Practical Covert Channel Implementation through a Timed Mix-Firewall

Richard E. Newman[1] and Ira S. Moskowitz[2]
[1]Dept. of CISE, PO Box 116120
University of Florida, Gainesville, FL 32611-6120
nemo@cise.ufl.edu
&
[2]Center for High Assurance Computer Systems, Code 5540
Naval Research Laboratory, Washington, DC 20375
ira.moskowitz@nrl.navy.mil

Abstract. The ability to communicate covertly through Mix-firewalls has been discussed in previous work [**?**]. This paper discusses the issues of implementing this theoretical construction in a prototype system. The practical issues involved in implementation are analyzed, and the actual design is given here. Suggested parameters for concatenated codes for the experimental channel are given. 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 Simple Covert Channel Experimental Results

Although we implemented the testbed in an actual LAN, the firing (tick) rate of the Mix-firewall had to be low (at most once per second or so) for technical reasons. This limited the rate at which we could send data through the covert channel, and consequently, the speed with which we could collect data on variations of the system (e.g., error correcting code strategies, copy code parameter, etc.).

In order to collect more data faster, a single-host simulator was built that simulated the Alice, the Clueless, and Eve all on one system. Alice and the Clueless were simulated as one process, and Eve was simulated as another process. Since the challenge of Eve synchronizing with the Mix-firewall firing had been solved, the Alice process simply sent the Eve process the number of packets Eve would observe each tick. Eve still had to synchronize with the copy code, decode the logical symbols from the copy code, and decode the results using the error-correcting code.

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 Clueless (N=1 to 15) and probability that the Clueless sent a message during a tick (p=0.1 to 0.9). 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 code, at least when p is near 0.5. Graphs of the results for hard and soft input decoding are shown in Figure **??**. 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 (bits per tick) for $N = 2$ and $p = 0.1$, 0.6 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. These may not be 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 [**?**] 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 rates obtained experimentally are 1/3 to 1/10 the information theoretic predicted theoretical maximum rates (capacity).

The characteristic U (that is, $C(p)$ decreases as p approaches 0.5, where $C(p)$ is the capacity for input probability p) shape of the graphs giving the coding rate as a function of p are comparable to the numerical capacity computed in [**?**], although much lower. A graph for this is shown for N=1 Clueless in Figure **??**. In both [**?**,**?**] numerical evidence and plots attempted to show that this "U" shape is standard. However, in [**?**,**?**] the authors attempted to put this on firm theoretical ground. Unfortunately, there is a gap in in the proof of [**?**,**?**, 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, without proof, the "U" shaped behavior as stated in [**?**,**?**, Cor. 1].

1.1 Estimating the Number of Clueless

For the best results, 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 *a priori* knowledge. When Alice underestimates the number of Clueless, then there is a chance that the code selected is not powerful enough to correct all the errors encountered due to the additional noise. Figure **??** 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=5 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 8 Clueless, then Eve will experience a bit error rate around 12%, after decoding, rendering communication ineffective.

Figure **??** 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 12 Clueless, then Eve will experience a bit error rate around 15%, after decoding, seriously disrupting communication.

Eve, on the other hand, only needs to know the average number of packets that the Clueless send per tick (Np). This she obtains when averaging the number of packets over a large number of ticks. The copy code used by Alice (which should depend upon Np) is determined by Eve's observations when she synchronizes with Alice. 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 decisions and the behavior of the Clueless. We have derived recommendations for copy code and convolutional code redundancies as a function of N so that R uniquely determines C (see Table 1.1).

Table 1. Recommended combinations for Copy Code and Convolutional Code

Number of Clueless N	Soft Decoding			Hard Decoding		
	Copy Code Param. R	Conv. Code Param. C	Product RC	Copy Code Param. R	Conv. C ode Param. C	Product RC
1	3	3	9	3	5	15
2	4	4	16	4	6	24
3	5	5	25	8	5	40
4	6	6	36	7	7	49
5	7	6	42	9	7	63
6	8	7	56	11	7	77
7	9	7	63	13	7	91
8	9	7	63	14	7	98
9	10	8	80	12	9	108
10	10	8	80	12	9	108
11	10	8	80	15	10	150
12	11	8	88	15	10	150

In the absence of *a priori* information on the number of Clueless, it is possible for Eve to estimate the effective number of Clueless from observations (i.e., from the distribution of packets sent per tick). Alice, on the other hand, must either have this information already, or must also monitor

the medium to determine the number of Clueless present. 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. In these experiments, a combination of a copy code and a convolutional code were used to demonstrate successful exercise of the channel. Naturally, Eve must decode the detected signals in a manner consistent with the way in which Alice encoded them. As long as Alice is consistent in the copy code she uses (see below), and sends a sufficient preamble, it is possible for Eve to detect the copy code length, even if this is not known ahead of time. If Alice uses convolutional code parameters that depend on the copy code length, then Eve will also be able to determine these from observation. So in short, it is incumbent upon Alice to select channel coding that suffices to deliver the messages she sends to Eve error-free (with high probability). 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 messages may not be delivered error-free (see Figures **??** and **??**).

1.2 Recommended Code Parameter Combinations

Table 1.1 shows recommended combinations of copy code redundancy and convolutional code reduncancy depending on the number of Clueless present. Eve is able to discern the copy code rate by autorcorrelation of the received raw signal, and can synchronize by observing windowed deviations with windows of the copy code redundancy size. By pairing a unique convolutional code rate with each copy code rate, Eve can then know which convolutional code to use to decode the received symbols, depending on whether she and Alice have agreed to use hard coding or soft coding (this is a computational power issue). Table 1.1 was obtained by analyzing our output and searching for code combinations that gave little, or no, error.

2 Conclusions

This work has provided proof-of-concept for the model covert channel first proposed in [**?**]. The channel exists, and can be exercised, though the practical error-free rate is much lower than the information theoretic rate derived earlier.

While the restriction that each host send at most one packet per tick is certainly artificial, we suspect that this limitation is more of a burden to Alice and Eve than to the Clueless (that is, it is harder for Alice and Eve to exercise the covert channel successfully than it would be if Alice could send more messages per tick). If at most k messages can be sent by each host per tick, then this should have a similar effect as 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 convolution codes are effective at higher rates than convolutional codes alone, so this begs 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 at 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.