Predictable Design of Network-Based Covert Communication Systems

Ronald W. Smith, and G. Scott Knight
Computer Security Laboratory
Royal Military College
Kingston, Ontario, Canada
smith-r@rmc.ca
knight-s@rmc.ca

Abstract

This paper presents a predictable and quantifiable approach to designing a covert communication system capable of effectively exploiting covert channels found in the various layers of network protocols. Two metrics are developed that characterize the overall system. A measure of probability of detection is derived using statistical inference techniques. A measure of reliability is developed as the bit error rate of the combined noisy channel and an appropriate error-correcting code. To support reliable communication, a family of error-correcting codes are developed that handle the high symbol insertion rates found in these covert channels. The system metrics are each shown to be a function of the covert channel signal-to-noise ratio, and as such the two can be used to perform system level design trade-offs. Validation of the system design methodology is provided by means of an experiment using real network traffic data.

1. Introduction

Covert channels present an interesting problem in secure system design. Traditional covert channels and associated research focus primarily on multi-level secure systems [6, 15, 19]. Recognizing that complete elimination of these hidden channels is impossible [20], the classic defense involves some means of ensuring that the bandwidth (throughput) is reduced to some arbitrarily low number [7]. Network-based covert channels differ from classical covert channels in that they exploit properties of computer network communication protocols. In 1987 Girling [11] is perhaps the first to report on network-based covert channels. Over the next two decades several varieties of exploits within the various network protocols are revealed. Examples include [1, 12, 28, 30, 37]; [13, 26] summarize several typical exploits. A rigorous analysis of network-based covert channels has not been pursued to the same extent as the classic covert channels, and defense against these newer covert channels is an open area of research. Notable exceptions to this last statement are revealed in recent publications from the US Naval Research Laboratory on channel capacity analysis [21, 22] and bandwidth restrictions [14].

It has been amply demonstrated that basic storage and timing covert channels exist that will support a low bandwidth one-way communication system in network communications. However, none of the published research has offered a rigorous or unified approach to the design of a predictably quantifiable covert communication system. Detectability of the channel is only expressed in qualitative terms. Informal capacity estimates are often cited, but an information theoretic analysis of the channel capacity and channel error types is typically omitted. The reliability of communications across the channel is rarely expressed or considered, and the application of coding theory has been ignored or ad hoc. Interesting system designs are presented in [1, 3, 4, 10, 31]; however, each suffers from one or more of the above criticisms, and each is based on a single specific type of exploit.

In this paper we present a general purpose methodology for network-based covert communication system design. Specifically we investigate highly stealthy, low-bandwidth applications. We demonstrate that a predictably measurable covert communication system can be designed that exploits certain predictable properties of network communications. We provide a sound engineering foundation for the design of undetectable and reliable communication systems hidden within existing Internet traffic. A low-bandwidth covert channel is formed through exploitation of well-chosen network properties. Selection and design of an exploit is guided by the criteria to minimize probability of detection. Given an exploit design, the channel is characterized for both capacity and noise. Based upon the exploit parameters and the noise characteristics of the channel, an appropriate coding scheme is devised. The combination of these activities yields a covert communication system designed to

achieve predictable levels of both probability of detection and reliability.

Section II describes a general scenario in which such a system may be employed, and introduces the concept of exploit signal-to-noise ratio. Section III presents the the metrics against which the system design can be measured. Mathematical representations for system detectability and reliability are derived. A unique family of trellis codes is presented as a means of handling the high error rates of these channels. Section IV introduces a system validation experiment that demonstrates the effectiveness of the design methodology. Section V concludes the paper.

2. Background

2.1. Applications

Perhaps the most widely held view of a covert channel is of a guarded secret being "leaked" to an unintended audience. This view is consistent with the traditional example of a covert channel in a multi-level secure system where a Trojan Horse surreptitiously modulates some parameter of the system visible to some lower-level process.

The introduction of network-based covert channels increases the scope of vulnerable computing systems. Leaking sensitive information is still a primary goal of most covert channels; however it is not the only application. Traffic analysis is a potentially new application afforded by network-based covet channels; covert channels exposed within assumed secure and anonymous network communications afford the opportunity for an observer to defeat some level of the anonymity [21]. Covert channels opened in one layer of a network may also provide the vehicle for infiltrating higher-level networks. Network-based covert channels have also been demonstrated to provide new attacker tracing techniques [36, 9].

2.2. Exploits, Channels and Systems

In much of the literature a covert channel refers to both the technique used to signal hidden messages as well as the channel formed by such a technique. For the purposes of this research it is helpful to use separate terms for these concepts. The term *covert exploit*, or simply *exploit*, is used to refer to the specific technique used to inject hidden data into the network packet stream. The term *covert channel*, or simply *channel*, is used to refer to the type of theoretical communication channel formed by a given technique. To illustrate the utility of such a naming convention, consider the following example. Manipulation of the lower order bit of the Transport Control Protocol timestamp field or the urgent flag of the control bits field are two potential covert channel exploits, each with its own merits in terms of ease

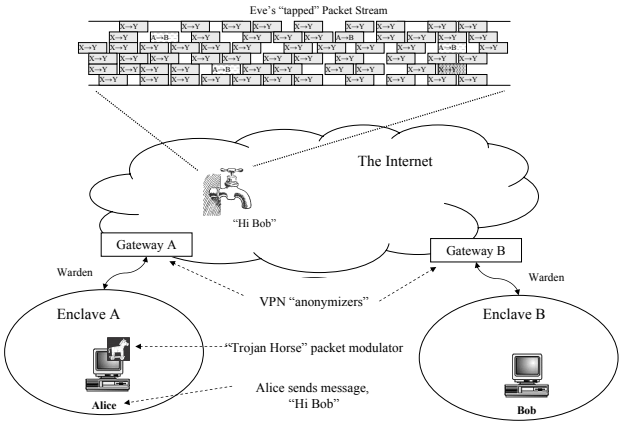

Figure 1. A Representative Network-Based Covert Channel

of implementation and detectability. However, both might yield a very similar covert storage channel in terms of the channel's information theoretic properties.

