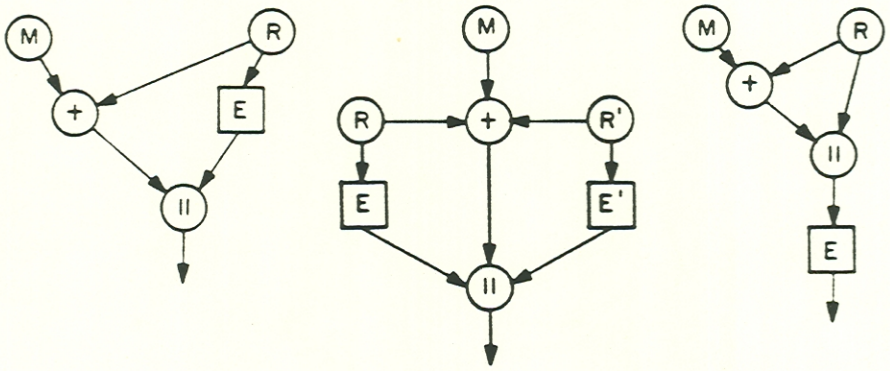
MIT/LCS/TM-234

RANDOMIZED ENCRYPTION TECHNIQUES



Ronald L. Rivest and Alan T. Sherman

January 1983

Randomized Encryption Techniques

Ronald L. Rivest and Alan T. Sherman

January 1983

This paper was presented on August 23, 1982, at the CRYPTO 82 conference at the University of California, Santa Barbara, and will appear in *Advances in Cryptography: Proceedings of CRYPTO 82*, Plenum Press, (New York, 1983).

The research was supported by NSF grant MCS–8006938.

Keywords and Phrases: Cryptographic security, cryptography, cryptology, modes of operation, randomization, randomized encryption.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LABORATORY FOR COMPUTER SCIENCE

Cambridge Massachusetts 02139

Randomized Encryption Techniques[1]

Ronald L. Rivest and Alan T. Sherman
MIT Laboratory for Computer Science
Cambridge, MA 02139

Abstract

A *randomized encryption procedure* enciphers a message by randomly choosing a ciphertext from a set of ciphertexts corresponding to the message under the current encryption key. At the cost of increasing the required bandwidth, such procedures may achieve greater cryptographic security than their deterministic counterparts by increasing the apparent size of the message space, eliminating the threat of chosen plaintext attacks, and improving the *a priori* statistics for the inputs to the encryption algorithms. In this paper we explore various ways of using randomization in encryption.

I. Introduction

Cryptographers often enhance the security of their codes and ciphers by using randomization in the encryption process.[2] For instance, it is common practice to define codes by associating with each plaintext word a set of codewords; to encode a plaintext word, the encoding procedure randomly selects a codeword from the corresponding set. An elegant example of this idea is the Jefferson-Bazeries Wheel Cipher and its variations, ciphers used by the U.S. during both world wars [Kru81], [Kah67]. Another common randomized encryption technique is to intersperse null

[1]This research was supported by NSF grant MCS-8006938.

[2]We assume the reader is familiar with the general principles of cryptology — as presented in [DiH79], for example.

plaintext words in the input stream or to intersperse null codewords in the output stream of an encryption algorithm.

New randomized encryption techniques have been recently published by Wyner [Wyn75], McEliece [McE78], Lempel [Lem79], Sloane [Slo82], Nicolai [Nic82], and Goldwasser and Micali [GoM81,82]. Randomization is also proposed to achieve secure digital signatures in Rabin's variation of the RSA cipher [Rab79], to enlarge the apparent size of the message space in the encryption of passwords under the *Unix* operating system, and to create the *join* of two cryptosystems — an encryption scheme as strong as the stronger of two component encryption schemes [AsB82]. This paper is strongly motivated by these examples.

The goal of randomized encryption is increased security, and this goal may be achieved through each of the following means:

- Smoothing out the *a priori* statistics for the distribution of inputs to the encryption algorithm or one-way function. This may eliminate or reduce the effectiveness of statistical attacks.
- Eliminating the possibility of chosen plaintext attacks. If the encryption function encrypts only randomly generated bit sequences, a chosen plaintext attack is impossible to mount.
- Increasing the apparent size of the message space to the enemy. If the message space is small, a non-randomized scheme runs the risk of being defeated by simple statistical or forward search [Sim82] attacks.

