WYNER'S ANALOG ENCRYPTION SCHEME: RESULTS OF A SIMULATION

Burt S. Kaliski

MIT Laboratory for Computer Science
545 Technology Square
Cambridge, Massachusetts 02139

ABSTRACT

This paper presents the results of a simulation of an analog encryption scheme. The scheme, introduced in 1979 by Aaron Wyner of Bell Telephone Laboratories, provides secure, accurate scrambling of speech waveforms, while conforming to the bandlimitedness of a telephone channel. The simulation confirms the scheme's theoretical properties, based on numerical measures and on listening to encrypted and decrypted waveforms.

INTRODUCTION

Security in communications is increasingly important. While the thrust of research is in digital encryption methods, analog encryption is also of growing interest. Such encryption is useful, for example, in transmission to mobile telephones in automobiles, and to cellular radios.

Analog encryption for telephone speech poses special problems. Unlike digital encryption, it must be performed in real time. The result of encryption must conform to the bandwidth of the telephone channel, so that no information is lost. And it must be secure.

A scrambling scheme introduced by Aaron Wyner of Bell Telephone

G.R. Blakley and D. Chaum (Eds.): Advances in Cryptology - CRYPTO '84, LNCS 196, pp. 83-94, 1985.

Laboratories [Wyner, 1979a and 1979b] answers a theoretical question: Are bandlimitedness and security mutually exclusive? The answer is a theoretical "yes"; we present here a practical "yes," as confirmed by a software simulation of the scheme.

Let us now define a scrambler, since it is a term we will use frequently. One may view a scrambler as a black box whose input is an element of one space, and whose output is an element of another. The black box computes a one-to-one mapping from the input space to the output space, based on some secret quantity, or key.

Associated with the scrambler is a descrambler, a black box that computes a one-to-one mapping from the output space to the input space. The mapping is the inverse of that of the scrambler, when the secret quantity or key is the same. Without the key, however, it is computationally difficult to compute the inverse.

The input and output spaces may be bit-strings of a given length, or a given number of samples of an analog signal. Although the samples of an analog signal can be considered as bit-strings (for instance, by concatenating the bits representing the samples), we choose to differentiate between analog and digital scramblers, based on the interpretation of the inputs and outputs.

An practical scrambler, digital or analog, must have two characteristics: security and speed. An analog scrambler must also be accurate. With regard to the model here, such a scrambler must conform to the bandlimitedness and noise properties of a telephone channel. Speed also is critical, because an analog scrambler for telephone communication must operate in real time.

RESULTS OF THE SIMULATION

The simulation of Wyner's speech scrambler gives good results. Accurate decryption and secure encryption are shown to be achievable with reasonable processor power. In particular, a signal-to-noise ratio of about 13 dB is possible with a processor that can perform a multiply-and-add in 4 us.

Several parameters define the operation of the scrambler. These parameters are adjusted to conform to the characteristics of the telephone channel, the quality of output desired, and the available processor power for the scrambler. A set of parameters that provides high-quality output and requires reasonable encryption speed for a typical telephone channel is the following:

- o <u>block size</u> -- 32 points
- o <u>sampling rate</u> -- 8000 per second
- o <u>input band</u> -- [0, 2700]
- o <u>output band</u> -- [300, 3200]

The significance of each of these parameters is described below.

The main test of the scrambler is the encryption and decryption of an utterance of the word "potato." A qualitative, and somewhat subjective, analysis of the scrambler provides encouraging results.

The decrypted signal sounds clear and accurate, with little difference in quality from the original signals. The transmitted signal, as expected, resembles white noise. One is able, however, to identify vowels when hearing the waveform. The signal sounds like "zhuh-zhuh-zhuh," one "zhuh" for each vowel. This problem, a result of orthogonality in the scrambler, is discussed below.

