Practical Covert Channel Implementation through a Timed Mix-Firewall

Richard Newman
Department of Computer & Information Science & Engineering
University of Florida
Gainesville, FL 32611-6120
nemo@cise.ufl.edu

Ira S. Moskowitz
Center for High Assurance Computer Systems—Code 5540
Naval Research Laboratory
Washington, DC 20375
ira.moskowitz@nrl.navy.mil

Abstract
A theoretical channel to communicate covertly through Mix-firewalls has been discussed in previous work. This paper describes implementing this theoretical construction in an actual system. Practical issues involved in implementation are analyzed, and an actual design is given here. Suggested parameters for concatenated codes for the experimental channel are also provided, based on experimental results. We compare our numerically computed information theoretic results for the data rate of covert channels through timed Mix-firewalls to the data rate obtained by our actual construction of these covert channels.

1. Introduction
Mixes are often used in an attempt to hide communication patterns [1]. A timed Mix collects encrypted messages padded to the same length, re-encrypts them, holds them until its timer expires, then sends them in scrambled order. From appearances alone, it is impossible to associate an incoming message from an outgoing message. When combined with a firewall, a Mix-firewall is formed, in which messages from inside a protected enclave are collected and sent in a burst to the Mix-firewall of another enclave. Here, the adversary is only able to observe the messages sent from one Mix-firewall to the other and the time they are sent. Since the messages are indistinguishable, and all have the same observable source and destination (the source and destination Mix-firewalls), the adversary, Eve, can only count the messages she sees each time the Mix-firewall fires. A malicious insider, Alice, can attempt to communicate to Eve by modulating the number or rate at which she sends messages, thereby affecting the message counts observed by Eve.

In the model used in earlier theoretical work [2, 4, 5], the senders behind the Mix-firewall are able to send at most one message per Mix firing interval (tick). There may also be other senders (Clueless) aside from Alice, whose transmissions introduce noise into the covert channel between Alice and Eve. Earlier work provides information-theoretic limits on the rate (in bits per tick) that Alice can send data covertly to Eve, depending on the number and transmission characteristics of the Clueless.

This work demonstrates the feasibility of exercising the covert channel hypothesized and analyzed in [2]. While the experimental data rates are (predictably) not as high as the maximum rates predicted by the earlier theoretical analysis, the covert channel demonstrably exists and can be exploited. The actual rate of the channel (in bits per second) will depend on the firing rate of the Mix-firewall (as this determines the rate at which raw symbols are sent), the number of Clueless (as this affects the potential amount of noise), and the behavior of the Clueless (as this determines the extent to which the potential for noise is realized in the system).

Figure 1 shows the basic model, Alice and the Clueless (which is the set of Clueless, i = 1, . . . , N) are in Enclave 1. For clarity, we will call the information sent from Alice to Eve over the covert channel the message, while the transmissions sent by Alice (or not) in a Mix firing interval will be called packets. Figure 1 shows that all outgoing packets from Enclave 1 go through a Mix-firewall into Enclave 2 via a second Mix-firewall, which sends the packets to the
appropriate receivers in Enclave 2. All, theoretically, Eve can do is count packets as they go by.

Alice sends a packet in a firing interval to send a 1 on the raw covert channel, or refrains from sending a packet during a firing interval to send a 0 on the raw covert channel. These are the raw symbols sent by Alice. The raw symbols received by Eve are the packet counts per burst, which are affected by the raw symbol sent by Alice and the activity of the Clueless, which appears as noise on the covert channel.

In order to achieve essentially error-free communication in spite of great amounts of noise, Alice and Eve must employ a number of techniques. If \( R \) raw covert channel bits are used to send a logical symbol using copy coding with redundancy \( R \) (or equivalently, if senders may send up to \( R \) packets in a firing interval), Alice will have to provide logical symbol synchronization. Alice will also have to delimit the message and provide Error Correction Coding (ECC) with redundancy \( C \) (inverse of the coding rate) since the raw channel can be very noisy if there are any active Clueless present. The ECC may be a single code, concatenated code, or turbocode, for example. For block codes, Alice will also need to provide frame synchronization. The issues and solutions to the problems of synchronization and error handling are explained below.

