert timing channels being tested and the easiest to detect, because it exhibits abnormality in both shape and regularity. The abnormal shape of IPCTC is caused by the encoding scheme. The encoding scheme encodes a 1-bit by transmitting a packet during an interval, and encodes a 0-bit with no packet transmission. Thus, the number of 0-bits between two 1-bits determines the inter-packet delays. If the bit sequence is random, then we can view the bit sequence as a series of Bernoulli trials and, thus, the inter-packet delays approximate a Geometric distribution. The timing-interval t is rotated among 40 milliseconds, 60 milliseconds, and 80 milliseconds after each 100 packets, as suggested by Cabuk et al. [2], to avoid creating a regular pattern of inter-packet delays at multiples of a single t. However, this instead creates a regular pattern of inter-packet delays at multiples of 20 milliseconds. The regularity of IPCTC is due to the lack of significant correlations between inter-packet delays. That is, the inter-packet delays are determined by the bit sequence being encoded, not by the previous inter-packet delays.

We run each detection test 100 times for 2,000 packet samples of both legitimate traffic and IPCTC traffic. The mean and standard deviation of the test scores are shown in Table 1. The detection tests all achieve lower average scores for IPCTC than those for legitimate traffic. The regularity test has a very high standard deviation for legitimate traffic, which suggests that this test is sensitive to variations in the behavior of legitimate traffic. The corrected conditional entropy test has a mean score for covert traffic that appears somewhat close to that of legitimate traffic, 1.96 for legitimate and 2.22 for covert. However, in relative terms, these scores are not that close, since the standard deviation of the corrected conditional entropy test is relatively low. The mean score for IPCTC is much closer to the maximum entropy than to the mean score of legitimate traffic. The maximum entropy is the most uniform possible distribution [36]. The maximum entropy for $Q = 5$ is:

$$H(X) = Q \cdot \frac{1}{Q} \log\left(\frac{1}{Q}\right) = 5 \cdot \frac{1}{5} \log\left(\frac{1}{5}\right) \approx 2.3219 \quad (8)$$

The corrected conditional entropy score is bounded from above by the first-order entropy. The first-order entropy is then bounded from above by the maximum entropy. Therefore, the corrected conditional entropy scores for IPCTC are close to the highest values possible.

As shown in Table 2, the detection rates for IPCTC (i.e. true positive rates for detecting IPCTC) are 1.0 for all tests except the regularity test, whose detection rate is only 0.54. The regularity test measures sets of 100 packets and the timing-interval t is rotated after each set of 100 packets, so the regularity test observes three distinct variances and accurately measures the regularity of IPCTC. The problem though is not measuring IPCTC, but measuring legitimate traffic. The very high standard deviation of the regularity test against legitimate traffic makes it impossible to differentiate IPCTC from legitimate samples without a higher false positive rate. Moreover, if we increase the timing-interval t to greater than 100 packets, the regularity test observes a different number of packets for each t value within each window, as the sets of t packets overlap with the window at different points, making the test less reliable. However, if we decrease the timing-interval t to much less than 100 packets, the regularity test observes a similar number of packets for each t value within each window and the variance for each window is similar, which makes the test more reliable.

Still, the main problem with the regularity test is its high standard deviation for legitimate traffic. The regularity test is very sensitive to outliers in legitimate traffic. For example, if σ_i is very small, due to a sequence of similar inter-packet delays, and σ_j is average or larger, then $\frac{|\sigma_i - \sigma_j|}{\sigma_i}$ is very large, especially for the values of σ_i close to zero, which are not uncommon. In fact, one

TABLE 1
IPCTC test scores

	HTTP-TEST		IPCTC	
test	mean	stdev	mean	stdev
$KSTEST$	0.180	0.077	0.708	0.000
$regularity$	35.726	36.635	0.330	0.056
EN	10.454	0.152	6.250	0.028
CCE	1.964	0.149	2.216	0.013

TABLE 2
IPCTC detection rates

	HTTP-TEST	IPCTC
test	false positive	true positive
$KSTEST \geq 0.36$	.00	1.00
$regularity \leq 0.41$	.01	.54
$EN \leq 8.56$	.01	1.00
$CCE \geq 2.16$	.01	1.00

TABLE 3
TRCTC test scores

	HTTP-TEST		TRCTC	
test	mean	stdev	mean	stdev
$KSTEST$	0.180	0.077	0.180	0.077
$regularity$	35.726	36.635	7.845	9.324
EN	10.454	0.152	10.454	0.152
CCE	1.964	0.149	2.217	0.012

TABLE 4
TRCTC detection rates

	HTTP-TEST	TRCTC
test	false positive	true positive
$KSTEST \geq 0.36$	.00	.01
$regularity \leq 0.41$	.01	.04
$EN \leq 8.56$	.01	.02
$CCE \geq 2.16$	.01	1.00

such outlier in a sample is more than sufficient to make a covert sample appear to be a legitimate sample. The high variance of the regularity test demonstrates that it is important to examine more than the average test score, since the variance and distribution of test scores are critical to the successful detection of covert timing channels.