Figures 1 and 2 show the waveforms and spectrograms for the transmitted and decrypted signals. The figures are hardcopies of a Lisp machine display using a speech analysis system called Spire, which was developed at the MIT Speech Laboratory.

The transmitted signal's spectrogram is distributed nearly uniformly across the desired band, but not across time. Note that its short-time energy resembles that of the decrypted signal. The decrypted signal's spectrogram looks very much like speech, with the vowels (O and A) clearly visible.

The results, although favorable, are not complete. Certain simplifications in the software model lead to results perhaps better than those attainable in practice. Nonetheless, high quality with reasonable processor power is still possible. The simplifications are explained below.

CHOICE OF PARAMETERS

Four parameters determine the operation of the scrambler. Their values determine the running time of the scrambling algorithm, the accuracy of transmission, and the level of security achieved.

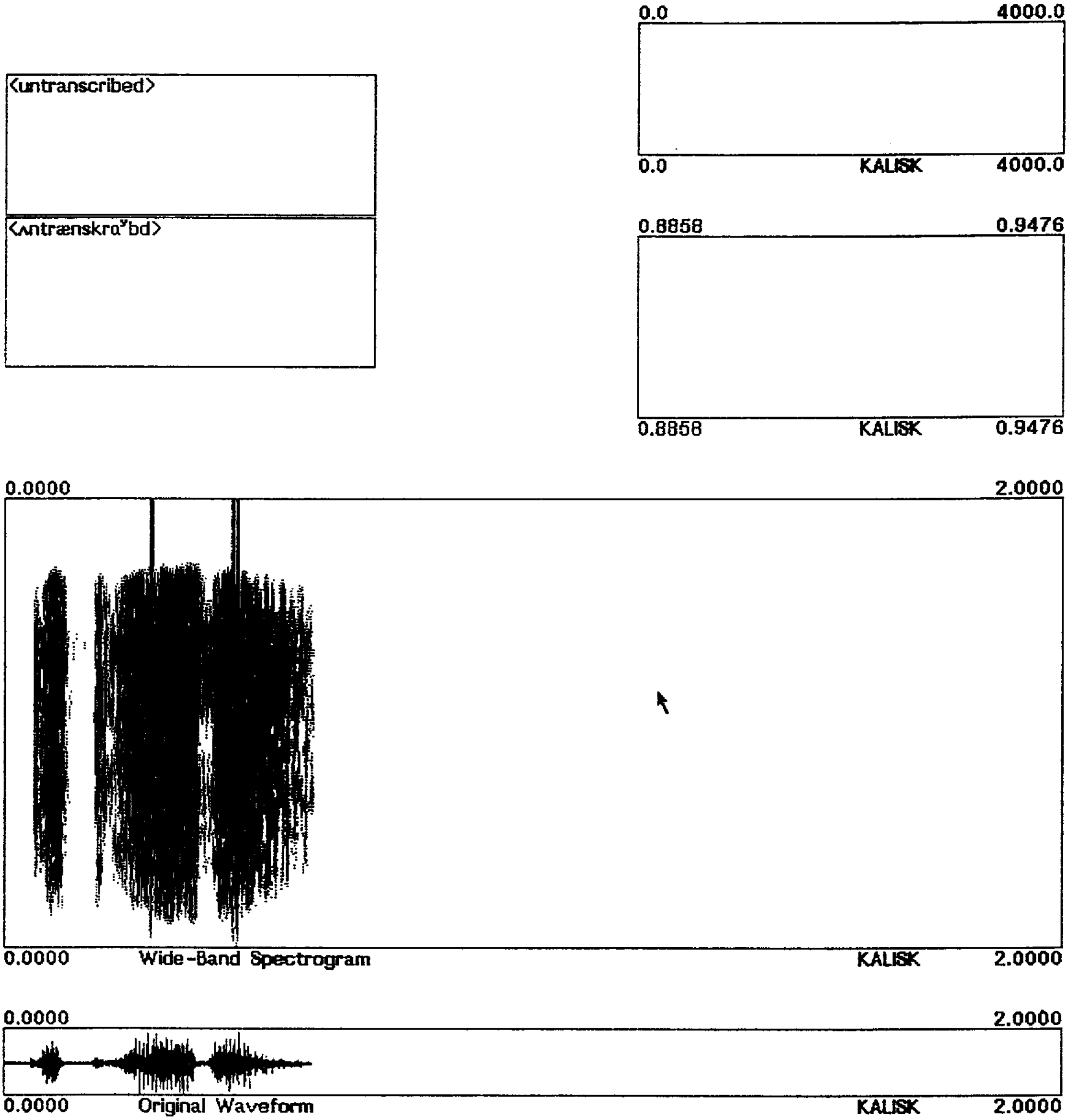

Figure 1 -- Spectrogram of trasmitted waveform

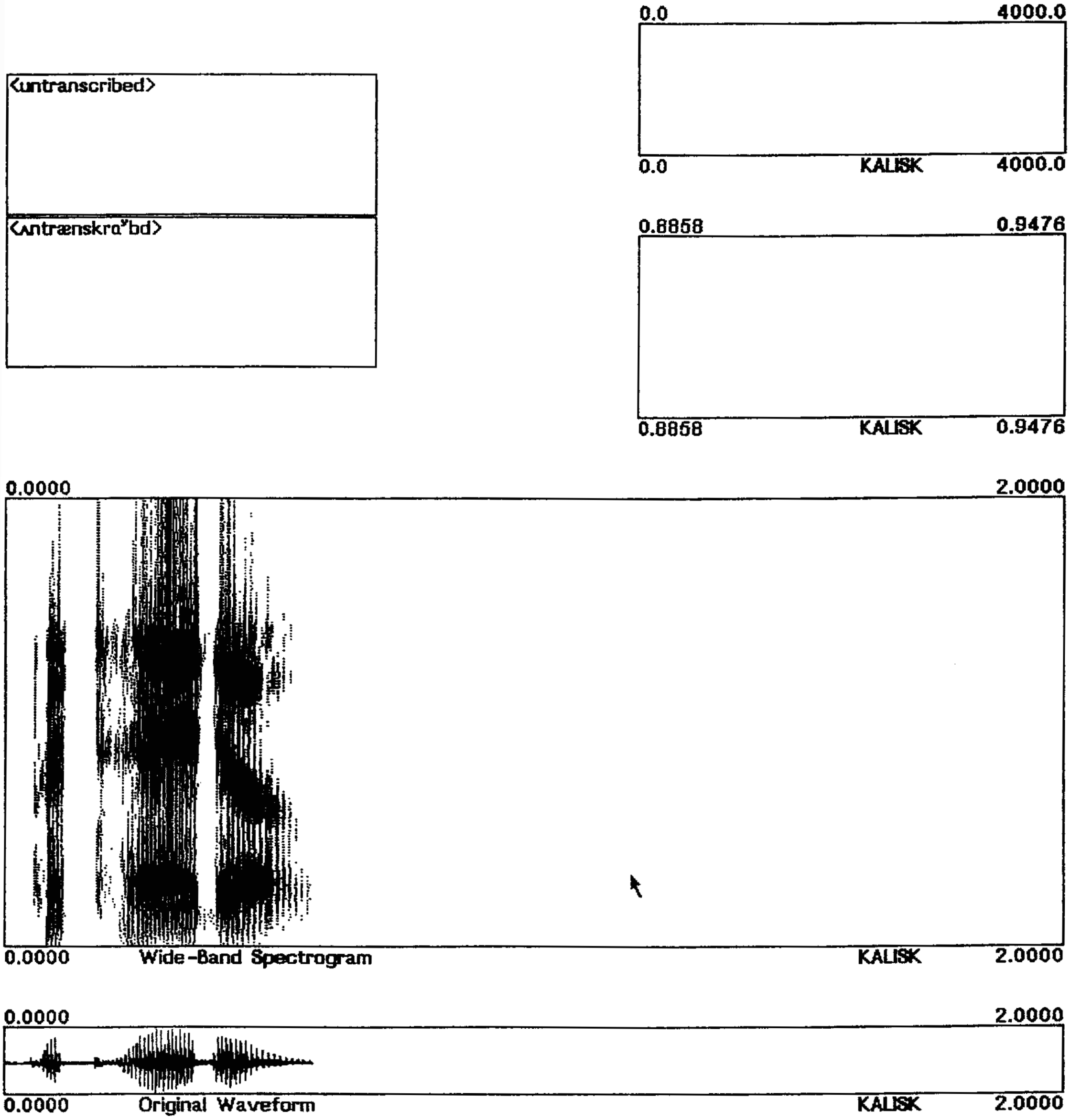

Figure 2 -- Spectrogram of decrypted waveform

- o <u>block size</u> -- The scrambler operates on blocks of samples, producing N outputs for every N inputs. The block size determines the security, and also the speed requirements. Typical values might be 32, 64, or 96. We choose 32 to minimize speed requirements.