In information theoretic terms, randomized encryption attempts to obtain higher levels of security through increasing the entropy of the plaintext [Sha49], [Gal68]. In this respect, randomized encryption is similar to source coding. However, while source coding reduces redundancy through message compression, randomized encryption increases entropy by adding random bits. Although randomized encryption increases the bandwidth, it is a relatively simple technique that can be easily applied in many applications.

In this paper we explore how randomization can be used during encryption. In section II, we explain what randomized encryption is. Next, in section III, we categorize many ways randomization can be used. In section IV, we discuss extensions and applications of randomized encryption, and, in section V, we discuss directions for further research. We summarize our results in section VI.

II. Randomized encryption

A *randomized encryption procedure* produces ciphertext via a nondeterministic function of the message to be encrypted and the encryption key. For each key, a given message may be encrypted in several ways; the encryption procedure makes

a random choice to decide which way to use. We assume, though, that encryption is uniquely decodable: for each key, each ciphertext corresponds to at most one message.

More precisely, a randomized encryption procedure is a relation $\Pi \subseteq \mathcal{M} \times \mathcal{K} \times \mathcal{C}$, where $\mathcal{M}$ is the message space, $\mathcal{K}$ is the key space, and $\mathcal{C}$ is the ciphertext space, such that for each key $k \in \mathcal{K}$ and each ciphertext $c \in \mathcal{C}$, there is at most one $x \in \mathcal{M}$ such that $(x, k, c) \in \Pi$. Also, for each message $x \in \mathcal{M}$ and each key $k \in \mathcal{K}$ there is at least one ciphertext $c \in \mathcal{C}$ such that $(x, k, c) \in \Pi$. The quadruple $(\mathcal{M}, \mathcal{K}, \mathcal{C}, \Pi)$ is called the *randomized encryption system.*

In practice, a randomized encryption procedure would encrypt a message x under a key k as follows. First, a bit sequence r would be randomly chosen from a set $\mathcal{R}$ of bit sequences. Second, the value $\psi(r, x, k)$ would be computed, where ψ is a deterministic function such that $\psi: \mathcal{R} \times \mathcal{M} \times \mathcal{K} \mapsto \mathcal{C}$ and $\Pi = \bigcup_{r \in \mathcal{R}, k \in \mathcal{K}, x \in \mathcal{M}} (x, k, \psi(r, x, k))$.

Randomized encryption schemes require a source of random bits. Such a source may be built using neon discharge tubes, noisy diodes [Mad72], radioactive decay [Kle60], or other natural sources of randomness [Gif82]. We assume such generators are capable of producing unbiased and totally unpredictable bit sequences as rapidly as needed. Pseudo-random bit sequence generators, such as the one proposed by Blum and Micali [BlM82], are not suitable replacements for random bit sources here.

Since, for any key, each message corresponds to several ciphertexts and each ciphertext corresponds to at most one message, the ciphertext space will be larger than the message space. This requires a certain amount of *bandwidth expansion* in the communication channel since more bits must be transmitted to specify the ciphertext than are needed to identify the message. Bandwidth expansion is the major cost of using randomized encryption, and it is unavoidable. As a measure of this cost, we define the *bandwidth expansion factor* of an encryption procedure to be the ratio of the the number of ciphertext bits transmitted to the corresponding number of message bits. If the bandwidth expansion factor is not constant between keys or even between messages, a more general notion of an *average bandwidth expansion factor* could be used instead. Most of the schemes reviewed in this paper have uniformly constant bandwidth expansion.

Figure 1 shows the block diagram for the secure transmission of data over an insecure communications line using a randomized encryption procedure. Note that the physical random bit sequence generator is within the secure area of the encryption unit. We assume an enemy cannot determine the intermediate results of the encryption and decryption computations. In particular, the output of the physical random bit sequence generator is hidden from the enemy, unless it is part of the ciphertext.