4.2.2 TRCTC

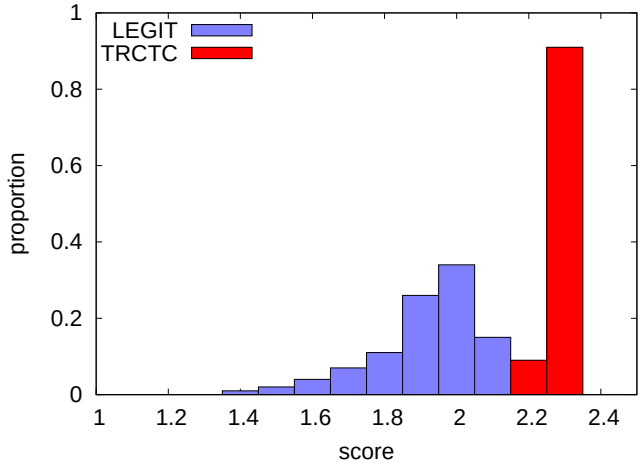
Our second set of experiments investigates how our detection tests perform against TRCTC [1]. TRCTC is a more advanced covert timing channel that makes use of a replay attack. TRCTC replays a set of legitimate inter-packet delays to approximate the behavior of legitimate traffic. Thus, TRCTC has approximately the same shape as legitimate traffic, but exhibits abnormal regularity, like IPCTC. The regularity of TRCTC, like IPCTC, is due to the lack of significant correlations between inter-packet delays. Although TRCTC replays inter-packet delays, the replay is in random order, as determined by the bit sequence that is being encoded, thus breaking the correlations in the original inter-packet delays.

We run each detection test 100 times for 2,000 packet samples of both legitimate traffic and TRCTC traffic. The mean and standard deviation of the test scores are shown in Table 3. The test scores for TRCTC and legitimate traffic are approximately equal for the Kolmogorov-Smirnov and entropy tests. These tests strictly measure first-order statistics, and, as such, are not able to detect TRCTC. The regularity test achieves a much lower average score for TRCTC than that for legitimate traffic, which is due to the similar variance between groups of packets in TRCTC. However, the standard deviation of the regularity test is again very high for legitimate traffic and, this time, is high for covert traffic as well. At the same time, the corrected conditional entropy test gives similar results to those for IPCTC. The corrected conditional entropy test has a mean score for TRCTC that appears somewhat close to that of legitimate, 1.96 for legitimate and 2.22 for covert. However, if we examine the distribution of test scores for TRCTC and legitimate traffic, as illustrated in Figure 5, then we can see that, although some scores are in adjacent bins, there is no overlap between the distributions. Furthermore, the distribution of legitimate test scores is strongly skewed to the left, away from the distribution of TRCTC test scores. The detection rates for TRCTC, as shown in Table 4, are very low (0.04 or less) for all the detection tests except the corrected conditional entropy test, which has a detection rate of 1.0. The corrected conditional entropy test scores of TRCTC are again close to the maximum entropy, therefore the corrected conditional entropy test is successful in detecting TRCTC.

4.2.3 MBCTC

Our third set of experiments investigates how our detection tests perform against MBCTC [5]. MBCTC is a

Fig. 5. CCE test scores for TRCTC



more advanced covert timing channel that exploits traffic modeling to mimic legitimate traffic. The traffic model is determined by using maximum likelihood estimation (MLE) to determine model parameters and then selecting the model with the lowest root mean squared error (RMSE) from several models. The model selected for legitimate HTTP traffic is Weibull with a mean scale parameter λ of 0.125 and a mean shape parameter k of 0.426. With these parameters, the mean inter-packet delay is 0.3524, approximately 3 packets per second. The model is then refitted in sets of 100 packets to better model changes in the traffic over time. Thus, MBCTC has a similar shape to legitimate traffic, due to modeling the distribution, and a similar regularity for sets of 100 packets or more, due to the refitting process.