- o <u>sampling rate</u> -- A new sample of the input waveform is taken every T seconds. For speech, a rate of 8 kHz (T = 125 us) is about the minimum to avoid aliasing. Higher rates are generally not required, but they may improve the quality of the decrypted signals.
- o <u>input band</u> -- This is the frequency band used to determine the basis vectors for scrambler input (and hence descrambler output). It is generally selected to maximize the amount of input energy included. For small N, much energy appears as DC, so a low end of 0 is chosen. The high end is at 2700 to provide a bandwidth narrower than that of the output.
- o <u>output band</u> -- This is the band used to determine basis vectors for scrambler output and descrambler input. It is generally wider than the input band, and conforms to the characteristics of the telephone channel. We use [300, 3200], corresponding to the simulator's channel model.

Design of most of the scrambler depends not on the actual frequencies--such as [300, 3000]--but on their discrete equivalents, those scaled by the sampling period. With the parameters above, the equivalent band would be [.0375, .375]. In the discussion that follows, both input and output bands are loosely referred to as [W1, W2], where W1 and W2 are in the range [0, .5]. The quantity W represents the bandwidth, or W2 - W1.

HOW THE SCRAMBLER WORKS

The bandlimitedness of the telephone channel constrains the output space of the black box scrambler model. Only those outputs resembling speech in bandwidth may be produced. Similarly, only those inputs of such form should be expected. Thus a mapping between the speech-like subspaces of the sets of all samples must be provided.

We can approximate those subspaces by saying they correspond to the band [W1, W2] within the frequency spectrum on [0, .5]. Thus, while the spaces obtained from N independent samples may have dimension N, the subspaces--constrained to be near zero outside the selected band--really have dimension 2WN. With the parameters above, this quantity is about 22 for the input band, and 23 for the output band.

The definition of bandlimited also includes indexlimited in the context of a speech scrambler. Each block of samples, if considered alone with samples in all other blocks set to zero, should be bandlimited for sufficiently large N. This means the discrete-time Fourier transform (that is, the transform from an infinite set of discrete values to an infinite, continuous waveform) of the block considered alone, should be roughly limited to [W1, W2].

Assuming that the subspace we want is of dimension 2WN, we must find a way to describe it. A naive description involves the discrete Fourier transform (that is, the transform from a set of discrete values to a set of discrete values). Perhaps the subspace is all sets of samples whose DFT is zero, except at those 2WN points in the desired band. Since the transform is one-to-one, the subspace would be the correct size.

The solution is not quite so simple. We seek sequences that are both indexlimited and bandlimited. Those whose DFT is zero, except at certain points, include sine waves. Certainly these are not indexlimited.

