Appeared in:Proceedings of the 23 IEEE International Conference on Distributed Computing Systems (ICDCS 2003), Providence, R.I., May 2003.

On Effectiveness of Link Padding for Statistical Traffic Analysis Attacks*

Xinwen Fu, Bryan Graham, Riccardo Bettati and Wei Zhao
Department of Computer Science
Texas A&M University
College Station, TX 77843 - 3112
E-mail:{xinwenfu, bwg7173, bettati, zhao}@cs.tamu.edu

Abstract

Traffic analysis attacks aim at deriving mission critical information from the analysis of the traffic transmitted over a network. Countermeasures for such attacks are usually realized by properly "padding" the payload traffic so that the statistics of the overall traffic become significantly different from that of the payload traffic. In this paper, we propose a analytical framework for traffic analysis attacks based on statistical pattern recognition techniques. We study the effectiveness of countermeasures for traffic analysis attacks within our proposed framework. Two basic countermeasure strategies are (a) to pad the traffic with constant interarrival times of packets (CIT) or (b) to pad the traffic with variable interarrival times (VIT). Our experiments show that CIT countermeasures fail when the adversary uses sample variance or sample entropy of packet interarrival times for statistical analysis. On the other hand, VIT countermeasures are effective regardless of which sample statistics are used by the adversary. These observations are validated by analysis of detection rates based on sample distributions of packet interarrival times.

Key Words packet interarrival times, traffic analysis attacks, statistical analysis, statistical pattern recognition

1 Introduction

In this paper, we investigate the effectiveness of link padding against statistical traffic analysis attacks. Computer networks are a critical infrastructure in supporting important services including telecommunication, banking, medicine, military, government, transportation, and electrical systems [13]. With the increasing usage of encryption to protect traffic content, *traffic analysis* is one of the common attacks that threaten privacy, anonymity, and confidentiality in such computer networks.

Traffic analysis attacks aim at deriving mission critical information by analyzing the traffic transmitted over a network. It is well known that even if the content of packets has been encrypted, characteristics of the traffic, such as traffic rate, pattern, or density, can reveal important mission critical information about the underlying applications [5, 22]. In this paper, we consider a class of traffic analysis attacks based on statistical pattern recognition techniques. We will assume that the information of interest to the adversary is the rate of payload traffic, but our analysis can be easily extended to other situations.

To facilitate this discussion, we propose a formal model for traffic analysis attacks. While traffic analysis attacks have been studied for decades, this is the first time a formal model is proposed. We assume that the adversary launching traffic analysis attack works in the following way: (1) The adversary first tries to collect information about the target system, including the system configuration and countermeasure algorithm used. In order to analyze the worst-case effectiveness of countermeasures, we assume that such information is available to the adversary. (2) Using the information collected, the adversary may simulate the target system. The challenge for the adversary is how to derive the payload information from the statistics of the simulating traffic. This is called *classification* in pattern recognition. For this purpose, the adversary is assumed to use pattern recognition techniques [12] to develop classification rules by comparing traffic statistics for various underlying traffic rates. Methods such as Bayesian decision rules can be used here [7, 8]. (3) At run time, the adversary then taps the network, collects samples of the traffic transmitted, derives the statistics of the sample, and uses this information to derive the traffic rate of the true payload traffic. The statistics used by the adversary include the mean, variance, and entropy of the interarrival times of the packets.

We believe that the above framework covers a broad

*This work was supported in part by the National Science Foundation under Contract EIA-0081761, by the Defensive Advanced Research Projects Agency under Contract F30602-99-1-0531, and by the Texas Higher Education Coordinating Board under its Advanced Technology Program.

range of traffic analysis attacks and provides a solid foundation to study the security of a communication system under the scrutiny of an adversary who uses traffic analysis.

Countermeasures for traffic analysis attacks are usually realized by properly "padding" the payload traffic so that the statistics of the overall traffic transmitted over the network become significantly different from that of the payload traffic. In this way, one hopes that analyzing statistics of the payload traffic will be difficult or even impossible. The padded traffic (also called covered traffic) can have either a constant interarrival time of packets (CIT) or a variable interarrival time (VIT).

The rationale for traffic padding is explained by Shannon's perfect secrecy theory: if we can map any payload traffic flow to a predefined pattern or a few predefined patterns with equal probability, then the adversary cannot obtain any information on the original payload traffic. While in theory this technique sounds extremely simple, in reality a perfect mapping cannot be achieved due to uncontrollable disturbances in the system. When this happens, the question is if the (small) disturbances help leak information and whether or not we can still establish a perfect secrecy system. If not, metrics must be defined to assess the effectiveness of the particular implementation.