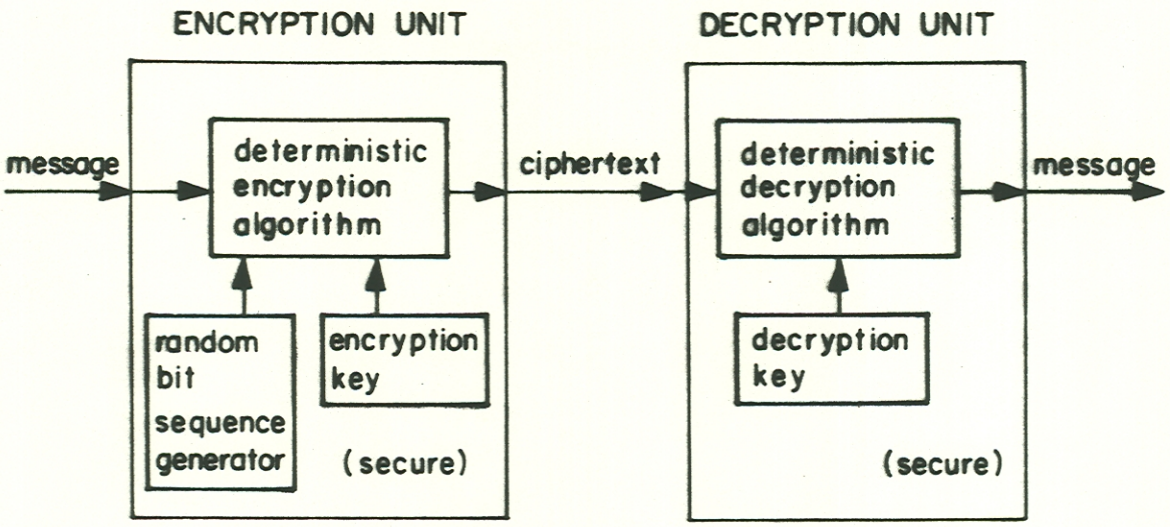


Figure 1. Block diagram of a randomized encryption procedure

III. Classification of randomized encryption methods

In this section, we will present several ways in which randomization can be used in encryption. We organize the schemes in two categories: randomized block ciphers and randomized stream ciphers.[3]

III.A. Notation

We use the following notation:

M, C denote the message space and ciphertext space of a randomized encryption algorithm.

K denotes a set of keys.

$\mathcal{R}$ denotes a set of bit sequences.

M denotes a message source that generates messages in M.

R, R' denote sources of random bits that generate sequences in $\mathcal{R}$.

[3] A *stream* or *finite state* cipher is based on an encryption function that can be in one of finitely many different states. First, the state is set to some initial value; then, after each application of the cipher, the state is updated by some *state transition function*. The encryption process depends on the message, the key, and the current state. A *block* or *memoryless* cipher is a cipher that does not depend on any state information.

k, k' denote cryptographic keys in K.

$|M|$ and $|R|$ denote the lengths of messages and bit sequences generated by M and R respectively.

E, E' denote deterministic encryption functions mapping $K \times Domain(E)$ into $Target(E)$, where $Domain(E)$ is the message space of E and $Target(E)$ is the ciphertext space of E. For many examples it will be true that $M = Domain(E) = Target(E) = \{0,1\}^n$, for some n. When E is subscripted, the subscript refers to the key in use: thus, for each key $k \in K$, E_k denotes an injective map from $Domain(E)$ into $Target(E)$.

D, D' denote decryption functions corresponding to E, E'. For any message x, and any key k, it must be true that $D_k(E_k(x)) = x$.

F, F' denote key-dependent deterministic one-way functions mapping $K \times Domain(F)$ into $Target(F)$. Note that F_k need not be injective for any key k.

P, P' denote pseudo-random bit stream generators.

We will use the descriptive but slightly abusive notation $E_k(R)$ to denote an encryption scheme in which the sequences generated by the random source R are encrypted with encryption function E under key k. When the meaning is clear, we will also abbreviate $E_k(R)$ by E_kR or ER. We will also use similar notations involving M, D, and F.

$\|$ denotes concatenation. For example, $a \| b \| c$ denotes the concatenation of the strings a, b, c.

$\oplus$ denotes exclusive or (XOR).

For example, $EM \| R$ denotes the scheme in which the final ciphertext is obtained by concatenating (1) the ciphertext obtained by enciphering M with encryption function E, and (2) a randomly generated block of bits.[4] Although this scheme increases the required bandwidth for no apparent benefit, it can serve as a useful reference when considering other schemes.