A *covert communication system* is another term introduced within this research. It refers to the collective set of the covert exploit(s) and channel, the coding scheme, and any other modules necessary to effectively use the channel for communications. The design of covert communication systems is a central theme of this paper, and it is the systems-based approach that, in part, makes this research unique.

2.3. The Attack Scenario

An illustration of an Internet-based covert channel is provided in Figure 1. *Alice*, from Enclave A, communicates legitimately with *Bob*, from Enclave B. The enclaves use the Internet as a cost effective communication link and add some measure of security and anonymity, perhaps a virtual private network layer. Assume that a Trojan Horse is somehow installed on *Alice's* computer. This malicious software now modulates the stream of packets sent by *Alice* in such a way as to allow an eavesdropper, *Eve*, to identify *Alice's* transmissions from those of other hosts within enclave A. A hidden message is then encoded in the modulation scheme thus allowing secret information to be exfiltrated from the enclave. Further, assume that the enclaves are monitored for suspicious and malicious use of the secured network, by a *warden*. The warden is an abstraction of the monitoring agency whose job it is to ensure that the security of the enclave(s) is not jeopardized. In the context of this research, the warden is assumed to be a powerful monitoring agency, one with significant computing resources; the specific assumptions are covered in Section III.

2.4. Signal-to-Noise Ratio

Consider the following generic storage-based exploit. The designer of the system chooses a protocol field for exploit that contains N_F total field values. From this field a subset of N_S distinct exploit symbols is chosen wherein each symbol is represented by some mapping to the exploit field values. Covert signaling is then achieved by injecting exploit symbols into the network traffic according to an encoding scheme. Natural occurrences of the exploit symbols are assumed to exist in the traffic at the point in the network at which the eavesdropper is listening, and therefore, all observed exploit symbols are either natural or injected (signal). It is important that each exploit symbol have a non-zero occurrence in the natural traffic, otherwise detection by the warden would be trivial once unnatural exploit symbols appeared.

Let ν_s denote the natural proportion of exploit symbols in normal traffic; ie, the proportion without any covert signaling present. Let ν_s^* denote the proportion of covert (injected) symbols within the exploited network traffic. During a period of covert signaling, the observable proportion of exploit symbols is $\nu_s + \nu_s^*$, and the ratio ν_s^*/ν_s becomes a key system design parameter; call this parameter the *exploit signal-to-noise ratio* (SNR). A small SNR implies that few covert symbols are added to the network traffic, thus detection of the signaling by the warden is difficult. However, a small SNR also implies a slow transmission rate and poor reliability of the received message by the eavesdropper since *Eve* will have difficulty discriminating the exploit symbols from the naturally occurring ones. Conversely, a very large ν_s^*/ν_s increases the transmission rate and the reliability, but now at the expense of detectability.

3. Covert Communication System Design Parameters

3.1. Detecting a Covert Channel

In this subsection a set of underlying assumptions are made about the *warden*. Based upon these assumptions a measure of the probability of detection is proposed that is a function of the exploit signal-to-noise ratio.

It is impossible to know the extent to which any warden will go to protect an enclave from covert channels. A common assumption under this scenario is that the warden is no more suspicious of one host than another [10]. The implication being that no host within the enclave, including the covert sender, is subject to forensic investigation or is deemed untrustworthy. Without this assumption no exploit is possible, as the sender Trojan may likely be found. A second assumption is that the warden attempts to detect covert messaging by monitoring the traffic across the network. The type of monitoring ranges from simple signature based detection schemes to statistical anomaly detection. Signature based detection techniques are relatively easy to implement and therefore must be assumed to exist on any network of interest to this study. Therefore, any exploit design must be signature-free. A wise warden will also realize this fact, and thus a further assumption is that any meaningful monitoring by the warden must involve anomaly detection techniques. Specifically this implies that the modulation of the exploit field values must not produce a signature and must not appear anomalous in order to be undetectable. It is argued here that traditional anomaly detection techniques will not detect the covert channels proposed herein since by their very design they contain no signature and violate no protocol usage. It is not sufficient however to suggest that they are undetectable. Instead a measure of detectability is proposed under the assumption that if a warden suspected such covert channels, they would employ the most appropriate anomaly detection; even if that is not currently the practice. The detection of an anomaly is by definition a probabilistic event, and it is with this in mind that detectability is explored.

Application of traditional hypothesis testing techniques to covert (steganographic) channels is not new [5]. A common technique within statistical quality control involves comparing some attribute(s) within a known process to the attribute in other instances of the process. This technique allows for subtle changes in the process to be detectable. Recall that in the attack scenario above, the warden is assumed to possess powerful resources, and therefore it is assumed that the monitoring techniques employed by the warden involve inspection of every field of every packet at every protocol layer, and every value within each packet is treated as an inspected attribute. This assumption is extreme, but it provides a good basis for asserting a near worst-case estimate of probability of detection.

Returning to statistical quality control, an operating characteristic (OC) curve is used to describe the ability of an inspection scheme to detect attribute shifts [35]. The OC curve is a plot of the probability of accepting a hypothesis concerning some known attribute versus the true measure of the attribute. To illustrate, consider the operating characteristic curves depicted in Figure 2. The graph illustrates a fraction defect control chart. The underlying process is assumed to have a known fraction defect of value p_0, from which an upper control limit is established; meaning that if a sample is found to contain greater than the number of defects defined by the upper control limit, the process is assumed to be out of control. For this type of chart, the probability that a sample is within the control limit is defined by the binomial distribution [35], and therefore the probability of accepting a sample for some arbitrary value, p, given the true fraction defect, p_0, and sample size, N, is given by