The next section describes the experimental setup, though the details of the actual implementation are scant due to space constraints. Section 3 gives the implementation issues and solutions for the practical covert channel, and Section 4 describes the results, including our recommendations for code combinations to handle errors. We conclude in Section 5 with a summary and some open questions.

### 2 Experimental Setup

An actual Mix-firewall was implemented and a testbed using four computers and three LANs was set up to test the system. We determined that it worked, and that the covert channel could be exercised through it. However, it took about one second per raw bit, or about 2 hours to attempt to send a single, 100-bit message at typical coding rates. Having accomplish the proof of concept for the raw channel, we desired to know how best to handle the noise on the raw channel in practice. In order to run millions of tests, we implemented a simulation system that allowed us to experiment with synchronization and coding methods more efficiently.

In the theoretical work [2] the Clueless are denoted as Clueless, \( i = 1, \ldots, N \). Each Clueless, is a Bernoulli random variable with parameter \( p \), and they are all i.i.d.s. Therefore the aggregate can be modeled as a Binomial random variable with parameters \( (N, p) \). So every time the sending Mix fires, it is modeled as sending out a burst of size \( S + 0 \), or size \( S + 1 \); \( S = 1, \ldots, N \). \( S \) is determined by the aggregate behavior of the Clueless transmitters, and the +1, or not, is determined by the actions of Alice.

Of course modeling in such a manner is a gross simplification, but we take an approach similar to that of a non-informed prior in statistics. That is, we do not bias any one transmitter more than another. Of course, in future work we plan to relax this assumption and to also consider traffic loads.

### 3 Covert Channel Implementation

Practically, Eve is able to note bursts of messages, and hence to synchronize with the Mix firing interval and to count the number of packets sent in a tick. Once Eve has ascertained the Mix firing interval, she must determine if the channel is being exercised by Alice, and if so, achieve symbol synchronization. Section 4.1 considers how to deal with various numbers of Clueless and for Eve to determine the channel coding parameters if they are not known a priori. If Alice uses \( R > 1 \) raw symbols to encode a single logical symbol, then Eve must also determine the redundancy (if \( R \) is variable) as well synchronize with the boundary of these logical symbols. Once Eve can receive logical symbols, she reverses Alice’s other processes to receive the actual covert channel message.

In our implemented system, Alice concatenates a convolutional outer code with constraint length 7 over an inner copy code. We vary the code rate of both.

Now consider how Eve can synchronize with Alice. Eve buffers a certain amount of data (packet counts per burst) at her end. Using autocorrelation analysis over the buffer, Eve can determine the most likely redundancy \( R \) for the copy-coded symbols. The copy code parameter can be ascertained by the periodicity of the autocorrelation during synchronization. In fact, through experimentation, once Eve knew the copy code parameter \( R \), we found that it was much more reliable to take the standard deviation over a window of \( R \) samples. When Alice is sending the synchronization preamble (01010101...), the standard deviation will peak when half of the window is in the region where Alice is sending a 1, and half in the adjoining region where Alice is sending a 0. Likewise, the standard deviation will have a minimum when all of the samples in the window are in a
region where Alice is sending the same value (all 0s or all 1s). Even if the exact copy code parameter is not known to Eve, this approach yields distinct peaks as the window crosses the boundary between a 0 region and a 1 region, so Eve can use a first phase with a small window to discover the copy code parameter \( R \), and then use \( R \) to perform synchronization with the copy code.