We run each detection test 100 times for 2,000 packet samples of both legitimate traffic and MBCTC traffic. The mean and standard deviation of the test scores are shown in Table 5. The test scores of MBCTC are higher than those of legitimate traffic for the Kolmogorov-Smirnov test, though less than the standard deviation, due to the model being very close but not a perfect fit. The regularity test achieves a lower average score for MBCTC than that of legitimate traffic, though the standard deviation is again very high for legitimate traffic and covert traffic. The entropy test scores of MBCTC are higher on average than those of legitimate traffic, indicating that MBCTC traffic is consistently a somewhat close fit to the legitimate traffic distribution. The corrected conditional entropy test scores are significantly lower for MBCTC than for legitimate traffic. However, when we examine the distribution of test scores for MBCTC and legitimate traffic, as illustrated in Figure 6, we can see that there is a slight overlap between the distributions. This shows that the refitting process used by MBCTC, i.e., changing the model after each set of 100 packets, is relatively successful, but not sufficient to capture the true regularity of legitimate traffic. In particular, MBCTC traffic is more regular over time than legitimate traffic, i.e., the sequences of inter-packet delays are more predictable. For example, if a burst occurs, then the expected value of the model will be small and MBCTC will generate a larger portion of small inter-packet delays for the next 100 inter-packet delays. As a result, small inter-packet delays will be more likely to be followed by small inter-packet delays in MBCTC traffic than in legitimate traffic, which results in lower scores for the corrected conditional entropy test. The detection rates of MBCTC, as shown in Table 4, are very low (0.04 or less) for all the detection tests except the entropy test and the corrected conditional entropy test. The entropy test is able to sometimes detect MBCTC, with a detection rate of 0.55. The corrected conditional entropy test is very successful in detecting MBCTC, with a detection rate of 0.95.

TABLE 5
MBCTC test scores

	HTTP-TEST		MBCTC	
test	mean	stdev	mean	stdev
$KSTEST$	0.180	0.077	0.208	0.073
$regularity$	35.726	36.635	18.440	22.605
EN	10.454	0.152	10.739	0.078
CCE	1.964	0.149	1.156	0.223

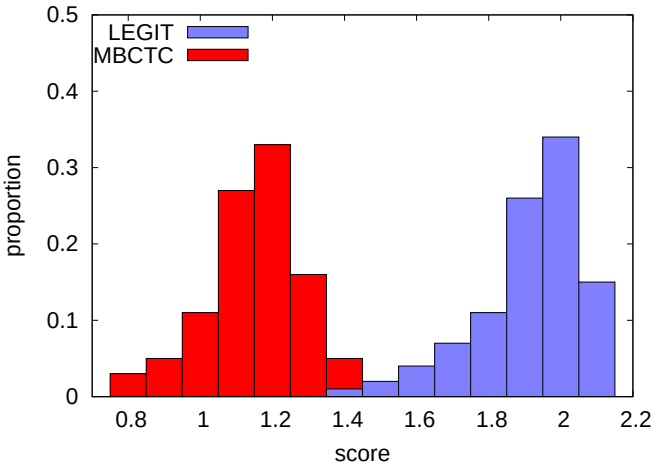
TABLE 6
MBCTC detection rates

	HTTP-TEST	MBCTC
test	false positive	true positive
$KSTEST \geq 0.36$	.00	.03
$regularity \leq 0.41$	.01	.02
$EN \geq 10.74$	.01	.55
$CCE \leq 1.50$	.00	.95

4.2.4 *JitterBug*

Our fourth set of experiments investigates how our detection tests perform against JitterBug [3]. JitterBug is a passive covert timing channel, so no additional traffic is generated to transmit information. Instead, JitterBug manipulates the inter-packet delays of existing legitimate traffic. The timing-window w, which determines the maximum delay that JitterBug adds, is set at 20 milliseconds, as suggested by Shah et al. [3]. The average inter-packet delay of the original SSH traffic is 1.264 seconds, whereas, with JitterBug, the average inter-packet delay is 1.274 seconds. In addition, while 10 milliseconds on average might be noticeable with other protocols, SSH traffic has a small proportion of short inter-packet delays, i.e., only about 20% of inter-packet delays are less than 30ms in the training set. Therefore, because of having legitimate traffic as a base and only slightly increasing the inter-packet delays, JitterBug is able to retain much

Fig. 6. CCE test scores for MBCTC



of the original correlation from the legitimate traffic. Moreover, by slightly increasing the inter-packet delays, JitterBug only slightly affects the original shape. Thus, JitterBug has similar shape and regularity to legitimate traffic.

Also JitterBug is very difficult to detect for several other reasons. From a practical perspective, the machine itself has not been compromised, so conventional host-based intrusion detection methods fail. Moreover, the traffic is encrypted, so the contents of the packets cannot be used to predict the appropriate behavior. Additionally, the position of JitterBug, between the machine and the human, further complicates detection because of the variation in human behavior, i.e., different typing characteristics. However, as JitterBug is a covert timing channel and transmits information, there is some effect on the entropy of the original process.