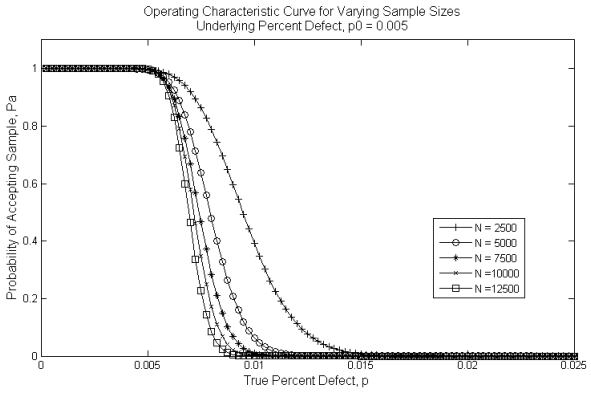

Figure 2. OC Curves for Different Values of Sample Size

$$p_{Accept}(p) = \sum_{x=0}^{UCL} \binom{N}{x} p^x (1-p)^{N-x}, \quad (1)$$

where

$$UCL = \left\lfloor N(p_0 + \kappa\sqrt{(p_0(1-p_0)/N)}) \right\rfloor.$$

From the operating characteristic curves several observations are made. For a fixed underlying fraction defect, the probability of detecting an out of control process increases as sample size increases. The likelihood of false negatives, or the probability that an out of control process is accepted, is given by the probability $p_{Accept}(p)$. The likelihood of false positives, or the probability that an in-control process is found to be out of control, is given by the probability $1-p_{Accept}(p = p_0)$. In the example the upper control limit is based upon a fairly standard usage of *3-Sigma* (κ=3) in control charting. Increasing the control limit lowers the false positives, but at the expense of decreasing the sensitivity of the technique to detect changes in the fraction defect.

Making the parallel between the attack scenario and the statistical quality control technique above, the *known* fraction defect in the context of the exploit is the natural fraction of each exploit symbol expected in normal traffic, ν_s/N_S. The true fraction defect for each symbol during periods of covert transmission is ν_{true}, or $(\nu_s + \nu_s^*)/N_S$. This assumes that the exploit and message symbols are roughly uniformly distributed. The probability that the warden detects the covert signaling, based upon a sample of the traffic during which time the covert signaling is fully present, is equivalent to the probability of rejecting the sample, or $(1 - p_{Accept}(\nu_{true}))$. Therefore the following expression for the probability of detecting an arbitrary exploit symbol is proposed,

$$p_{SymbolDetect}(\nu_{true}) = 1 - \sum_{x=0}^{UCL} \binom{N}{x} (\nu_{true})^x (1-\nu_{true})^{N-x}, \quad (2)$$

where

$$\nu_{true} = (\nu_s + \nu_s^*)/N_S, \text{ and}$$

$$UCL = \left\lfloor N\left(\frac{\nu_s}{N_S} + \kappa\sqrt{\nu_s/N_S(1-\nu_s/N_S)/N}\right)\right\rfloor.$$

Prior to postulating a general expression for probability of channel detection, three considerations remain. A distinction must be made between the probability of detecting an "out-of-control" symbol and the probability of detecting the channel. A criteria for the selection of sample size must be determined. Finally, it is useful for system design if the probability is expressed as a function of the exploit SNR.

Equation 2 represents the probability of detecting occurrences of one type of exploit symbol. Since the covert usage of the channel involves a total of N_S unique exploit symbols, the probability of channel detection is given by

$$p_{ChannelDetect}(\nu_{true}) = 1 - (1 - p_{SymbolDetect}(\nu_{true}))^{N_S}. \quad (3)$$

In the context of quality control, the inspection sample size is chosen so as to maximize the likelihood of detecting a change in an observed process constrained by the classic consumers and producers risks, and by the practicality of the physical inspection. In the case of the network exploit, it is assumed that the traffic is only out of control intermittently; in other words covert signaling only occurs when a message is being sent. Under this assumption, the warden would want to choose sample size to coincide exactly with the duration of the covert signaling. Sampling a larger set would dilute the increased fraction of exploit symbols within the normal traffic, and sampling a smaller set would decrease the probability of detecting the change. The warden has no practical means of knowing such a size or when to begin the sampling window. However, for the purposes of maintaining a worst-case scenario, assume that the warden does in fact sample at this size and time. Let N_o denote this *optimal* sample size and let L denote the size of the message in number of symbols. Assuming the message symbols are roughly uniformly distributed, the fraction of all signal symbols within a transmission window of the optimal sample size is given by

$$\nu_s^* = L/N_o. \quad (4)$$

Combining equations 3 and 4 with 2 and recalling by definition that $\nu_s^* = \nu_s SNR$, yields an expression for probability of detection for a given exploit as a function of signal-to-noise ratio for a given message size, and the number of

unique exploit symbols,

$$p_{ChannelDetect}(SNR) = 1 - (1 - p_{SymbolDetect}(SNR))^{N_S}, \quad (5)$$

$$where$$

$$p_{SymbolDetect}(SNR) = 1 - \sum_{x=0}^{UCL} \left[\begin{smallmatrix} N_o \\ x \end{smallmatrix}\right] \left(\frac{\nu_s(1+SNR)}{N_S}\right)^x \left(\frac{1-\nu_s(1+SNR)}{N_S}\right)^{N_o - x}$$

$$UCL = \left\lfloor \nu_s + \kappa\sqrt{\nu_s(1-\nu_s)/N_o} \right\rfloor, \; and$$

$$N_o = \lfloor L/(\nu_s SNR) \rfloor .$$

In practice, for any chosen exploit field there will be some limited number of values suitable for exploitation, N_S, representing some total natural exploit symbol proportion, ν_s. The probability of detection becomes strictly a function of the ratio of injected symbols to natural symbols (SNR) and the message size.

3.2. Symbol Insertion Error-Correcting Codes

Define channel noise as "any unwanted signal or effect in addition to the desired signal" [33]. Noise sources are varied and depend wholly upon the medium of the channel. Considering the typical exploit scenario described in Section 2, two types of noise may be present on the covert channel. Foremost, there will be symbol insertions since all suggested exploits use naturally occurring exploit field values as potential signals. Given the intent to use the channel with low probability of being detected, symbols arriving at the receiver will therefore be some mix of signal and noise; the greater the stealth, the greater the probability of symbol-insertion noise events.

The other type of noise occurs as a result of packet loss in the underlying (intended) channel. A dropped packet manifests itself as a symbol deletion. Packet reordering within the underlying channel will similarly yield symbol reordering noise in some covert channels. Note that symbol insertion and symbol deletion are sufficient to describe all noise types on this channel. Symbol reordering is not needed as a separate category since it can always be described as a combination of deletions and insertions.