Systems with this kind of countermeasure have been developed and utilized before. Prior to the work reported in this paper, there has not been a systematic study on the effectiveness of countermeasures versus statistical analysis attacks. We evaluate the performance of these countermeasures in terms of detection rate, that is, the probability that the adversary can correctly detect the traffic rate of payload traffic for a collected sample.

Our performance evaluation reveals a number of surprising results.

(a) Many traditional link padding systems have used CIT techniques. Our experimental data shows that this kind of system is only effective for attacks that use sample means for their statistical analysis.

(b) Our experimental data show that CIT-based countermeasures fail when the adversary uses sample variance and sample entropy for statistical analysis.

(c) On the other hand, VIT-based countermeasures seem to be effective regardless of which sample statistics is used by the adversary - sample mean, sample variance, or sample entropy.

By analyzing sampling distributions of packet interarrival times, design guidelines for VIT-based countermeasures can then be developed.

2 Related Work

Shannon in [18] describes his *perfect secrecy* theory that is the foundation for any ideal countermeasure system against traffic analysis attacks.

The study of traffic analysis and its countermeasures for computer networks is not new. Baran [1] proposes the use of heavy unclassified traffic to interfere with the adversary's tampering on the links of a security network system for classified information communication. He also suggests adding *dummy*, i.e. fraudulent, traffic between fictitious users of the system to conceal traffic loading.

To protect the anonymity of email transmission, Chaum [3] proposed the use of a *Mix* - a computer proxy. One technique used by a Mix is that it collects a predefined number K of fixed-size message packets from different users, shuffles the order of those packets, and then sends them out. The reality is that a mix cannot always get K packets efficiently from the users. So, it is suggested that users send dummy messages of random and non-meaningful content to maintain the Mix's security and efficiency. Most researchers have suggested CIT padding between the user and the proxy, e.g., [20]. CIT padding is also used here for preventing packet counting attacks [17].

A survey of the countermeasures for traffic analysis is given in [24]. To mask the frequency, length, and origin-destination patterns of end-to-end communication, dummy messages are suggested to pad the traffic to a predefined pattern.

The authors in [15, 16, 23] give a mathematical framework to optimize the bandwidth usage while preventing traffic analysis of the end-to-end traffic intensities. But, this optimization discloses the overall network link bandwidth usage, and the system cannot be said to be totally secure. Raymond in [17] gives an informal survey of many *ad hoc* traffic analysis attacks on systems providing anonymous service and possible solutions. One conclusion is that dummy messages must be carefully used to achieve high information assurance of the system. The authors of NetCamo [11] provide end-to-end prevention of traffic analysis while guaranteeing QoS (the worst case delay of message flows).

3 Models

3.1 Network Model

In this work, we assume that the network consists of *protected subnets*, which are interconnected by *unprotected links*. Traffic within protected subnets is assumed to be shielded from observers. Unprotected links use either public networks, or an easily accessible broadcast medium. These links are accessible to observation by third-parties, and are therefore open to traffic analysis. This model captures a variety of situations, ranging from battleship convoys (where the large-scale shipboard networks are protected and the inter-ship communication is wireless) to

communicating PDAs (where the protected networks consist of single nodes).

A common countermeasure for traffic analysis over unprotected links is *link-level padding*: to prevent the adversary from inferring, say, the *rate* of payload traffic over the unprotected link, additional "dummy" packets are properly inserted and transmitted over the link. In this way, the overall traffic appears to be at a constant rate, regardless of the true amount of payload traffic. Figure 1 illustrates the system configuration for link padding used in this paper.

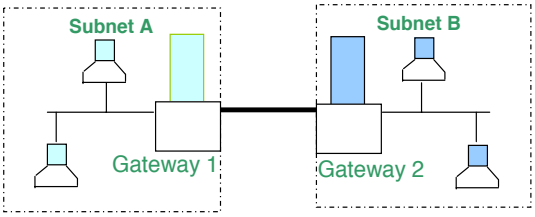

Figure 1. System Model

The hosts in the protected subnets (*subnet A* and *subnet B*) exchange traffic with each other through the unprotected link. Gateways are placed at the two boundaries of the unprotected link and provide the link-level padding necessary to prevent traffic analysis. We note that the gateways can be realized either as stand-alone boxes, or as modules to routers and switches, or software additions to network stacks or device drivers at the end hosts. For our experiments, they are realized as stand-alone boxes.