We run each detection test 100 times for 2,000 packet samples of both legitimate traffic and JitterBug traffic. The mean and standard deviation of the test scores are shown in Table 7. The test scores for JitterBug and legitimate traffic are close to each other for all the tests except the entropy test. If we examine the distribution of entropy test scores for JitterBug and legitimate traffics, as illustrated in Figure 5, we can see that the distributions of JitterBug and legitimate test scores are quite distinct. The detection rates for JitterBug shown in Table 8, are very low (0.04 or less) for all the detection tests except the entropy test, which has a detection rate of 1.0. Note that the other tests do detect some difference between JitterBug and legitimate traffic, but the differences are so small that it is impossible for these tests to differentiate JitterBug from legitimate traffic without a much higher false positive rate. Although the corrected conditional entropy test is successful at detecting all the other covert timing channels, it is unable to detect JitterBug. The corrected conditional entropy test bins the data into $Q = 5$ bins. For SSH traffic, the typical bin ranges (based on equiprobable binning) are $0.0 < bin_1 \leq 0.032$, $0.032 < bin_2 \leq 0.088$, $0.088 < bin_3 \leq 0.160$, $0.160 < bin_4 \leq 0.305$, and $0.305 < bin_5$. JitterBug adds a maximum of 20ms (10ms on average) to the inter-packet delays, so the bin numbers for inter-packet delays are rarely changed. Therefore, the corrected conditional entropy scores of JitterBug traffic are close to those of the original legitimate SSH traffic. In short, the corrected conditional entropy test is simply insensitive to small changes in the distribution.

In contrast, the entropy test is able to detect JitterBug. The entropy test uses a large number of bins, with bin widths determined by the distribution of legitimate traffic. The entropy test measures how uniformly the inter-packet delays are distributed into the bins, and how uniformly the inter-packet delays fit the legitimate traffic distribution. JitterBug creates small changes throughout the distribution. Since these changes fall within the variance that is typical of legitimate traffic, the tests that measure the maximum distance, like the Kolmogorov-Smirnov test, fail to detect the changes. However, the entropy test is sensitive to such changes throughout the distribution. JitterBug increases the inter-packet delays and, due to the rotating window, redistributes the inter-packet delays in an Equilikely distribution. However, the increases do not follow the legitimate distribution, leading to slight increases or decreases in the proportion of inter-packet delays for different bins. The entropy test measures how evenly the inter-packet delays are distributed into the bins, with the legitimate traffic distribution resulting in the most even or uniform distribution of bins and the most entropy, since the bins are sized to be equiprobable for the legitimate distribution. Therefore, the entropy test score for JitterBug is lower than that for legitimate traffic, which can be easily detected.

TABLE 7
JitterBug test scores

	SSH-TEST		JitterBug	
test	mean	stdev	mean	stdev
$KSTEST$	.270	.133	.273	.123
$regularity$	6.230	5.847	6.038	5.624
EN	10.663	0.374	8.199	0.720
CCE	1.779	0.261	1.837	0.220

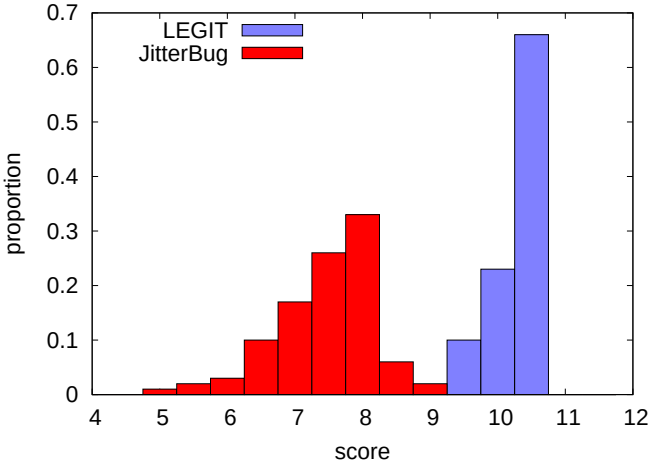
TABLE 8
JitterBug detection rates

	SSH-TEST	JitterBug
test	false positive	true positive
$KSTEST \geq 0.60$	.02	.03
$regularity \leq 0.15$	.03	.03
$EN \leq 8.84$	.01	1.00
$CCE \geq 2.16$	.01	.04

4.2.5 *All Channels - Variable Sample Size*

Our last set of experiments investigates how our detection tests perform with different sample sizes against all

Fig. 7. EN test scores for JitterBug



four covert timing channels, IPCTC, TRCTC, MBCTC, and JitterBug. We vary sample sizes from 500 to 2,200 inter-packet delays for the entropy test and the corrected conditional entropy test. The sample size is important because it determines (along with the packet rate) the amount of time it takes to detect a covert timing channel, and thus, the amount of information that a covert timing channel can transmit before it is detected. Of course, the faster a covert timing channel is detected, the less information it transmits prior to detection. However, there is a tradeoff between detection speed and detection accuracy. While a smaller sample size means faster detection, it tends to be less accurate compared to larger sample sizes.