The rates of packet loss and packet reordering are fairly well studied [2, 23, 38]. Typical packet loss values range from 0.1 to 1% while packet reordering typically ranges from 0.6 to 2%. To some extent the effects of these types of noise are controllable by the selection and design of the exploit. However, in order to achieve a high level of stealthy transmission, the injected symbols must remain sparse as compared to those occurring naturally. As a result the symbol insertion rate will be orders of magnitude larger than the deletion and reorder rates above, and thus the overwhelming majority of all noise in the channel will be symbol insertions.

Analyzing the attack scenario, it is clear that no communication is permitted from the eavesdropper back to the sender. This dictates that a forward error correcting (FEC) scheme is required. [8, 16, 24, 25, 32] offer various bit insertion block coding schemes, but are not appropriate for high rates of symbol insertion. Further, trellis codes, of which convolutional codes are the most prevalent, offer several advantages over block coding. Trellis codes employ the use of memory to improve the error-correcting capability of codes [29] over that of block codes. The use of a trellis code allows for a natural mapping of the encoder output bits (code words) onto the exploit symbols, and thus the output of the channel (and input to the decoder) can be interpreted as a stream of symbols, not bits. Finally, trellis codes have an inherent ability to self-synchronize; that is to say that a matched decoder can correctly decode a stream of received blocks without knowledge of the beginning state of the code [34].

Combining the natural symbol usage with the inherent ability to self-synchronize, a convolutional coding scheme can be expected to yield a design that allows for decoding to automatically synchronize anytime the received stream of symbols is error free for at least the constraint length of the code. This is a major advantage for a system where the receiver has no knowledge of when the sender may send a message, and one where the noise is predominantly insertions. The remaining problem is to find a set of good convolutional codes.

Convolutional codes are a well studied category of trellis codes. A careful study of known good convolutional codes will reveal that without significant modification, they are not well suited for symbol insertion correction despite the advantages listed above. Refer to the state machine view of the representative known good convolutional code [17] in Figure 3. This code is said to be a $(n = 4, k = 1, m = 3)$ code; by convention the code has k input bits, n output bits, and 2^m states. Three undetectable symbol insertion error modes are identified for codes of this construction.

a) *Memory-loss* errors occur when symbol insertions cause the code to lose memory as a result of a short *path-to-self*. From an arbitrary state, once a code returns to that state, any memory and thus any error correction ability, is lost. For example, from some arbitrary state of the decoder, the right series of symbol insertions can cause the decoder to move through transitions that return it to that same state. The shorter these paths-to-self, the higher the probability that this associated series of symbol insertions may occur. Memory-loss errors cause a loss of synchronization within the decoded message as indis-

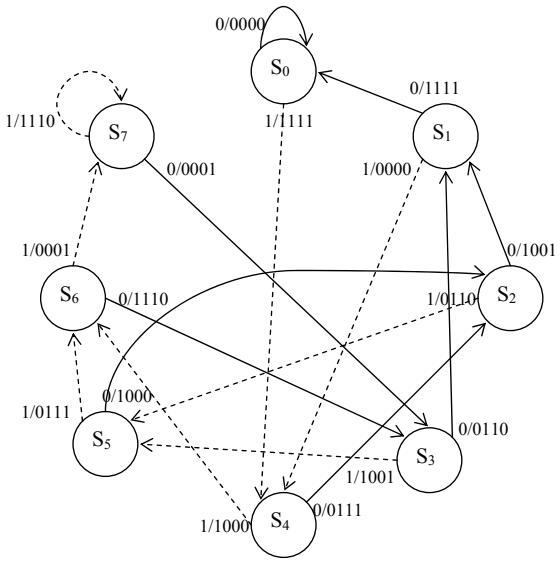

Figure 3. Representation of a Known Good (4,1,3) Code - Finite State Machine

tinguishable "extra symbols" are added to the decode sequence. The code in Figure 3 is very prone to memory-loss errors as seen by the self-transitions (path-to-self of length 1) in states S_0 and S_7.

b) *Equal-alternate-path* errors occur when symbol insertions introduce an alternate but equal length path causing an indistinguishable alternate decode sequence. Equal-alternate-path errors cause a finite number of message bit errors, but do not cause a loss of synchronization within the decoded message as there are no extra symbols added to the decode sequence.

c) *Unequal-alternate-path* errors occur when symbol insertions introduce an alternate but unequal length path causing an indistinguishable alternate decode sequence. Unequal-alternate-path errors do cause a loss of synchronization within the decoded message as there are either extra or fewer symbols added to the decode sequence.

This categorization of undetectable error modes is a key consideration of code construction and theoretical reliability prediction. A family of trellis codes is presented next that are capable of handling the three symbol insertion error modes above. For brevity, the full development of these codes and the derivation of the reliability expression will not be explored here. However, evidence of the capabilities

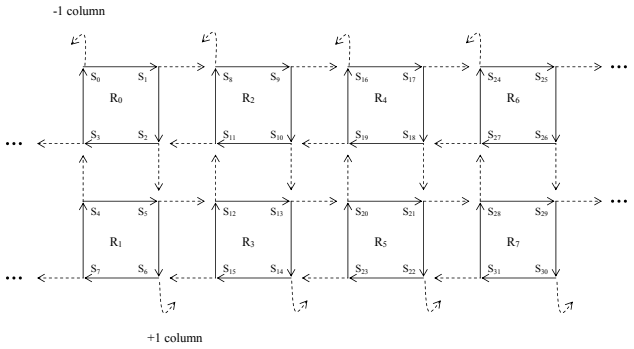

Figure 4. Toroid of Squares Code / (6,1,5)

of the codes and the validity of their theoretical reliability is offered in the next section.

The construction of the state machines characterizing the symbol insertion error correcting codes begins with the creation of a ring of states of size r, defined as a clockwise *zero-path* ring. The code is then constructed of some number of such rings joined together by *one-paths*. This implies that any *zero* input bit results in a transition of the code to states around a ring, and any *one* input bit results in a transition of the code to another ring. Toroidal structures can be formed by linking the rings in such a way that the one-paths wrap to complete the surface of the structure. The total number of states of such a code is restricted to be an exact multiple of r.