While the gateway at the sender's side generates and appropriately inserts padding traffic into the traffic flow to the receiver, the gateway at the receiver's site is responsible for detecting and discarding the padding traffic, i.e., converting the traffic on the link to the original payload traffic.

3.2 The Adversary

The goal of the adversary is to infer a set of characteristics of the traffic exchanged over the unprotected link. In this paper we will limit the interest of the adversary to the *payload traffic rate*, that is, the rate at which real traffic is exchanged between protected networks.

We assume that the payload traffic has n different states, $\omega_0, \ldots, \omega_{n-1}$, in terms of payload traffic rate. We call ω_0 the state where there is no payload traffic and $\omega_1, \ldots, \omega_{n-1}$ states with increasing payload traffic rates.

The adversary has full access to the traffic on the unprotected link and carries out its traffic analysis. In this paper, we make the following assumptions about the capabilities of the adversary.

(1) The adversary's access to the system is limited to the unprotected links. The protected subnets and the hosts within are not accessible. Neither is the link padding infrastructure. This means that, in Figure 1, the adversary can only tap the unsecured link between *Gateway 1* and *Gateway 2*.

(2) The contents (payload and headers) of packets transmitted between the gateways are perfectly encrypted. The adversary cannot obtain any useful information from the content of packets it observes. In particular, she cannot distinguish between payload packets and "dummy" packets used for padding based on packet content.

(3) Similarly, we assume that all packets have a constant size. Extensions to variable packet size will be discussed later.

(4) The adversary has complete knowledge about the gateway machines and the algorithms used in the countermeasures to prevent traffic analysis. The adversary may take advantage for such *a priori* knowledge of the system. In fact, we will assume in Section 4.1, that the adversary will indeed do so.

(5) The adversary limits itself to passive attacks, i.e., observations of the traffic. Based on the observed traffic, the adversary will try to infer the payload traffic rate on the unprotected link. In this paper we describe an effective method for how this can be done using statistical pattern recognition techniques.

Given that packets are non-distinguishable by packet size or packet content, the most valuable observation data collected by the adversary is limited to packet *interarrival times*. We will show in the following section that this limited observation space can yield interesting results in conjunction with appropriate statistical techniques.

4 Traffic Analysis Based on Statistical Pattern Recognition

4.1 Basic Framework

In this section, we describe a class of traffic analysis attacks that are based on statistical pattern recognition. We proceed by first giving a short overview on statistical pattern recognition and then describing how this general methodology can be applied to the traffic analysis problem described earlier.

Generally speaking, the goal of any statistical pattern recognition process is to try to *classify* an unknown pattern as belonging to one of several existing pattern *classes* with the help of a special feature (or attribute). In many cases, the classifier is trained from collected data ([6, 7, 12]). Following this common practice, a classifier that utilizes this strategy will consists of two subsystems: (1) on-line observation and classification and (2) off-line training.

Figure 2 (a) shows a flowchart for the on-line observation and classification subsystem, which consists of three

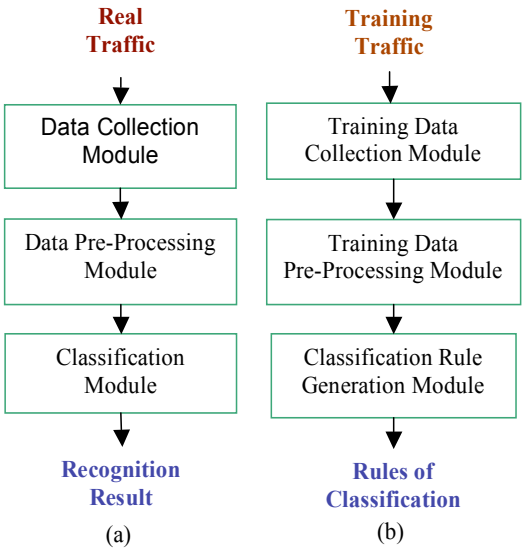

Figure 2. (a) on-line observation and classification (b) off-line training subsystem subsystem

modules:

Data Collection Module is responsible for sampling the traffic information from the real system. In our case, this consists of collecting interarrival times of packets for a prespecified number of packets, called *sample size*. The sample generated here is called the *real sample*.