When evaluating a randomized encryption scheme it is useful to distinguish between intrinsic weaknesses of the scheme and weaknesses due to bad component cryptographic functions. An *intrinsic* weakness occurs regardless of what component functions are used.

[4] We adopt the convention that function application takes precedence over concatenation. Thus, $(EM \| R) = ((EM) \| R)$.

III.B. Randomized block ciphers

In this section, we review a number of ways of using randomization in block ciphers. We assume that the sizes of the message and ciphertext blocks are fixed parts of each encryption system.

III.B.1. $E(M \parallel R)$

Perhaps the simplest randomized encryption technique is to concatenate a random sequence[5] to the message and then encrypt the result. The decryption unit applies the decryption function and then discards the random sequence. We call this method the $E(M \parallel R)$, or *random padding* technique. See Figure 2.

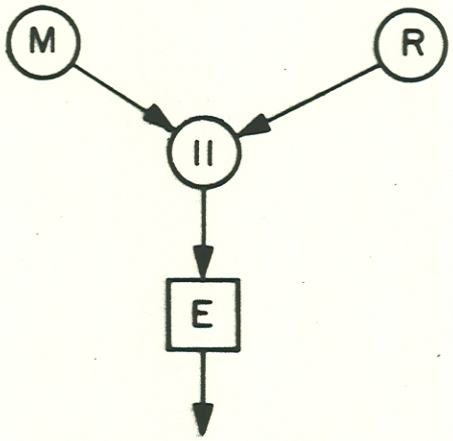


Figure 2. $E(M \parallel R)$

There are five advantages of the $E(M \parallel R)$ scheme. First, this scheme increases the apparent size of the message space to the enemy in the sense that $Domain(E)$ is $\mathcal{M} \times \mathcal{R}$ rather than $\mathcal{M}$. Second, under a chosen plaintext or chosen ciphertext attack, this scheme does not allow the enemy to obtain any complete input to the encryption function. Therefore this scheme partially protects against such attacks. Third, this scheme somewhat improves the distribution of inputs to the encryption function in that part of each input to the encryption function is randomly selected. Fourth, this scheme is not intrinsically vulnerable to *bit twiddling* — an active attack in which the enemy selectively modifies ciphertext bits in order to alter corresponding plaintext bits. Finally, since $\mathcal{R}$ may be chosen to be any size, this scheme allows for variable bandwidth expansion. Of course, if random padding is to do much good, $\mathcal{R}$ must be sufficiently large. If $|M| = |R|$, then the bandwidth expansion factor is two.

At least one provably secure public-key cryptosystem fits this classification. In particular, the Goldwasser/Micali scheme [GoM81,82] can be viewed as an $E(R \parallel M)$ scheme, where $E(x) = g^x \pmod{n}$; n is a large composite number;

[5]For brevity, we use the phrase *random sequence* to mean randomly generated bit sequence.

$|M| = 1$; and g is a generator of Z_n^*, the multiplicative group modulo n. In the Goldwasser/Micali scheme, each message is sent one bit at a time. The security of their scheme is based on the assumption that it is difficult to determine in general whether or not an integer z is a quadratic residue modulo n, where $0 < z < n$ [NiZ80]. To encrypt a 0, the sender randomly selects a quadratic residue modulo n. Similarly, to encrypt a 1, the sender randomly selects a quadratic non-residue modulo n with Jacobi symbol equal to 1. Here n is a large composite number published by the intended recipient of the message. To enable the sender to encrypt, the intended receiver also publishes a quadratic nonresidue y modulo n with Jacobi symbol equal to 1. Since the intended recipient knows the factorization of n, he can determine whether the transmitted numbers are quadratic residues modulo n. Randomization is essential to the security of the system: unless the encryption of each bit is randomly chosen, the enemy could break the system. Unfortunately, the bandwidth expansion factor of the Goldwasser/Micali scheme is several hundred, making it impractical for many applications.

Avis and Tavares [AvT82] present a similar example of a simple cipher that they conjecture is hard to break when used in an $E(R \parallel M)$ mode. For the Avis and Tavares scheme, $E(x) = xe \pmod{n}$, where n is a large composite number and e is relatively prime to n. The security of this scheme remains to be tested.