The code depicted in Figure 4 is the first of a family of codes constructed as toroidal structures and is referred to as a *Toroid of Squares* code; the name emphasizes its ring shape and size, as well as the overall code structure. Note that the code is a trellis code, and not a convolutional code. Also note that standard code size notation (n, k, m) is continued for convenience, even though a trellis code need not necessarily use all 2^m states. Let the ring (square) in the upper left corner be denoted as R_0, the one below it R_1, the ring to its right R_2, and so on. By construction, every zero-path ring forms a path-to-self of exactly 4 transitions. Table 1 enumerates the one-paths originating from each ring R_0 state. Every one-path also forms a path-to-self of exactly 4 transitions. Note the diversity of rings traversed for each one-path. It can be shown that from any state the minimum path-to-self, equal-alternate-path, and unequal-alternate-path is no less than 4 transitions. Therefore, this code is not vulnerable to an undetectable error mode as a result of any combination of 3 or less symbol insertions.

A family of toroidal codes can be constructed by varying the size (shape) of the rings and total state space. Intuitively the error-correcting capability of the codes increases with increasing ring and state sizes. Several codes of varying sizes have been developed by the authors. Two additional

Start	One-Path	Ring Traversal	Length
S_0	$S_0S_{31}S_{22}S_{25}S_0$	$R_0R_7R_5R_6R_0$	4
S_1	$S_1S_8S_7S_{30}S_1$	$R_0R_2R_1R_7R_0$	4
S_2	$S_2S_5S_{12}S_{11}S_2$	$R_0R_1R_3R_2R_0$	4
S_3	$S_3S_{26}S_{29}S_4S_3$	$R_0R_6R_7R_1R_0$	4

Table 1. Enumerated Ring0 One-Paths of the Toroid of Squares Code

codes are presented in Appendix A.

3.3. Reliability

The capability of a code is generally defined as its ability to detect and correct for errors [27, 17]. This capability of the code, along with an analysis of the channel, can be combined to provide an estimate of the reliability of the code, generally expressed as a bit error rate (BER) [29]. Note that *code* reliability is a function of the noisiness of the channel and it provides a bound on the BER, below which no decoder can achieve guaranteed error free decoding.

In the previous subsection a family of trellis codes were introduced as capable of handling increasing numbers of symbol insertions. The regularized construction of the codes affords that a general expression for reliability is attainable. Specifically each code is designed to be capable of detecting and correcting errors resulting from I or less consecutive insertions; beyond I consecutive insertions, undetectable errors may arise. A key factor in determining the BER of the code is therefore the probability that an undetectable error event will occur. Intuitively the probability of such events is driven by the "size" of the code, and the SNR of the channel.

Assume that at any arbitrary time at the receiver, the current state of the decoder is known. The next symbol received can then be characterized as either *valid* or *invalid*. Valid refers to a received symbol that leads from the current state to a valid next state; invalid refers to a received symbol that does not lead to a valid next state. The valid received symbols can be further subdivided into the one that leads to the correct next state and those that lead to a valid but incorrect next state. The valid symbols can also be subdivided into those that originated as signals and those that originated as noise. Mapping the selected exploit symbols onto the code there are 2^n, or N_S, total symbols, and leaving any state there are 2^k valid symbols. Let p_s and p_{vi} denote the probability that a received valid symbol is either a signal or noise, respectively. Each probability can be expressed in terms of the channel SNR:

$$p_s = 1/(1 + 2^k/(N_S * SNR)), \tag{6}$$

and

$$p_{vi} = 1/(1 + (N_S * SNR)/2^k). \tag{7}$$

Further, under the assumption that all sequences of more than I incorrect valid symbols received before I or less signals will cause an undetectable error event, the probability of an undetectable error event can be computed as follows:

$$p(U) = \sum_{j=I+1}^{2I+1} \left[{}^{2I+1}_{j}\right] p_{vi}^{j}\, p_s^{2I+1-j}. \tag{8}$$

The expression in Equation 8 is independent of code construction beyond the size and its symbol insertion capability rating, I; in fact, it assumes that the code is capable of handling up to exactly I symbols insertions for every path from every state. Therefore a means is desired where an estimate of reliability can be provided that accounts for the uniqueness of each specific code. Considering that all memory-loss errors result in an undetectable error, and under the simplifying assumption that all equal- and unequal-alternate-paths lead to undetectable errors, a new estimate can be provided for the overall probability of undetectable errors:

$$p^*(U) = \rho_{vulnerability} \sum_{j=I+1}^{2I+1} \left[{}^{2I+1}_{j}\right] p_{vi}^{j} p_s^{2I+1-j}, \tag{9}$$

where $\rho_{vulnerability}$ represents the fraction of paths of length $I+1$ that make the code vulnerable to undetectable error modes.

In arriving at a conservative estimate of the bit error rate three further assumptions are applied. The first assumption is that, on average, undetectable errors will be considered to occur at uniformly distributed locations in the received message. The second assumption is that all undetectable errors lead to loss of synchronization beyond the point in the message of the error. This assumption allows for an estimate to be determined without enumerating all possible combinations of error modes when multiple undetectable errors do occur. The third assumption recognizes the fact that when an undetectable error does occur and the decoding is out of synchronization or otherwise faulty, there remains a random chance that each bit is still decoded correctly. Under these assumptions, the following estimate is provided for the bit error rate of the designed codes:

$$BER = \frac{1}{2^k L} \sum_{i=1}^{L} \frac{iL}{i+1} \left[{}^{L}_{i}\right] p^*(U)^i (1 - p^*(U))^{L-i}. \tag{10}$$

To better understand the equation, each major term is revisited:

i) the inner $\binom{L}{i} p^*(U)^i(1 - p^*(U))^{L-i}$ term represents the likelihood of having exactly i undetectable events times the number of ways in which this exact number could occur in a message of size L;

ii) the $(iL/(i+1))$ term estimates the number of bits in the message which are subject to loss of synchronization under the assumption that the i error events are uniformly distributed throughout the message;

iii) $1/L$ normalizes to yield bit error rates; and

iv) $1/2^k$ accounts for the fact that even under the assumption of lost synchronization, the decoder still has this random chance of getting the bit correct.