The true positive rates for the entropy test against IPCTC, TRCTC, MBCTC, and JitterBug with 500 to 2,200 inter-packet delays are shown in Figure 8. The true positive rates degrade at different rates in different covert timing channels. On one extreme, for IPCTC there is no decrease in true positive rate and it is easily detected with as little as 500 inter-packet delays. The pattern of IPCTC is obvious, so there is no need for a large amount of data. On the other extreme, the true positive rates of JitterBug degrade more rapidly with the decrease of sample size, and it is difficult to detect JitterBug with less than 1,600 inter-packet delays. JitterBug is more subtle. It adds only small delays and has a relatively low capacity, so its inter-packet delays are indistinguishable from normal without more traffic. In the middle, the true positive rates of MBCTC degrade gradually with the decrease of sample size, starting at 0.59 and ending at 0.14, showing approximately a linear relationship between its true positive rate and its sample size. Lastly, TRCTC is not detected by the entropy test, so its true positive rates remain close to zero.

The true positive rates for the corrected conditional entropy test against IPCTC, TRCTC, MBCTC, and Jitterbug with 500 to 2,200 inter-packet delays are shown in Figure 9. IPCTC and TRCTC demonstrate a similar trend in their true positive rates. Both have true positive rates close to 1.0 with more than 700 inter-packet delays and then degrade quickly with the decrease of sample size. As neither covert timing channel attempts to capture inter-dependencies between inter-packet delays, this likely indicates that the minimum sample size required by the corrected conditional entropy test for accurate detection is around 700. The true positive rates of MBCTC again decline gradually with the decrease of sample size, starting at 1.00 and ending at 0.27, similar to the corresponding entropy test results. JitterBug is not detected by the corrected conditional entropy test, so its true positive rates are close to zero for all sample sizes.

Overall, combining the results of both tests, we can see that IPCTC and TRCTC are easier to be detected than MBCTC and Jitterbug when sample size is small. IPCTC and TRCTC can be accurately detected at the true positive rates of 1.0, with as little as 500 inter-packet delays and 1,000 inter-packet delays, respectively. MBCTC and JitterBug are much more difficult to detect, and they require close to 2,000 inter-packet delays or more for accurate detection. These results are attributed to the fact that MBCTC and JitterBug effectively capture both traffic shape and traffic regularity, while TRCTC only captures traffic shape and IPCTC captures neither of these two properties.

4.3 Discussion

The detection tests that we present are all able to detect some covert timing channels under certain conditions. However, the previous methods fail for detecting most of the tested covert timing channels. One major reason lies in the high variation of legitimate traffic. For example, the regularity test exhibits obvious weakness in this regard. Interestingly, the regularity test is the only test, other than the corrected conditional entropy test, that achieves lower average scores for all the covert timing channels. However, due to the high standard deviation of the regularity test in measuring legitimate traffic, the regularity test is not an effective detection method.

The other main reason lies in the properties of covert traffic. For example, while the Kolmogorov-Smirnov test is better able to deal with legitimate traffic variation, it has problems with covert timing channels whose distribution is very close to that of legitimate traffic. The Kolmogorov-Smirnov test measures the maximum distance between the two distributions, rather than measuring differences throughout the distribution. Thus, when the distribution of covert traffic is very close to that of legitimate traffic, the variance of the test scores is sufficiently large so that the test cannot differentiate covert traffic from legitimate traffic.

Our entropy-based approach proves more effective than previous schemes. Based on the advantages of different binning strategies, we make use of both entropy and corrected conditional entropy for detecting covert timing channels. The entropy test is sensitive to small changes throughout the distribution. However, for a covert timing channel whose distribution is nearly identical to that of legitimate traffic, the entropy test fails. By contrast, the corrected conditional entropy test measures the regularity or complexity of the traffic, rather than the distribution. Thus, it is effective to detect such a covert timing channel. However, if the original correlations of traffic are retained and the distribution is changed, then the conditional entropy test fails; but the entropy test works in this scenario by detecting slight changes in the distribution. Therefore, when both tests are combined, our entropy-based approach is effective in detecting all the tested covert timing channels.