III.B.2. $(M \oplus R) \parallel ER$

The $(M \oplus R) \parallel ER$ scheme is similar to the Vernam one-time pad in that the message is XORed with a randomly generated bit sequence. But unlike the one-time pad, the random sequence is encrypted and then concatenated to the enciphered message. Asmuth and Blakely refer to this scheme as the cryptographic exponential of the underlying cryptosystem [AsB82]. See Figure 3.

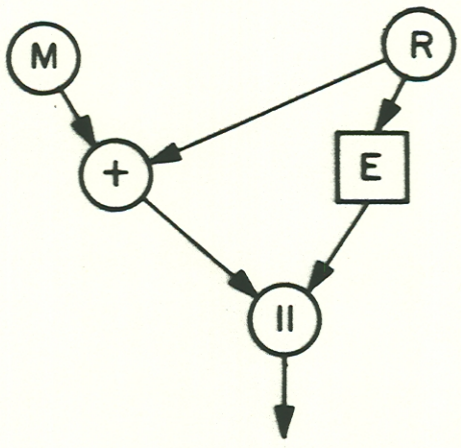


Figure 3. $(M \oplus R) \parallel ER$

The main idea of this scheme is to apply the encryption function only to random bit sequences, guaranteeing uniform *a priori* statistics of the inputs to the encryption function. Whereas the previous method uses the encryption function to mix the random bits with the message bits, this scheme directly randomizes each bit of the message. There are, however, some possible weaknesses with this technique. First, as with the one-time pad, there is the danger with this scheme that an active wiretapper could selectively change certain bits of the plaintext by modifying the corresponding bits of the ciphertext. Second, a chosen ciphertext attack against the system amounts to a chosen ciphertext attack against the encryption function. (See section IV.A. for ways to prevent such attacks.) Third, although the inputs to E are randomly selected, the conditional probability distribution $Prob(R \mid M \oplus R)$ is identical to the *a priori* probability distribution of the message space. Finally, the scheme is vulnerable to a spoofing attack in which an active wiretapper interjects spurious messages into the communications channel: to forge messages, the enemy uses a known plaintext attack to obtain a pair of numbers r, y such that $E(r) = y$. Assuming $|M| = |R|$, this method also has a bandwidth factor of two.

III.B.3 $E_R(M) \parallel E'R$

The $E_R(M) \parallel E'R$ technique encrypts the message with a random message key that is encrypted under the current session key. Note that the $(M \oplus R) \parallel ER$ scheme is also special case of the $E_R(M) \parallel E'R$ technique, where E is the Vernam one-time pad. See Figure 4.

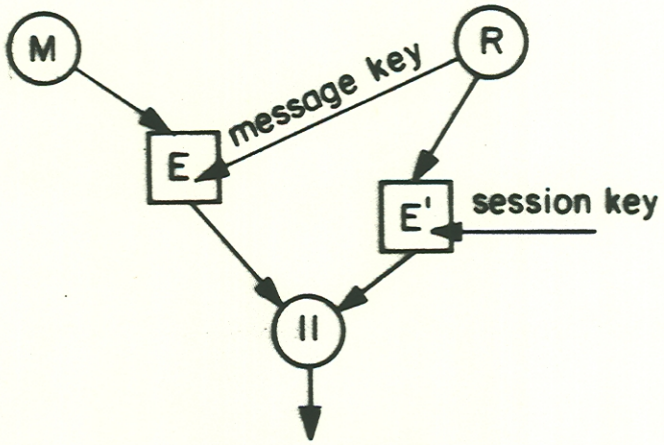


Figure 4. $E_R(M) \parallel E'R$

Provided $|M|$ is much larger than $|R|$, the bandwidth expansion factor can be quite small. The main advantage of this scheme is to reduce the threat of attacks on the the encryption function E that require many ciphertexts produced under the same key.

III.B.4. $E(M \oplus R) \,||\, R$

The $E(M \oplus R) \,||\, R$ technique XORs each message with a random bit sequence and then encrypts the result. The random bit sequence is sent in the clear. See Figure 5.