3.4 Representative Symbol Insertion Error Correcting Codes with Results

The theoretically predicted and experimental BERs for a sample of the developed codes is presented next. The first code illustrated in Figure 5 is referred to as a *Toroid of Hexagons* code. This binary (input) code consists of 10 hexagons for a total of 60 states and 120 output symbols. Each hexagon (ring) defines the *zero-paths*, and thus the minimum length *zero-path* is 6. Again the *one-paths* provide the transitions from hexagon to hexagon, with these traversals designed so as to yield long paths-to-self as well as long equal- and unequal-alternate paths, again of length 6 or longer. The predicted and empirical reliability of this code is provided in Figure 6.

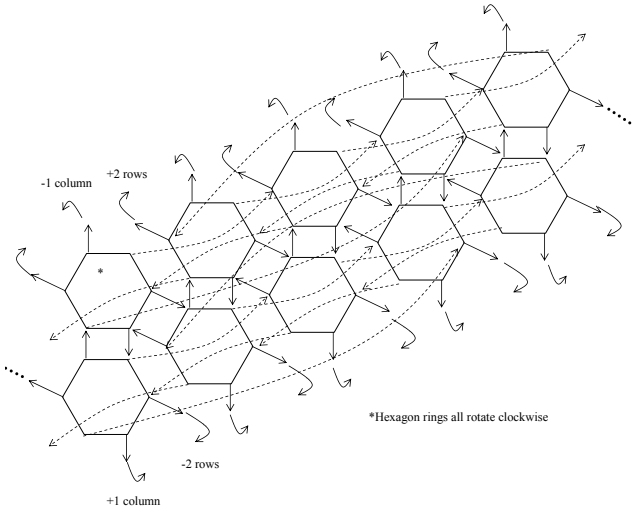

Figure 5. Toroid of Hexagons / (7,1,6)

The next code presented in Figure 7 is the *Square Toroid of Octagons* code, a binary code consisting of a square grid of 16 octagons with a total of 128 states and 256 output

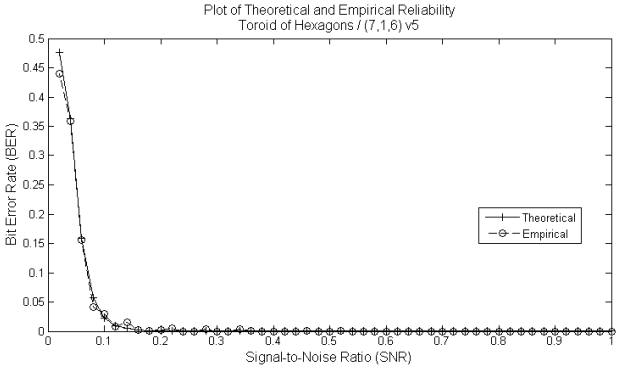

Figure 6. Theoretical and Empirical Reliability - Toroid of Hexagons

symbols. As the ring size and the total state size increases so too do the path lengths, yielding codes with ever-increasing symbol insertion error correction capabilities. This is evidenced in the reliability results presented in Figure 8.

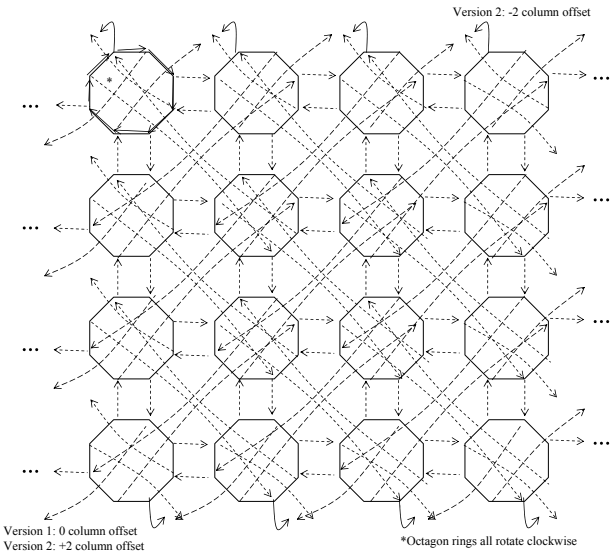

Figure 7. Square Toroid of Octagons / (8,1,7)

4. A System Design with Experimental Results

This section will describe a system design and an experiment that illustrates the general steps to the design of a covert communication system and validates the theoretical results. In summary, the steps of the design process are: selection of the exploit field, selection of an appropriate set of exploit symbols from within the exploit field, characterization of the exploit, characterization of the channel, selection of an appropriate code, and selection of an appropriate