For any detected packet count between \( R \) and \( N \), Eve cannot be certain whether Alice sent a logical 0 or 1. Eve should use the Bayesian threshold (maximum likelihood) if she must make a hard decision (assign a 0 or a 1 to the logical symbol). Without knowledge of \( N \) or \( p \), Eve can determine the long-term average of the number of packets sent per firing interval. As long as the number of 0s and the number of 1s sent by Alice are equal\(^1\), this will be a good estimate of the threshold to determine whether or not Alice sent a 0 or a 1 for hard input decoding. When the copy code boundaries have been found, Eve can output a 0 for the logical symbol if the windowed average is below the threshold or a 1 if it is above the threshold. This is “hard” decoding of the copy coded bits.

A significant improvement can be made if Eve inputs soft data to the error-correcting code (i.e., Eve provides a confidence level that the bit is a 0 or a 1) to the error-correcting decoder (e.g. [7]). This requires Eve to assign a probability that a given observation represents a 0 or a 1. \((n, k, m)\) convolutional codes (e.g., [6]) with log-likelihood ratio (LLR) soft input were used for this purpose to good effect. To compute the LLRs, the channel was analyzed for a given combination of \( N \), \( p \), and \( R \). Alice’s input is \( X \), which is either 0 or 1 (logical), and is repeated \( R \) times. The \( N \) Clueless each may transmit between 0 and \( R \) times to contribute to the observation, \( Y \), which is the sum of all the packets sent in the \( R \) ticks. First, the probability that a particular \( Y \) would be observed given \( X = 0 \) was computed. This is just

\[
Pr(\text{Eve sees } Y \mid X = 0) =
\]

\[
Pr(\text{Eve sees } Y + R \mid X = 1) = \binom{NR}{Y} p^Y (1-p)^{NR-Y}
\]

This is also the probability that Eve sees \( Y + R \) when Alice sends \( X = 1 \), since this is the same distribution shifted over by \( R \). Eve sees some output \( Y \), which has contributions from \( X = 0 \) and \( X = 1 \). The ratio of these contributions is the likelihood ratio, and the log base 2 of this number (if not zero or infinity) is the LLR.

To achieve message synchronization, Eve must determine the first bit of the actual message. To do this, Alice first sends the 0-1-0-1-... preamble so Eve can determine the redundancy and logical symbol boundaries of the copy-coded channel, as well as estimate the product \( Np \) (for thresholding). Then Alice sends a Unique Word (UW) distinct from the synchronization preamble pattern, and with low autocorrelation values so that it can be reliably detected. The pattern 11011000 was used as a UW in our experiments (justified below). Since bit errors are common when trying to interpret the output of the copy code, it is desirable that the UW be recognized even if a bit is in error. Failing to recognize the UW results in missing a frame, while incorrectly detecting the UW results in an attempt to decode a non-message sequence. The latter is handled by including a frame sequence check (cyclic redundancy check) at the end of the message, but the former results in frame loss.

The choice of 11011000 as the UW allows us to accept all 1 bit variations of the UW as valid UWs and thus reduces the chances of a frame miss caused due to a lost or garbled UW. To put this into perspective, if the (logical) bit error rate after decoding from the copy code is 20\%, then the probability that all 8 bits of the UW are received correctly is only about 17\%, for a false negative rate of over 83\% (i.e., over 83\% of the frames will be lost, regardless of ECC strength). Allowing a match when there is one bit error boosts the probability of reception to a little over 50\%, so the false negative rate is below 50\%. At the same time, the false positive rate increases from less than 1 in 1000 to about 0.26\%. False positives can be managed by the error detection code on the message, so these are not fatal, but they are a wasted effort.

4 Covert Channel Experimental Results

This section presents experimental data collected from numerous test runs on our testbed. The data portion of the message has 100 bits in all the test runs. Tests were run with the number \( N \) of Clueless varying between 1 and 15 and the probability that a Clueless sent a message in a given tick varying from \( p=0.1 \) to 0.9, in increments of 0.1. A variety of copy code redundancies ( \( R \) ) and convolutional code rates were used to determine the bit error rate (BER) in the decoded frames, and also computed the \( 1 \rightarrow 0 \) and \( 0 \rightarrow 1 \) bit error rates and the message error rate.