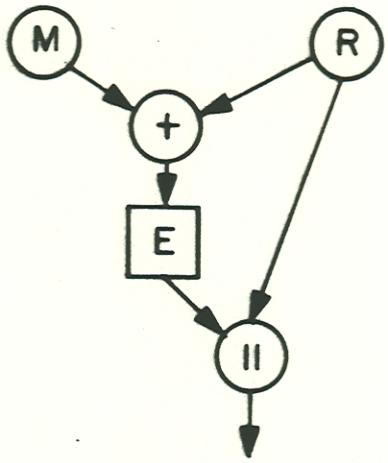


Figure 5. $E(M \oplus R) \,||\, R$

This method has bandwidth expansion factor two and has all of the cryptographic properties mentioned for the $(M \oplus R) \,||\, ER$ scheme.

III.B.5. $E((M \oplus R) \,||\, R)$

The $E((M \oplus R) \,||\, R)$ method combines some of the ideas from the random padding scheme and the $(M \oplus R) \,||\, ER$ scheme. This method XORs each message with a random bit sequence and encrypts the result concatenated with the random sequence. See Figure 6.

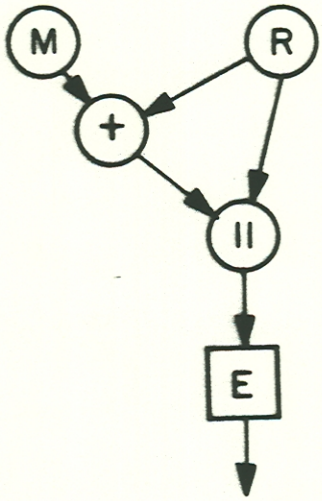


Figure 6. $E((M \oplus R) \,||\, R)$

This scheme increases the apparent size of the message space to the enemy, smoothes out the distribution of inputs to the encryption function, and is not intrinsically vulnerable to bit twiddling. Furthermore, this scheme intrinsically allows neither a chosen plaintext attack nor a chosen ciphertext attack against the encryption function. One potential weakness of this scheme is that under a chosen plaintext attack the enemy can force the encryption function to be applied to bit sequences of the form $r \| r$.

III.B.6. $(M \oplus FR) \| R$

Finally, we consider the $(M \oplus FR) \| R$ method. This method is similar to the $(M \oplus R) \| ER$ technique, except that the random sequence is transmitted in the clear and XORed into each message in encrypted form. Unlike the $(M \oplus R) \| ER$ technique, this method does not require the cryptographic function to be invertible. Asmuth and Blakely call this scheme the cryptographic exponential of a key dependent one-way function [AsB82]. See Figure 7.

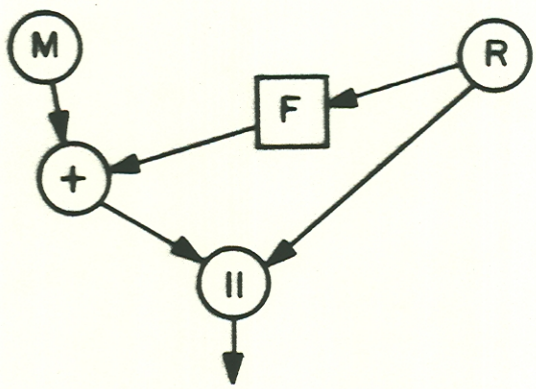


Figure 7. $(M \oplus FR) \| R$

For this scheme, a chosen plaintext attack amounts to a known plaintext attack against the cryptographic function F and a chosen ciphertext attack amounts to a chosen plaintext attack against F. Note that the way in which the $(M \oplus FR) \| R$ method applies F and combines M, FR, and R is similar to the operations performed in each round of DES [FIP77]. DES, however, does not involve randomized encryption.

We will now show that to every $(M \oplus R) \| ER$ scheme there is an equivalent $(M \oplus FR) \| R$ scheme, in the following sense. We say that two randomized encryption systems are *equivalent* if (1) they have the same message space (including probability distribution), keyspace, and ciphertext space, and (2) for each key and message, the two randomized encryption procedures produce the same set of ciphertexts with the same probabilities. Although equivalent randomized encryption systems may not produce the same ciphertext for a given randomly generated bit

sequence, they will yield the same set of ciphertexts with exactly the same probability distribution. Thus, equivalent randomized encryption systems are cryptographically identical in that the enemy cannot distinguish between them.