Data Pre-Processing Module pre-processes the real sample collected by the Data Collection Module in order to remove noise.

Classification Module performs two functions: First, calculates the *sample statistical feature* of interest (such as, mean, variance, or entropy) for a given sample. Second, it classifies the traffic based on this sample's statistical feature. The classification is done by a set of rules that are derived by the off-line training subsystem. The result of this module is a classification of the measured traffic (real sample) into a set of states (classes) $\omega_0, \ldots, \omega_{n-1}$, in our case levels of payload traffic rate.

Figure 2 (b) shows a flowchart for the off-line training subsystem. The purpose of the this subsystem is to derive a set of classification rules to be used by the Classification Module during run time. Typically, one would expect that the adversary has access to the hardware and software of the traffic analysis countermeasure infrastructure (or a copy thereof) and so can generate training traffic used as the input for the off-line training subsystem.

This off-line training subsystem consists of the following modules.

Training Data Collection Module is responsible for generating *training samples* from the training traffic information. In contrast to real samples collected on-line, the classification of training samples is known *a priori*, i.e., supervised training.

Training Data Pre-Processing Module pre-processes the training samples generated by the Training Data Collection Module in order to remove noise.

Classification Rule Generation Module first calculates the *sample statistical feature* of interest (e.g., mean, variance, entropy) for all training samples. As the training samples are classified *a priori*, the distributions of the sample statistical feature of interest can be computed for each class separately. The rules for classification can then be derived from these distributions.

In order to make this general framework applicable to our problem, we need to address a number of issues: First, we need to collect the sample's statistical feature of interest. Once the values for the sample's statistical feature are available for the training set, their distribution must be computed. Finally, the classification rules must be derived based on the distributions of the sample's statistical features. We will elaborate on these issues in the remaining part of this section.

4.2 Sample Statistical Features

The selection of feature statistics is key to the success of the adversary. In this paper, our feature vector is one-dimensional, i.e., we use one feature of the data sample for the classification. The statistics on packet interarrival times (PIATs) are chosen as the candidate features since all the packets have been padded to the same size and the content is perfectly protected, making PIAT the most valuable information.

The three most interesting candidate features are *sample mean*, *sample variance*, and *sample entropy* of PIATs.

Sample Mean: For a sample of size n, $\{X_1, \cdots, X_n\}$, the mean m of PIATs is given as follows

$$m = \sum_{i=1}^{n} X_i/n \tag{1}$$

Sample Variance: The unbiased estimate of the variance s^2 is used. That is,

$$s^2 = \sum_{i=1}^{n} (X_i - m)^2/n \tag{2}$$

Sample Entropy: The entropy can be computed based in the method developed in [14]. First, we create a histogram of the interarrival times in the sample for a given bin size (say, Δx). According to [14], the differential entropy estimator of a random variable X's continuous distribution is

given by

$$\widetilde{H} = -\sum_i \frac{k_i}{n} \log \frac{k_i}{n} + \log \Delta x \tag{3}$$

where W is the number of interarrival times in the sample, k_i is the number of samples in the i^{th} bin, and Δx is the histogram's bin size. If a constant bin size is used throughout the experiment, the term $\log \Delta x$ in (3) is a constant and hence does not influence the recognition result. It can therefore be discarded, and the entropy estimation formula simplifies to

$$\widetilde{H} = -\sum_i \frac{k_i}{n} \log \frac{k_i}{n} \tag{4}$$

4.3 Distributions of Sample Statistical Features

The off-line training subsystem computes estimates of the distributions, in particular the density functions, of sample statistical features. Among the many ways that can be used to estimate density functions [12, 19] we briefly describe the *Parzen Window* method [7].

The Parzen Window density estimation uses the superposition of a normalized window function centered on a set of random samples, such that

$$p(x) \approx \widetilde{p}(x) = \frac{1}{N} \sum_{i=1}^{N} G_\psi(x - x_i) \tag{5}$$

where $p(x)$ is the density function of features, $\widetilde{p}(x)$ is the estimation of the density function, N the sample size, $\{x_i : 1 \leq i \leq N\}$ the random sample, $G_\psi(x)$ the window function, and ψ the window width. N must be big enough to capture the class (population) characteristics. In this paper, we assume G_ψ to be a normalized Gaussian (Normal) density distribution, that is

$$G_\psi(x - x_i) = \frac{1}{\sqrt{2\pi}\psi} e^{\frac{(x-x_i)^2}{\psi^2}} \tag{6}$$

Generally, when $N \to \infty$ and $\psi \to 0$, $\widetilde{p}(x) \to p(x)$. We need to carefully choose ψ and N to control the smoothness and bias of the estimated density function.