5 Potential Countermeasures

In this section, we discuss possible countermeasures that could be used to harden covert timing channels against our entropy-based approach. Our discussion focuses on TRCTC, MBCTC and JitterBug. TRCTC and

Fig. 8. EN true positive rate vs. sample size

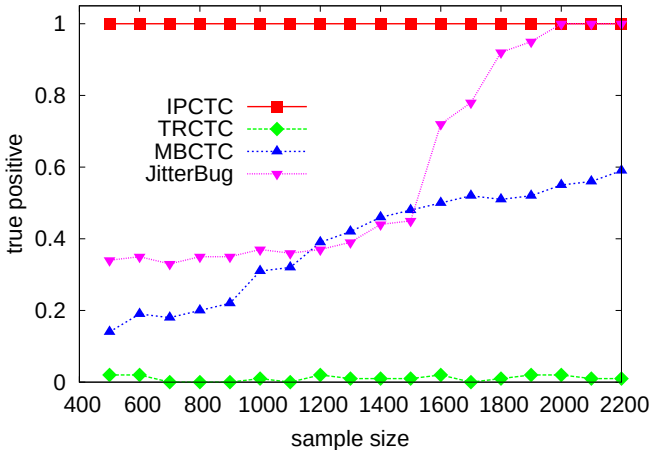
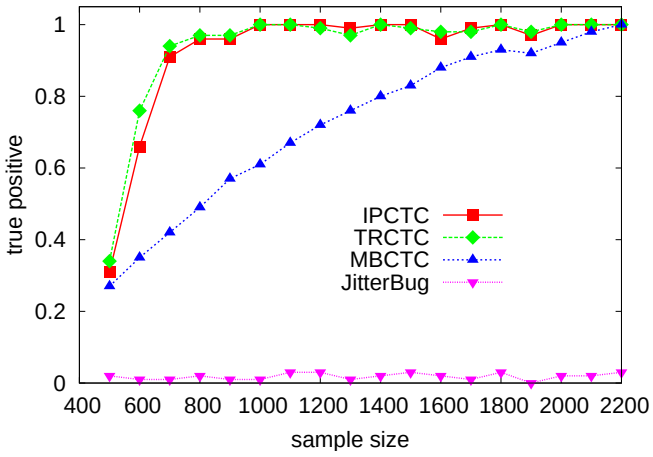


Fig. 9. CCE true positive rate vs. sample size



MBCTC are detected by the corrected conditional entropy test and JitterBug is detected by the entropy test.

In an attempt to evade the corrected conditional entropy test, TRCTC could be redesigned to replay longer correlated sequences of inter-packet delays. The corrected conditional entropy test could counter this technique for short sequences by increasing the minimum pattern length. Of course, with increasing sequence length, the corrected conditional entropy test would lose its capability to measure regularity, because of the issues discussed in Section 3, unless the sample size were increased. However, this is not a significant threat, because replaying long correlated sequences of inter-packet delays would greatly reduce the capacity of TRCTC. In an attempt to evade the corrected conditional entropy test, MBCTC could be changed to refit the model more frequently so as to better capture the regularity of traffic. Moreover, MBCTC could be redesigned to model conditional distributions to better capture inter-dependencies in traffic.

In an attempt to evade the entropy test, JitterBug could be reconfigured to use a smaller timing-window w. Eventually, as w becomes smaller, the entropy test would need a larger sample size to detect the JitterBug. However, using a smaller timing-window would, similar to our discussion of TRCTC, reduce the capacity of JitterBug. Additionally, JitterBug could be changed to transmit packets at more precise timing than milliseconds, as the millisecond-level precision could create a detectable pattern when the network delays are small. As another alternative, since a large number of inter-packet delays are required to detect JitterBug, JitterBug could attempt to transmit with fewer inter-packet delays than the minimum required for the entropy test. However, there is a problem with this approach. JitterBug uses forward error correction with repeated transmissions. This mechanism provides reliable communication even if packets are lost or some of the perturbed keystrokes go to a non-network application, neither of which can be detected by a JitterBug embedded in the keyboard. By reducing the number of repetitions, JitterBug could evade detection, but could also fail to deliver its message. It remains an open question whether these countermeasures would be practical.