Proposition 1

Let $\mathcal{E}$ be any $(M \oplus R) \,||\, ER$ randomized encryption system such that $\mathcal{M} = \mathcal{R}$ and such that, for each key k, the encryption function E_k permutes $\mathcal{M}$. Then, there is a $(M \oplus FR) \,||\, R$ randomized encryption system equivalent to $\mathcal{E}$.

Proof.

Since, for each key k, E_k permutes $\mathcal{M}$, it follows that $(M \oplus R) \,||\, E_k R$ and $(M \oplus D_k R) \,||\, R$ are equivalent randomized encryption systems. Hence, by choosing $F = D$, we are done. Q.E.D.

III.B.7. Schemes based on block error-correcting codes

In this section, we consider randomized block ciphers based on error-correcting codes. Here, G denotes an error-correcting code that is capable of correcting at most t errors, and ϵ denotes a source of random bit sequences with at most t ones. The bandwidth expansion factors of these schemes depend on what particular error-correcting codes are used.

III.B.7.a. $GM \oplus \epsilon$

In the $GM \oplus \epsilon$ scheme, the message is first encoded by a secret error-correcting code. Then, a randomly selected subset of the bits in the encoded message is changed. The number of bits changed is at most the error-correcting capacity of the code.

This scheme is based on a public-key cryptosystem proposed by McEliece [McE78]. In McEliece's system, the sender first encodes a message by using a public generator matrix for an error-correcting code. The sender then changes a randomly selected subset of the the bits of the codeword. Provided the sender changes no more bits than the error-correcting code is capable of correcting, the receiver will be be able to decode the message. To decode, the receiver uses a secret trapdoor error-correcting code related to the public generator matrix. The security of the system is based in part on the NP-completeness of the general decoding problem for linear error-correcting codes [BMT78]. The security also depends on the sender randomly selecting what bits to change.

III.B.7.b. $E(GM \oplus \epsilon)$

As with random padding, in the $E(GM \oplus \epsilon)$ method, random bits are combined with the message and then the resulting block is encrypted. In this scheme random

bits are combined with the message by first encoding the message with an error-correcting code and then altering a randomly selected subset of bits in the resulting codeword. The difference between random padding and the $E(GM \oplus \epsilon)$ scheme is in the way in which the random bits are combined with the message. In this method, G need not be secret.

III.B.8. Wire-tap channel schemes

Randomized encryption appears to be especially powerful for situations in which the enemy must listen to a noisy communications channel known as the *wire-tap channel.* As an example, we will review a simple scheme described by Sloane [Slo82]. This example illustrates some of the important ideas in Wyner's research on the wire-tap channel [Wyn75].

Sloane [Slo82] describes a secure way to send one bit over an insecure channel under the assumption that the enemy is forced to listen to a noisy line. To encrypt a bit, the sender randomly selects a bit sequence whose parity is equal to the message bit. The sender chooses the bit sequence long enough so that, due to the noise in the wiretap channel, the enemy will be unable to determine the parity of the codeword.

III.C. Randomized stream ciphers

Since the message of a stream cipher may be viewed as a stream of characters or bits to be encrypted one at a time, stream ciphers provide a convenient context in which to describe encryption techniques that intersperse nulls into the plaintext or ciphertext streams. Furthermore, because the encryption process of a stream cipher depends on the current state of the cipher, randomization can be applied not only to the message, but also to the state. We will describe three randomized techniques for stream ciphers: first, we will discuss two methods for interspersing nulls into the input and output streams; and, finally, we will propose a technique for constructing a randomized stream cipher that uses a nondeterministic state transition function. For another interesting randomized stream cipher based on a nondeterministic state transition function, see Carl Nicolai's paper entitled "Non Deterministic Cryptography" [Nic82].

Many stream ciphers use a pseudo-random stream generator. Such a pseudo-random stream generator may be a nonlinear feedback shift register, for example. All of the randomized techniques described in this section will be discussed in terms of pseudo-random stream generators whose keys are known to both sender and receiver. A class of ciphers related to the randomized stream ciphers presented here can be obtained by replacing the pseudo-random bit generators with cipher-feedback arrangements.