The product of the copy code redundancy \( R \) and the convolutional code redundancy \( C \) is the overall coding redundancy (the inverse of coding rate). Tests were run for copy code redundancies of \( R=1 \) to 15 and convolutional code redundancies from \( C=2 \) to 10. The highest overall coding rate (lowest redundancy) that resulted in zero message errors was recorded over all combinations tested, for each combination of number of \( N \) and \( p \). One thousand runs was found to produce stable results. For these “best” cases, the copy code rate and the convolutional code rate were determined.

Surprisingly, the results showed that more redundancy should be put into the copy code than the convolutional

---

\(^1\)By encrypting and/or compressing the messages, it is likely that the 0’s and 1’s will be balanced; however, Alice can use a code that deliberately does this, such as Manchester Coding.
code, at least when \( p \) is near 0.5. Graphs of the results for hard and soft input decoding are shown in Figure 2. These graphs show that the overall coding rate, and hence the covert channel rate, decreases as the number of Clueless increases, that it also decreases as the probability that a Clueless sends a packet during a tick approaches \( p = 0.5 \). Comparison of the results for hard input (from the copy code output) and soft input show that the soft input convolutional code has significantly better performance, as expected. The rates obtained experimentally are about 0.17 (bit per tick) for \( N = 2 \) and \( p = 0.1 \), 0.06 for \( N = 2 \) and \( p = 0.5 \) (soft decoding). For hard decoding, the rates were about 0.11 for \( N = 2 \) and \( p = 0.1 \), and 0.04 for \( N = 2 \) and \( p = 0.5 \), showing the benefit of soft decoding of the copy code. Again, these are not true “error-free” rates — they are just the best rates we obtained in which there were no errors observed. Numerically computed rates based on information theory [2] give the best rate for \( N = 1 \) and \( p = 0.5 \) as 0.5 (bit per tick), and about 0.3 for \( N = 2 \) and \( p = 0.5 \). With finite length and sub-optimal codes, this is not unexpected. Generally, the best rates obtained experimentally are 1/3 to 1/10 the information theoretic predicted theoretical maximum rates (capacity).

The characteristic “U” shape (that is, \( C(p) \) decreases as \( p \) approaches 0.5, where \( C(p) \) is the capacity for input probability \( p \)) of the graphs giving the coding rate as a function of \( p \) are comparable to the numerical capacity computed in [2], although much lower. In both [2, 3] numerical evidence and plots attempted to show that this “U” shape is standard. However, in [2, 3] the authors attempted to put this on firm theoretical ground. Unfortunately, there is a gap in the proof of [2, 3, Thm. 4 & Cor. 1] where the authors assumed that the mutual information, for a fixed input distribution, is convex up. Due to the non-linear nature of the channel matrix for multiple Clueless, that assumption is non-trivial and needs to be proved. However, we still (optimistically) accept, with out proof, the “U” shaped behavior as stated in [2, 3, Cor. 1].

4.1 Estimating the Number of Clueless

For the best results (i.e., highest error-free communication rate), Alice must know the number of Clueless. She may be able to detect this directly by eavesdropping (she is inside the enclave), or she may have some \textit{a priori} knowledge. Since the number of Clueless affects the amount of “noise” in the channel, it determines the information theoretic channel capacity and hence sets an upper bound on the code rate for forward error correction codes used. The danger of overestimating the number of Clueless is that the data rate of the channel will be considerably lower; the danger of underestimating the number of Clueless is that the code selected is not powerful enough to correct all the errors en-
countered due to the additional noise, so the messages may not be delivered error-free. It is wise for Alice to err on the side of overestimating $N$.

Figure 3 shows the effect of $N$ on the average BER (taken over 1,000 runs) for copy code redundancy $R=3$ and various convolutional code rates. If, for example, Alice had chosen $R=3$ and $c=1/5$, she should usually be able to transmit without errors to Eve if there are 3 or fewer Clueless present. If there are actually 6 Clueless, then Eve will experience a bit error rate around 4%, after decoding, giving less than a 2% chance that a 100-bit message will be error-free.