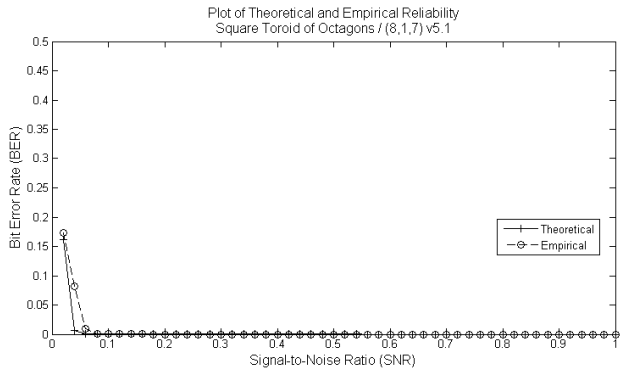

Figure 8. Theoretical and Empirical Reliability - Square Toroid of Octagons

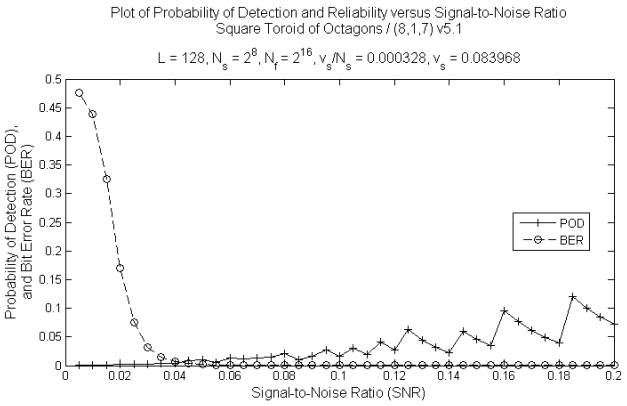

Figure 10. Predicted System Results as a Function of SNR

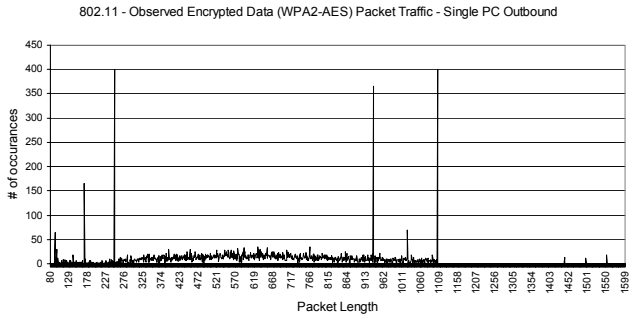

Figure 9. Representative Packet Sizes in Wireless Encrypted Traffic [18]

message length. The system is validated by transmitting arbitrary messages across the channel and observing the empirical results as compared to those predicted.

The exploit field chosen for the experiment is the 16-bit packet size field of a link layer protocol implementation for encrypted wireless data. Figure 9 depicts the number of occurrences of each packet size value for a period of approximately 40 minutes of traffic as recorded outbound from a single host. Approximately 35,000 total packets were recorded.

An analysis of the packet size data reveals a set of about 260 good candidate packet sizes for use as exploit symbols. Among the set of candidate symbols, the probability of occurrence of each symbol ranges from 0.000266 to 0.000387 with an average of 0.000328. The resultant covert channel is typical of that described above, namely that it is characterized as one where the predominant error type is that of symbol insertions.

The selection of an appropriate code is guided by the total available candidate exploit symbols. Ideally the code will use as many of the 260 exploit symbols as possible. A $(8, 1, 7)$ code is the most obvious choice, since the maximum number of output symbols is given by 2^8, or 256. For the experiment, the *Square Toroid of Octagons* from Appendix A is a good choice as it uses the full 256 exploit symbols and is capable of handling a high number of insertions, namely 7. The total number of exploit symbols, N_s, is known and the proportion of naturally occurring exploit symbols, ν_s, is computed as the number of exploit symbols times the average probability. Specifically, ν_s = 0.0839. The last system design parameter to be selected is the message length. A good heuristic for choosing the message length is to choose a value that is half the total number of exploit symbols, or 2^{n-1}. The message length is therefore set at 128 bits.

From the design decisions above, the system metrics can now be predicted. Figure 10 illustrates the predicted probability of detection and reliability as a function of exploit signal-to-noise ratio. The operational SNR is defined as the cross-over between detectability and reliability. From the graph this is determined to be 0.045; where the probability of channel detection is estimated to be 0.81% and the bit error rate is estimated to be 0.36%.

With an operational signal-to-noise ratio of 0.045 this implies that the sending host injects exploit symbols at a rate of 45 per 1000 naturally occurring exploit symbols. The rate of natural occurrence of exploit symbols is the overall exploit symbol proportion, ν_s, and from above is 8.39%. Therefore overall injection rate becomes 0.00378. Interpreting this result, a symbol injector should release signals (symbols) into the background traffic at a rate of about 38 per 10000 packets. Completing the analysis, it is known from Equation 5 that the optimal sample size for the warden is given by

$$N_o = \lfloor L/(\nu_s SNR_o) \rfloor \approx 33,900 \; symbols(packets).$$

Implementing the experiment involves transmitting an arbitrary 128-bit message across this representative channel. A channel encoder uses the *Square Toroid of Octagons* code to encode the message. A channel modulator maps the outputs of the encoder (code symbols) one-to-one onto the chosen exploit symbols. A symbol injector releases one signal for roughly every 263 packets of background traffic. At the receiver, a channel demodulator extracts from the received packets only those containing exploit symbols; in other words only those where packet size values are from the exploit symbol set. A channel decoder takes as input the sequence of exploit symbols and decodes until it finds a message of length 128; the message length is an assumed shared secret between sender and receiver.

The experiment was run 100 times. The empirical reliability was computed as the total number of bits decoded correctly over the total number of bits decoded. From this the observed BER was 0.25%. To confirm the probability of detection the observed number of each exploit symbol was recorded and then compared to the upper control limit, UCL, from Equation 5. If any exploit symbol count exceeded the UCL, it is considered an indicator of detection. For each trial, the observed number of each exploit symbol within the optimal sample size was compared to the UCL. Over the 100 trials, a total of 0 detections were observed across all exploit symbols. For the 100 trials conducted, the maximum counts of any exploit symbol ranged from 17 to 20, and in only a single instance were 20 exploit symbols counted; 65% of the time the highest count was 18, well below the UCL limit of 21.

5. Conclusion

A unique design methodology for network-based covert communication systems has been proposed. A quantitative measure of the probability of detection has been developed for a generic storage-based network covert channel. A general methodology for the design of error-correcting codes for high numbers of symbol insertion errors has been demonstrated, and a family of error-correcting codes have been developed based upon the methodology. A general expression for the reliability of the family of codes has been developed. Finally, a system design experiment has demonstrated the feasibility and predictability of the approach; the BER was predicted to be 0.36% while observed to be 0.25%, and the detectability was predicted to be 0.81% while observed to be 0%.

An expression for the efficiency (or throughput) of the system has also been derived by the authors. The type of channel formed by the experimental design yields a low system efficiency; at most one bit of message is transmitted in each packet sent. Designs of both higher efficiency exploits and codes is currently being pursued. Extensions to the trellis codes to support extremely low levels of SNR are being investigated. Techniques for detecting these types of channels are also being explored.

Acknowledgment The authors would like to thank Dr Paul Van Oorschot of Carleton University for his insightful comments on this work. This work was supported, in part, by MITACS through its research project entitled, Understanding and Mitigating Malicious Activity in Networked Computer.

References

[1] C. Abad. Ip checksum covert channels and selected hash collision. Web Site. http://www.gray-world.net/papers/ipccc.pdf, last visited April 2006.