Mathematical physics has a set of functions called prolate spheroidal waveforms. These waveforms are the only eigenfunctions of the finite Fourier transform (that is, the transform from a continuous, infinte waveform to a continuous waveform over [0, 1)). Recall that eigenfunctions are those that, when transformed, are but scaled over a certain range--in this case, [0, 1).

The discrete counterparts of the waveforms--discrete prolate spheroidal sequences--are similarly related to the discrete-time Fourier transform. Here the relationship is somewhat different. The sequences are "eigensequences" not of the transform itself, but of bandlimiting and of indexlimiting.

Specifically, when a sequence, defined by parameters N, W1 and W2, is bandlimited to [W1, W2], it is only scaled in indices [1, N]. The scaling is by the eigenvalue corresponding to the eigenvector.

Recalling digital signal processing, it is seen that to bandlimit a sequence is the same as to convolve the sequence with the non-causual impulse response of an ideal filter. For applying the response to a finite set of points, this is equivalent to multiplication by a matrix derived from the impulse response. The eigenvectors of such a matrix are the eigensequences above.

The key result is that the eigenvalues fall into three categories: those close to 1, those close to 0, and others between 1 and 0. The eigenvalue represents the amount energy in the desired band [W1, W2]. The majority of values are close to 1 or 0; 2WN such values are close to 1. The corresponding 2WN eigenvectors are the basis of the subspace.

An example of prolate spheroidal sequences is given in Figure 3. The sequences, their discrete-time Fourier transforms and their eigenvalues are shown, with N = 8 and band [.0375, .375]. Notice the high concentration of the first four sequences, and the low concentration of the last two.

It remains to be shown how to use these discrete prolate spheroidal sequences to scramble speech. Since 2WN sequences form an approximate basis for the desired subspace, we can compute for some set of N samples a weight corresponding to each sequence. The weights are easily found by dot-products.

We know that any combination of the 2WN sequences will remain bandlimited. Hence the weights computed may be scrambled in any fashion, and a new set of samples produced. Scrambling the weights by an orthogonal transformation is most accurate, because any error in computing the weights is not increased.

The security comes from the orthogonal transformation. Multiplying the weights, represented as a 2WN-element vector, by a 2WN x 2WN matrix, is equivalent. The matrix may be constructed randomly, based on a sequence of random numbers initiated by a secret key. The receiver, knowing the key, generates the same sequence of random matrices, and hence can reverse the orthogonal transformations.

One unfortunate result of the orthogonality is that the short-time energy of the output of the scrambler is the same as that of the input. This allows an adversary to differentiate between vowels and consonants. This problem is easily solved by adding a dummy waveform to the output to maintain a constant energy.

Considering the scrambling scheme as a series of matrix multiplications, we can estimate a running time. The transformation to prolate spheroidal weights is a 2WN x N matrix by N x 1 vector product. The 2WN x 2WN matrix by 2WN x 1 vector product follows, then reconstruction using a N x 2WN matrix by 2WN x 1 vector product.