4.4 Classification Rules

Classification can be done using a classical minimum-error-rate decision rule i.e., Bayesian decision rule. This kind of rule can be described as follows: For each state $\omega_j, j = 0 \cdots n-1$, from Section 4.3, we know the univariate probability density (or distribution) function of a given sample feature y conditioned on ω_j, $p(y|\omega_j)$ and the probability of occurrences of ω_j. The optimal decision rule that minimizes the average number of wrong decisions is called the *Bayesian decision rule*. For an even penalty function $(0, 1)$, by which a point is lost on a wrong decision and no points are lost on a correct decision, this rule is as follows:

The sample represented by y belongs to State ω_i if

$$\forall j \in \{1, \cdots, m\}, \;\; P(\omega_i|y) \geq P(\omega_j|y). \tag{7}$$

This rule tells us that the sample represented by x should be classified as class ω_i with the biggest *post priori* probability $P(\omega_i|y)$.

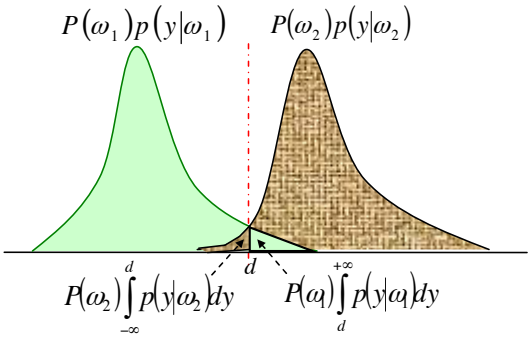

Figure 3. Bayesian Decision's Error Rate - The Case of Two Classes

Consider such a case in Figure 3 where we want to classify two states that have bell-shaped distributions. Note that, according to (7), the decision rule now becomes

If $y \leq d$, y is in state 1 else state 2. (8)

4.5 Detection Rate

The classification error is the measure of a classifier's performance. It is defined as the probability of error when the classifier is used to assign an unknown pattern to one of the pattern classes.

For a n-state decision problem, the error rate ϵ can be calculated as follows:

$$\epsilon = \int_{-\infty}^{\infty} (1 - \max_{i=1}^{n} P(\omega_i|y)) p(y) dy \tag{9}$$

For a two-state system shown in Figure 3, the above formula becomes as follows:

$$\epsilon = P(\omega_2) \int_{-\infty}^{d} p(y|\omega_2) dy + P(\omega_1) \int_{d}^{+\infty} p(y|\omega_1) dy \tag{10}$$

The *detection rate* is defined as the probability that a successful classification is made. Given an error rate ϵ, the detection rate, v, is given by

$$v = 1 - \epsilon = \int_{-\infty}^{\infty} \max_{i=1}^{n} P(\omega_i|y) p(y) dy \tag{11}$$

5 Countermeasures and Evaluation

In this section, we implement a set of countermeasures to traffic analysis attacks and evaluate them based on the type of statistical pattern recognition techniques described in Section 4. The goal of the adversary is to maximize its detection rate as defined in (11). Consequently, the purpose of our countermeasures is to minimize this detection rate.

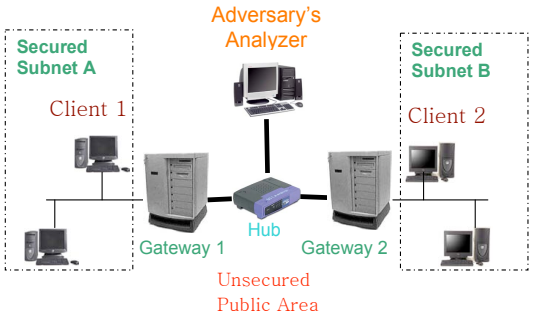

Figure 4. Experiment Setup

The experimental setup for the evaluation is illustrated in Figure 4, with two Linux PCs acting as countermeasure gateways. The PCs have two network interfaces each and isolate the protected subnet from unprotected link by appropriately inserting "dummy" packets into the unprotected link. The observations by the adversary are made using a network analyzer on the link between the two gateways. Our gateway machines use a real-time Linux operating system [21] produced by TimeSys Inc.

Data from other systems (e.g., standard Linux with kernel 2.4) we analyzed confirm the same conclusions made in this section. Thus, we are not going to show them in this paper due to space limitations.