[2] M. Allman and V. Paxson. On estimating end-to-end network path properties. In *ACM SIGCOMM 99*, Sept. 1999.

[3] K. Ashan. Covert channel analysis and data hiding in TCP/IP. Masters thesis, Department of Electrical and Computer Engineering, University of Toronto, 2002.

[4] S. Cabuk, C. Brodley, and C. Shields. IP covert timing channels: Design and detection. *CCS 2004*, Oct. 2004.

[5] C. Cachin. An information-theoretic model for steganography. In D. Aucsmith, editor, *Proceedings of 2nd Workshop on Information Hiding*, volume 1525, Portland, Oregon, USA, May 1998. Lecture Notes in Computer Science, Springer.

[6] T. N. C. S. Center. A guide to understanding covert channel analysis of trusted systems. NCSC-TG-030, Library No.S-240,572, Version 1, Fort George G. Meade, Maryland, Nov. 1993.

[7] N. Computer Security Center. Covert channel capacity limit policy. NCSC Interpretation of TCSEC0113. http://www.radium.ncsc.mil/tpep/library/interps/0113.html, last visited June 2004.

[8] M. Davey and D. MacKay. Reliable communication over channels with insertions, deletions and substitutions. *IEEE Transactions on Information Theory*, 47(2):687–692, Feb. 2000.

[9] G. Drawson. Temporal interval modulation: An investigation into the modulation of packet timings as a means of generating a detectable watermark. Masters thesis, Royal Military College of Canada, Apr. 2002.

[10] J. Giffin, R. Greenstadt, P. Litwack, and R. Tibbetts. Covert messaging through TCP timestamps. In *Workshop on Privacy Enhancement Technologies (PET)*, Apr. 2002.

[11] C. Girling. Covert channels in lan"s. *IEEE Transactions on software Engineering*, SE13(2):292–296, Feb. 1987.

[12] T. Handel and M. Sandford. Hiding data in the OSI network model. In *First International Workshop on Information Hiding*, June 1996.

[13] D. Hintz. Channels in TCP and IP Headers. PowerPoint Presentation. http://guh.nu/projects/cc/, last visited November 2007.

[14] M. Kang, I. Moskowitz, and D. Lee. A network version of the pump. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 144–154, 1995.

[15] B. Lampson. A note on the confinement problem. *Communications of the ACM*, 1973.

[16] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady Akademii Nauk SSSR*, 163(4):845–848, 1965. English translation in Soviet Physics Doklady, vol. 10 no. 8 pp:707710, 1966.

[17] S. Lin and D. C. Jr. *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, New Jersey, 1983.

[18] P. Martin. Covert channels in secure wireless networks. Master's thesis (draft) not available as of March 2007, Royal Military College of Canada, 2007.

[19] J. Millen. 20 years of covert channel modeling and analysis. In *IEEE Symposium on Security and Privacy*, Oakland, California, May 1999.

[20] I. Moskowitz and M. Kang. Covert channels - here to stay? In *Proceedings COMPASS '94*, pages 235–243, Gaithersburg, MD, 1994.

[21] I. Moskowitz, R. Newman, D. Crepeau, and A. Miller. Covert channels and anonymizing networks. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2003)*, Oct. 2003. http://chacs.nrl.navy.mil/publications/CHACS/2003/2003moskowitz-acm5.pdf,last visited April 2006.

[22] I. Moskowitz, R. Newman, and P. Syverson. Quasi-anonymous channels. In *Proceedings CNIS 2003*, Dec. 2003. http://chacs.nrl.navy.mil/publications/CHACS/2003/2003moskowitz-5_3.pdf,last visited April 2006.

[23] V. Paxson. End-to-end internet packet dynamics. *IEEE/ACM Transactions on Networking*, 7(3):277–292, June 1999. An earlier version appeared in Proc. ACM SIGCOMM Ś97, September 1997, Cannes, France.

[24] E. Ratzer. Marker codes for channels with insertions and deletions. to be presented at the 3rd International Symposium on Turbo Codes and Applications, Munich, Germany, Apr. 2006.

[25] E. Ratzer and D. MacKay. Codes for channels with insertions, deletions and substitution. In *In 2nd International Symposium on Turbo Codes and Related Topics*, pages 149–156, Apr. 2000.

[26] F. Raynal. Covert channels. PowerPoint Presentation. http://www.securitylabs.org/Download/lsm2003-Raynal.pdf.gz, last visited April 2006.

[27] S. Roman. *Introduction to Information and Coding Theory*. Springer Verlag, 1996.

[28] C. Rowland. Covert channels in the TCP/IP Protocol Suite. *first Monday - a peer-reviewed journal on the Internet*, 2(5), May 1997.

[29] R.Togneri and C. deSilva. *Fundamentals of Information Theory and Coding Design*. Chapman & Hall, 2002. TdS02.

[30] M. V. S.D. Servetto. Communication using phantoms: Covert channels in the internet. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Washington, DC, June 2001.

[31] G. Shah, A. Molina, and M. Blaze. Keyboards and covert channels. In *Proceedings of the 15th USENIX Security Symposium*, Vancouver, Cananda, Aug. 2006. ACM Press.

[32] T. Swart and H. Ferreira. A note on double insertion/deletion correcting codes. *IEEE Transactions on Information Theory*, 49(1):269–273, Jan. 2003.

[33] R. Togneri and C. deSilva. *Fundamentals of Information Theory and Coding Design*. Chapman & Hall, 2002.

[34] A. Viterbi and J. Omura. *Principles of Digital Communications and Coding*. McGraw-Hill, New York, 1979.

[35] H. Wadsworth, K. Stevens, and A. Godfrey. *Modern Methods for Quality Control and Improvement*. John Wiley & Sons, 1986.

[36] X. Wang, D. Reeves, S. Wu, and J. Yuill. Sleepy watermark tracing: An active network-based intrusion response framework. In *Proceedings of the IFIP TC11 Sixteenth Annual Working Conference on Information Security*, pages 369–384, June 2001.

[37] M. Wolf. Covert channels in LAN protocols. In *Proceedings of the Workshop on Local Area Network Security*, pages 91–102, 1989.

[38] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the constancy of internet path properties. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, Nov. 2001.