Figure 4 holds the convolutional code rate constant at 1/5, and shows the effect of $N$ on the average BER (taken over 1,000 runs) for copy code redundancy $R$ ranging from 4 to 10. If Alice had chosen $R=4$ and $c=1/5$, she should usually be able to transmit without errors to Eve if there are 5 or fewer Clueless present. If there are actually 10 Clueless, then Eve will experience a bit error rate around 9%, after decoding, effectively eliminating communication.

The optimum combination of copy code and convolutional code depends on $N$ and $p$, and is decided by Alice. It is beneficial for Alice to be able to adjust $R$ and $C$ according to her estimates of $N$ and $p$. Eve determines the copy code used by Alice when she synchronizes with Alice as noted above. As long as the copy code parameter $R$ uniquely determines the convolutional code parameter, then Eve can adjust her decoding behavior according to Alice’s choice of $R$. Table 4.1 gives our experimentally derived recommendations for copy code and convolutional code redundancy combinations as a function of $N$ so that $R$ uniquely determines $C$. While these code combinations do not always have the lowest error-free $RC$ product, they do allow Alice to signal Eve which convolutional code to use when decoding.

5 Conclusions

This work has provided proof-of-concept for the model covert channel first proposed in [2]. The channel demonstrably exists, and can be exercised, though the practical error-free rate is much lower than the information theoretic rate derived earlier. Using an inner copy code on the raw covert channel proved advantageous, and we provide combinations of copy code rate and convolutional code rate that should usually achieve error-free transmission. The recommended combinations allow Alice to select the code rate according to her perception of Clueless activity, and for Eve to determine the code rate by her observations. In addition, we give a unique word that allows Eve to recover synchronization even when there is considerable noise on the channel.

While the restriction that each host send at most one packet per tick is certainly artificial, this limitation is more of a burden to Alice and Eve than to the Clueless (that is,
Table 1. Recommended Combinations for $R$ and $C$ as Function of $N$

<table>
<thead>
<tr>
<th>Num Clueless $N$</th>
<th>Soft Decoding</th>
<th>Hard Decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>12</td>
<td>11</td>
<td>8</td>
</tr>
</tbody>
</table>

it is harder for Alice and Eve to exercise the covert channel successfully than it would be if Alice could send more packets per tick). If at most $k$ packets can be sent by each host per tick, then this should have an effect similar to having a copy code of length $k$ each tick (assuming that the activity of the Clueless remains the same as before, but with finer granularity). This is worth further investigation, as realistic Mix-firewalls are almost certain to have periods much longer than the time it takes to send a single packet.

From the standpoint of Alice and Eve, it would be of interest to determine what the best approaches are for exploiting the channel efficiently – are there ways better than using a copy code with another error-correcting code above it to send messages reliably? Our initial results here indicate that copy codes with convolutional codes are effective at higher rates than convolutional codes alone, so this warrants further examination.

Also, the Mix firing condition may be threshold-based instead of timed, or may be a combination of these. How best to exploit the channel and what the achievable rates are under these types of Mix-firewalls is of practical interest, as these types of Mixes are common.

Finally, the Mix-firewall may inject dummy packets into its transmissions. Certainly, if the Mix-firewall always pads traffic out to a fixed, constant level, there can be no covert channel as the channel matrix will have only one output regardless of the input. However, this is expensive, and Mix deployers are loathe to do much padding, if any at all. The trade-offs between traffic padding and covert channel disruption are of great interest, as knowledge of these will allow rational choices in Mix design to affordably achieve desired protection.

Acknowledgements We thank Mahendra Kumar, Piyush Harsh, and Prashant Jayaraman, who were involved in simulation and implementation of these systems. We also appreciate the help of Will Snook when he was at NRL, and Gerard Allwein, the TiPXX-meister, and Ruth Irene.

We are grateful to the reviewers for their observations and helpful comments.

References