5.1 Link padding with Constant Interarrival Times of Packets

Recall that the basic function of the gateway is to pad traffic by properly inserting certain dummy packets into the payload traffic flow so that the rate of real payload (user) traffic becomes unrecognizable. Let us call the output traffic from the gateway *covered traffic*. A key question in design of a countermeasure is how to define the interarrival times of packets in the covered traffic. Traditionally, constant interarrival times have been used for covered traffic giving rise to constant rate link padding. The idea behind this choice is that the covered traffic has constant interarrival time and thus leaks no information about any payload it covers and hence provides the best protection for the payload traffic.

We implemented such a gateway and carried out experiments. In our experiments, the payload packets arrive at the gateway as a Poisson process with different mean rates, i.e., different states that the adversary tries to differentiate. For the sake of simplicity in the following discussion, we will limit ourselves to data from a system where the payload traffic only has *two states*: State ω_0 in which there is a mean of 10 identical-sized packets per second transmitted from one subnet to the other, and State ω_1 in which payload traffic mean rate is 40 packets/second. We further assume that there is a 50% chance that the payload is in ω_0 or ω_1. It is clear from (10) that the maximum error rate in such a system is 50%[1] and the detection rate defined in (11) is lower-bounded by 50%. The covered traffic with the constant interarrival time departs from the gateway at the rate of 100 packets per second. The timing of covered traffic is controlled by a constant interval timer, i.e., a periodic timer with a period of 10 ms. That is, the timer generates an interrupt every 10 ms. The timer interrupt routine sends the payload packet if available and otherwise sends a dummy packet.

The adversary uses three sample statistical features, namely sample mean, sample variance, and sample entropy as defined in (1), (2), and (4) respectively. Figure 5 shows the results of these experiments. From this figure the following observations can be made:

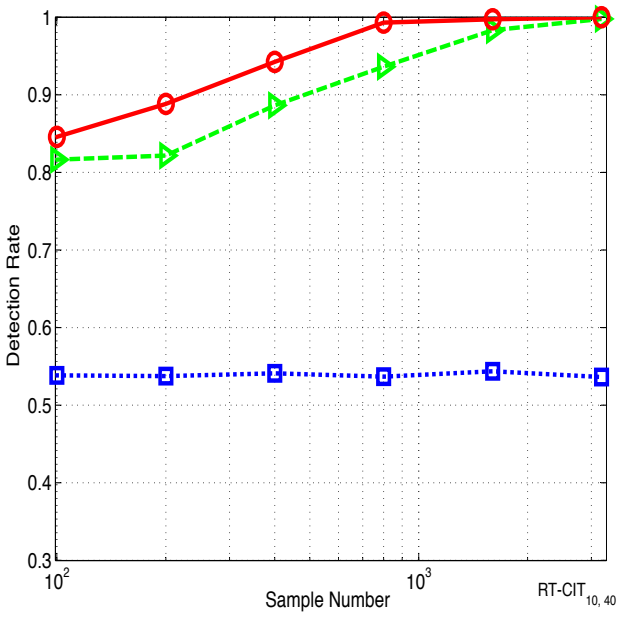

Figure 5. Detection Rate of CIT Link Padding

(a) Success rate is indeed lower-bounded by 50%. This at least partially validates the correctness of the statistical approaches we used.

[1] That is, the error rate for which an adversary that just randomly guesses would be 50%

(b) When the adversary uses sample mean as sample statistical feature in rate recognition, it can only achieve 50% detection rate. This clearly illustrates the effectiveness of link padding when used against this kind of adversary strategies.

(c) When the adversary uses sample variance or sample entropy as sample statistical feature in rate recognition, the figure becomes interesting: The detection rate increases as sample size increases and eventually reaches 100%! In this case, the constant interarrival time link padding completely fails to cover the rate of real payload traffic.

(d) Moreover, sample entropy provides a better detection rate than sample variance. This is because sample entropy is not sensitive to noise since it's a probability weighted sum and noise (big outliers) have a very small probability to occur. As we know, sample variance is very sensitive to big outliers.