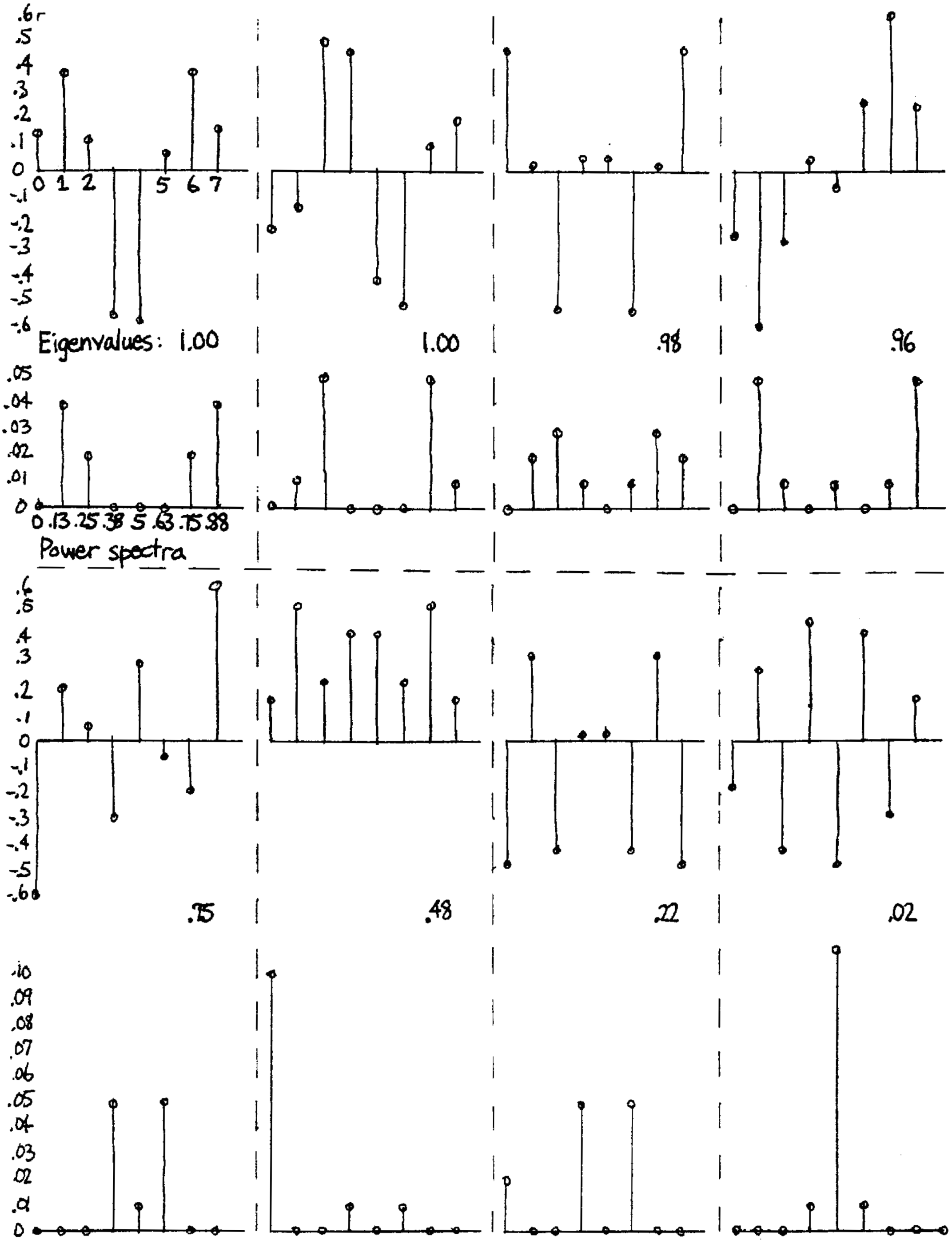

Figure 3 -- Prolate spheroidal sequences for N = 8

The serier of multiplications is equivalent to an N x N matrix by N x 1 vector product. This requires N2 multiply-and-add operations for every N samples. Hence one such operation is required each T / N seconds, giving the 4 us timing above.

The simulator, of course, is not so constrained. It operates about 4000 times more slowly than a real-time scrambler. To scramble "potato" took several hours.

SIMULATING A MODEL OF THE SCRAMBLER

Notice the description of the scrambler above said very little about its role in a complete transmission system. Here we describe the way in which the black box is connected to the world--the input, transmission, and output processing.