6 Conclusion and Future Work

In this paper, we introduced an entropy-based technique to detect covert-timing channels by employing both entropy and corrected conditional entropy. We designed and implemented the proposed entropy-based detection tool. The development of this tool addresses a number of non-trivial design issues, including efficient use of data structures, data partitioning, bin granularity, and pattern length. We observed that as bin granularity increases, entropy estimates become more precise, whereas corrected conditional entropy estimates become less precise. Therefore, based on this observation, we utilized the fine-binned entropy estimation and the coarse-binned corrected conditional entropy estimation for covert timing channel detection.

We then applied our entropy-based techniques for detecting covert timing channels. The corrected conditional entropy test is able to detect the covert timing channels with abnormal regularity, while the entropy test is able to detect the covert timing channels with abnormal shape. Our experimental results show that the combination of entropy and corrected conditional entropy is capable of detecting a variety of covert timing channels. In contrast, for a covert timing channel whose distribution is close to that of legitimate traffic, all the previous detection methods fail. In light of our results and the importance of entropy in covert timing channel capacity, we believe that entropy could be the key metric for covert timing channel detection.

There are a number of possible directions for our future work. We plan to further investigate the possible countermeasures that could be used by attackers to evade entropy-based detection. We also plan to explore the connection between our entropy-based detection

methods and entropy as it relates to covert timing channel capacity. With detection methods that limit entropy, it could be possible to reduce the capacity available to covert timing channels. We believe that this exploration could lead to better detection methods or lower overall bounds on the capacity of covert timing channels.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments. We are grateful to the DiRT Group at the University of North Carolina at Chapel Hill for providing packet header traces, and the Information Technology Department at the College of William and Mary for providing a testing environment. This work was partially supported by NSF grants 0901537 and 0916022.

REFERENCES

[1] S. Cabuk, "Network covert channels: Design, analysis, detection, and elimination," Ph.D. dissertation, Purdue University, West Lafayette, IN., USA, December 2006.

[2] S. Cabuk, C. Brodley, and C. Shields, "IP covert timing channels: Design and detection," in *Proceedings of the 2004 ACM Conference on Computer and Communications Security*, October 2004.

[3] G. Shah, A. Molina, and M. Blaze, "Keyboards and covert channels," in *Proceedings of the 2006 USENIX Security Symposium*, July–August 2006.

[4] X. Luo, E. W. W. Chan, and R. K. C. Chang, "Cloak: A ten-fold way for reliable covert communications," in *Proceedings of 2009 European Symposium on Research in Computer Security*, September 2007.

[5] S. Gianvecchio, H. Wang, D. Wikesekera, and S. Jajodia, "Model-based covert timing channels: Automated modeling and evasion," in *Proceedings of the 2008 Symposium on Recent Advances in Intrusion Detection*, September 2008.

[6] C. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, July and October 1948.

[7] C. Cachin, "An information-theoretic model for steganography," *Information and Computation*, vol. 192, no. 1, 2004.

[8] A. Porta, G. Baselli, D. Liberati, N. Montano, C. Cogliati, T. Gnecchi-Ruscone, A. Malliani, and S. Cerutti, "Measuring regularity by means of a corrected conditional entropy in sympathetic outflow," *Biological Cybernetics*, vol. 78, no. 1, January 1998.

[9] J. Giles and B. Hajek, "An information-theoretic and game-theoretic study of timing channels," *IEEE Transactions on Information Theory*, vol. 48, no. 9, September 2002.

[10] W.-M. Hu, "Reducing timing channels with fuzzy time," in *Proceedings of the 1991 IEEE Symposium on Security and Privacy*, May 1991.

[11] M. H. Kang and I. S. Moskowitz, "A pump for rapid, reliable, secure communication," in *Proceedings of the 1993 ACM Conference on Computer and Communications Security*, November 1993.

[12] M. H. Kang, I. S. Moskowitz, and D. C. Lee, "A network version of the pump," in *Proceedings of the 1995 IEEE Symposium on Security and Privacy*, May 1995.

[13] M. H. Kang, I. S. Moskowitz, and S. Chincheck, "The pump: A decade of covert fun," in *Proceedings of the 2005 Annual Computer Security Applications Conference*, December 2005.

[14] J. Agat, "Transforming out timing leaks," in *Proceedings of the 2000 SIGPLAN/SIGACT Symposium on Principles of Programming Languages*, January 2000.

[15] R. A. Kemmerer, "A practical approach to identifying storage and timing channels," in *Proceedings of the 1982 IEEE Symposium on Security and Privacy*, April 1982.

[16] ——, "A practical approach to identifying storage and timing channels: Twenty years later," in *Proceedings of the 2002 Annual Computer Security Applications Conference*, December 2002.

[17] X. Luo, E. W. W. Chan, and R. K. C. Chang, "TCP covert timing channels: Design and detection," in *Proceedings of 2008 Dependable Systems and Networks*, June 2008.