Why does CIT link padding fail under traffic analysis attacks using sample variance and sample entropy? One can explain this phenomenon by investigating the operation of the gateway. The network interface card of the gateway captures an incoming packet of the payload traffic. It then generates an interrupt request, which can preempt all the processes including the scheduled timer. For TimeSys Linux/RT, this request proceeds before the incoming packet reaches the IP layer (more precisely, halfway in the network device driver) [10]. From that instant on, the network subsystem in the kernel becomes preemptive. Other tasks (e.g., timer) can then proceed as scheduled. Thus, incoming packets may subtly delay the timer's interrupt routine because of hardware interrupt and process scheduling latencies. Payload traffic with a higher rate has more chance to delay the timer. Thus, the degree by which the timer is delayed has a correlation with the state (rate) of the real payload traffic.

To further validate our observations and develop better countermeasures, we carry out an analytical analysis that reveals how sample size and variance of covered traffic impact detection rates. The details of the analysis are reported in [9]. The particular result that are relevant to this paper is as follows. Let

$$r = \sigma^2_{\omega_1} / \sigma^2_{\omega_0} \tag{12}$$

where $\sigma^2_{\omega_0}$ and $\sigma^2_{\omega_1}$ are the variances of packet interarrival times of covered traffic for state ω_0 and ω_1, respectively.

Theorem 1. When sample variance or sample entropy is used as statistical feature, the detection rate approaches to 50% when r approaches to 1.

5.2 Link Padding with Variable Interarrival Times of Packets

Following the discussion made in Section 5.1, we now develop better countermeasures. Note that by its definition,

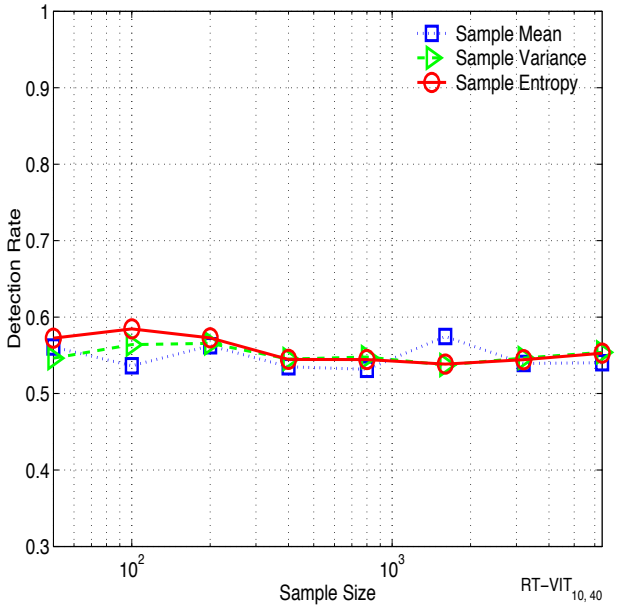

Figure 6. Detection Rate of VIT Link Padding

r can be approximately presented as follows:

$$r = (\sigma_t^2 + \sigma^2_{\omega_1}) / (\sigma_t^2 + \sigma^2_{\omega_0}) \tag{13}$$

where σ_t^2 is the variance of time interval between interrupts and $\sigma^2_{\omega_0}$ and $\sigma^2_{\omega_1}$ are the additional variance due to disturbances for states 0 and 1, respectively. We have no control on $\sigma^2_{\omega_0}$ and $\sigma^2_{\omega_1}$, but if we make σ_t^2 sufficiently large, r will be approximately 1. This implies that we should make the covered traffic with variable interarrival time. We carried out the experiments in this way. We set the interval of interrupt timer with a truncated normal distribution of $N(10ms, \sigma_t^2)$. The truncation results in time outs between 1 ms and 19 ms.

The results in Figure 6 clearly show the effectiveness of this new link padding scheme. The detection rate is very close to 50% for all the three sample statistical features where $\sigma_t = 2$ms. This confirms our prediction stated in Theorem 1.

6 Final Remarks

This paper introduced a formal framework for analyzing the security of a communication system that is subject to traffic analysis attacks. The framework is based on statistical pattern recognition techniques and covers a broad range of traffic analysis attacks.

Based on this framework, we have made interesting discoveries. We found that traditional CIT link padding technique may fail in preventing traffic analysis from determin-

ing the rate of real payload traffic. CIT link padding is *theoretically* sound. However, its implementation generally needs a timer as a mechanism to control the transmission of dummy and payload packets and for most of the modern operating systems, the payload traffic that goes into the padding machine (i.e., gateway) will interfere with the timer's behavior. Heavier load incurs more interference, so packet interarrival times of the covered traffic will have a correlation with the rate of the payload traffic. As such, CIT link padding fails completely when the adversary uses sample entropy or sample variance for statistical analysis to explore this correlation.