Figure 4 contains a block diagram of a model given by Wyner and used in the simulation. The diagram has eight components. The scrambler and unscrambler are those described above. The channel is a typical telephone transmission channel; the equalizer is typical tapped delay-line used to compensate for linear distortion in the channel.

```
   input    +---------+    +-----------+    +-----------+
 -------->| SAMPLER |-->| SCRAMBLER |-->| DESAMPLER |
 signal     +---------+    +-----------+    +-----------+
                                                   |
                                                   V
                                              transmitted
                                                 signal
                                                   |
                                                   V
                                              +---------+
                                              | CHANNEL |
                                              +---------+
                                                   |
                                                   V
                                              +---------+
                                              | SAMPLER |
                                              +---------+
                                                   |
                                                   V
output  +-----------+   +-------------+    +-----------+
<-------| DESAMPLER |<--| UNSCRAMBLER |<--| EQUALIZER |
signal  +-----------+   +-------------+    +-----------+
```

Figure 4 -- Block diagram of model for scrambler

The model of the scrambler naturally lends itself to object-oriented programming. Each module may be represented by a unique data type; indeed, even the signals and prolate spheroidal eigenvectors may be data types. Such an implementation is easily accomplished.

A Lisp Machine--a minicomputer designed specifically for programming in Lisp--is used for the simulation. Its system of "flavors" allows construction of the data types to make a block diagram, and its graphics capabilities are useful for viewing waveforms and sequences as they pass through the block diagram.

While the Lisp Machine is a sequential processor, the "transmission" of signals between modules is like that of a parallel, multiprocessor data-flow network. A driver passes input signals to the first module, which transmits to other modules, and so on, until the signals propogate to modules with no successors.

It should be noted that the test environment is incomplete. The simulated equalizer is built knowing the response of the telephone channel. In practice, the equalizer would not be able to compensate as well for linear distortion, and errors would be larger. In addition, most calculations in the simulator are performed with high-precision floating point numbers. In practice, fewer bits may be used, and quantization errors in sampling will be present. These factors are not included in the simulation.

WHERE CRYPTOGRAPHY FITS IN

It may appear that the scrambler is based more on signal processing than on cryptography. The selection of scrambling matrices, however, is a problem of current interest in cryptography. A good overview of present methods for creating scrambling matrices is found in [Sloane, 1983].

Two general methods for selecting the random matrices come to mind. Precomputation with random selection from a set of such matrices is fast; computation at run-time is secure. In either case, some sequence of random numbers is needed. The "key" to the black box determines this sequence.

The simulator uses precomputation, since the matrix selected has little effect on the accuracy of the system. A set of matrices is computed by the Gram-Schmidt algorithm--by orthogonalizing matrices containing normally-distributed random elements.

Hadamard matrices (that is, those of size m x m, orthogonal, with all elements either 1 / m or -1 / m), are best suited for run-time calculation. Random permutations and sign changes, all quickly done, further increase the security.

CONCLUSION

A faster implementation of the scrambler is necessary for further study, given the slowness of the software model. The slowness is due primarily to the object-oriented implementation, which is ideal for detailed, step-by-step testing, but impractical for system testing. Since construction of a hardware unit in a short time period would be difficult, use of a processor with a floating-point accelerator is a logical next step.

ACKNOWLEDGMENTS

The author is grateful to MIT Professor Ronald L. Rivest for advising the undergraduate thesis that led to this paper, and to Aaron Wyner for helpful comments on the thesis. I also wish to thank my fiancee, Cathy Oehl, for her love and support. Most importantly, for the blessing of an MIT education and for the opportunities it has made possible, I thank the Lord Jesus, to Whom my career and life are dedicated.

REFERENCES

Wyner, A.D. (1979a) "An analog encryption scheme which does not expand bandwidth -- Part I: Discrete time," IEEE Trans. Inform. Theory, 25.

Wyner, A.D. (1979b) "An analog encryption scheme which does not expand bandwidth -- Part II: Continuous time," ibid.

Sloane, N.J.A. (1983) "Encrypting by random rotations," in "Cryptography," Lecture Notes in Computer Science, 149, Springer-Verlag, Berlin.