[18] S. H. Sellke, C.-C. Wang, and S. Bagchi, "TCP/IP timing channels: Theory to implementation," in *Proceedings of the 2009 IEEE Conference on Computer Communications*, Rio de Janeiro, Brazil, April 2009.

[19] L. Leemis and S. K. Park, *Discrete-Event Simulation: A First Course*. Upper Saddle River, New Jersey: Prentice-Hall, 2006.

[20] V. Berk, A. Giani, and G. Cybenko, "Detection of covert channel encoding in network packet delays," Dartmouth College, Computer Science, Hanover, NH., USA, Tech. Rep. TR2005-536, August 2005.

[21] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Transactions on Information Theory*, vol. 18, no. 1, January 1972.

[22] R. E. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Transactions on Information Theory*, vol. 18, no. 4, July 1972.

[23] A. El-Atawy and E. Al-Shaer, "Building covert channels over the packet reordering phenomenon," in *Proceedings of the 2009 IEEE Conference on Computer Communications*, Rio de Janeiro, Brazil, April 2009.

[24] J. Giffin, R. Greenstadt, P. Litwack, and R. Tibbetts, "Covert messaging through TCP timestamps," in *Proceedings of the 2002 International Workshop on Privacy Enhancing Technologies*, April 2002.

[25] X. Wang and D. S. Reeves, "Robust correlation of encrypted attack traffic through stepping stones by manipulation of interpacket delays," in *Proceedings of the 2003 ACM Conference on Computer and Communications Security*, October 2003.

[26] X. Wang, S. Chen, and S. Jajodia, "Tracking anonymous peer-to-peer VoIP calls on the internet," in *Proceedings of the 2005 ACM Conference on Computer and Communications Security*, November 2005.

[27] P. Peng, P. Ning, and D. Reeves, "On the secrecy of timing-based active watermarking trace-back techniques," in *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, May 2006.

[28] X. Wang, S. Chen, and S. Jajodia, "Network flow watermarking attack on low-latency anonymous communication systems," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, Washington, DC, USA, May 2007.

[29] W. Yu, X. Fu, S. Graham, D. Xuan, and W. Zhao, "Dsss-based flow marking technique for invisible traceback," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, Washington, DC., USA, May 2007.

[30] A. Houmansadr, N. Kiyavash, and N. Borisov, "RAINBOW: A robust and invisible non-blind watermark for network flows," in *Proceedings of the 2009 ISOC Network and Distributed System Security Symposium*, February 2009.

[31] A. Giani, "Detection of attacks on cognitive channels," Ph.D. dissertation, Dartmouth College, Hanover, NH., USA, November 2006.

[32] X. Fu, B. Graham, R. Bettati, and W. Zhao, "On effectiveness of link padding for statistical traffic analysis attacks," in *Proceedings of the 23rd International Conference on Distributed Computing Systems*, Providence, RI., USA, May 2003.

[33] R. Moddemeijer, "On estimation of entropy and mutual information of continuous distributions," *Signal Processing*, vol. 16, no. 3, 1989.

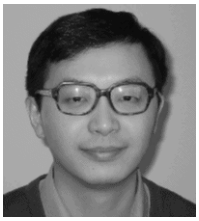
[34] Y. Gu, A. McCallum, and D. F. Towsley, "Detecting anomalies in network traffic using maximum entropy estimation," in *Proceedings of the 2005 Conference on Internet Measurment*, Berkeley, CA., USA, October 2005.

[35] R. Rosipal, "Kernel-based regression and objective nonlinear measures to assess brain functioning," Ph.D. dissertation, University of Paisley, Paisley, Scotland, UK, September 2001.

[36] T. M. Cover and J. A. Thomas, *Elements of information theory*. New York, NY., USA: Wiley-Interscience, 1991.

[37] "UNC CS network traces," http://www.cs.unc.edu/Research/dirt/ [Downloaded: Apr. 25, 2007].

Steven Gianvecchio is a Ph.D. candidate in Computer Science at the College of William and Mary. He received his M.S. in Computer Science from the College of William and Mary in 2006 and his B.S. in Computer Science from the State University of New York at Brockport in 2001. His research interests include networks, distributed systems, network monitoring, intrusion detection, traffic modeling, and covert channels.



Haining Wang received his Ph.D. in Computer Science and Engineering from the University of Michigan at Ann Arbor in 2003. He is an Associate Professor of Computer Science at the College of William and Mary, Williamsburg, VA. His research interests lie in the area of security, networking systems, and distributed computing. He is a senior member of IEEE.