Our theoretical investigation reveals the impact of covered traffic variance on detection rate, indicating that the ideal covered traffic should have sufficient large variance. Following this idea, VIT countermeasure was developed and evaluated. Our experimental data confirms that the VIT scheme indeed minimizes the detection rate and hence achieves high resistance to traffic analysis attacks.

The results of this paper are preliminary. Many extensions are possible. We focused on the case where there are only two classes of payload traffic. Our technique can be easily extended to the case of multiple classes. The major difference is that the adversary needs to perform more training for her system. This paper also assumes that the payload traffic has constant packet size. Recent measurements ([4, 2]) indeed indicate that the size of packets on the Internet follows certain distributions. Extensions can be made to take this factor into account.

References

[1] P. Baran. On distributed communications: Ix security, secrecy, and tamper-free considerations. *Memo RM-3765-PR, Rand Corp.*, Aug. 1964.

[2] CAIDA. Packet sizes and sequencing. *http://www.caida.org/outreach/resources/learn/packetsizes/*, June 2002.

[3] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2), Feb. 1981.

[4] K. Claffy, G. Miller, and K. Thompson. The nature of the beast: recent traffic measurements from an internet backbone. *Proceedings of ISOC INET '98*, July 1998.

[5] M. Coates, A. Hero, R. Nowak, and B. Yu. Internet tomography. *IEEE Signal Processing Magazine*, pages 47–65, 2002.

[6] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Number 31 in Applications of mathematics. Springer, New York, 1996.

[7] R. O. Duda and P. E. Hart. *Pattern Classification*. John Wiley & Sons, 2001.

[8] K. S. Fu. *Applications of Pattern Recognition*. CRC Press, Inc., Boca Raton, Florida, 1982.

[9] X. Fu, B. Graham, R. Bettati, and W. Zhao. An information assurance testing framework for systems under traffic analysis attacks and its application on systems using traffic padding. *Technical Report TR2003-2-1, Dept. of Computer Science, Texas A&M University*, February 2003.

[10] S. Ghosh and R. Rajkumar. Resource management of the os network subsystem. *Proceedings of the Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, April 2002.

[11] Y. Guan, X. Fu, D. Xuan, P. U. Shenoy, R. Bettati, and W. Zhao. Netcamo: Camouflaging network traffic for qos-guaranteed critical allplications. In *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans, Special Issue on Information Assurance*, volume 31 of *4*, pages 253–265, July 2001.

[12] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[13] mitre.org. Information assurance. *http://www.mitre.org/technology/mtp01/info_assurance.shtml*, 2002.

[14] R. Moddemeijer. On estimation of entropy and mutual information of continuous distributions. *Signal Processing*, 16(3):233–246, 1989.

[15] R. E. Newman-Wolfe and B. R. Venkatraman. High level prevention of traffic analysis. *Computer Security Applications Conference, Seventh Annual*, pages 102 –109, 1991.

[16] R. E. Newman-Wolfe and B. R. Venkatraman. Performance analysis of a method for high level prevention of traffic analysis. *Computer Security Applications Conference, Eighth Annual*, pages 123 –130, 1992.

[17] J. Raymond. Traffic analysis: Protocols, attacks, design issues and open problems. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Proceedings of International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *LNCS*, pages 10–29. Springer-Verlag, 2001.

[18] C. E. Shannon. Communication theory of secrecy systems. *Bell Sys. Tech. J.*, 28:656–715, 1949.

[19] B. Silverman. *Density Estimation for Statistics and Data Analysis, Monographs on Statistics and Applied Probability*. Chapman & Hall, London, 1986.

[20] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *IEEE Symposium on Security and Privacy*, pages 44–54, Oakland, California, 4–7 1997.

[21] TimeSys. Timesys linux/real-time user's manual. *http://www.timesys.com/prodserv/docs/tslinux-rt.html*, July 2002.

[22] Y. Vardi. Network tomography: Estimating source-destination traffic intensities from link data. *Journal of the American Statistical Association*, 91(433):365–377, March 1996.

[23] B. R. Venkatraman and R. E. Newman-Wolfe. Performance analysis of a method for high level prevention of traffic analysis using measurements from a campus network. *Computer Security Applications Conference, 10th Annual*, pages 288 – 297, 1994.

[24] V. Voydoc and S. Kent. Security mechanisms in high-level network protocols. *ACM Computing Surveys*, pages 135 – 171, 1983.