III.C.1. (if $P = 0$ then R else $(M \oplus P')$)

This scheme uses two pseudo-random bit stream generators P, P' and provides a way to intersperse nulls into the output stream. One of the pseudo-random bit stream generators is used to encrypt the message, while the other is used to select where random bits are to be interspersed into the ciphertext stream. Specifically, to generate the next ciphertext bit, a pseudo-random bit is taken from the generator P. If this bit is 0, the next ciphertext bit is obtained from the random bit generator R. Otherwise, the ciphertext bit is the XOR of the next message bit and the next output bit from the pseudo-random bit stream generator P'. We call this method *pseudo-randomly interspersing random bits after encryption.* See Figure 8.

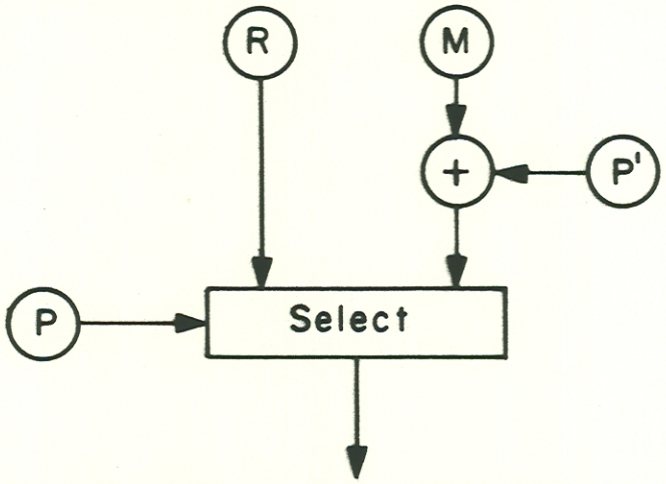


Figure 8. (if $P = 0$ then R else $(M \oplus P')$)

III.C.2. $(R \oplus P) \,||\, ((\text{if } R = 0 \text{ then } R' \text{ else } M) \oplus P')$

In this scheme, a random bit generator R is used to select where random bits are to be inserted into the message stream. The resulting stream is then encrypted by a pseudo-random bit stream generator P'. To enable the receiver to decipher the message, the bit sequence generated by R is also sent, encrypted by another pseudo-random bit stream generator P. We call this method *randomly interspersing random bits before encryption.* See Figure 9. Although this scheme and the previous one look promising, they have not yet undergone critical analysis.

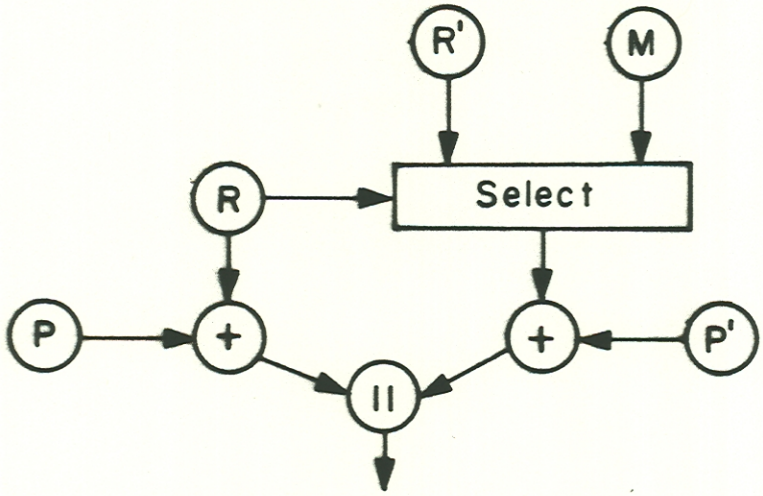


Figure 9. $(R \oplus P) \,||\, ((\text{if } R = 0 \text{ then } R' \text{ else } M) \oplus P')$

III.C.3. Random walk

We now propose a randomized stream cipher that is based on a a nondeterministic state transition function. In this scheme the encryption unit maintains a register of length n holding its current state. On each cycle, the state is updated and one message bit is encrypted and transmitted. To update the state, the state register is left-shifted one position, and a new truly random bit is brought in as the new rightmost bit of the state register. Each message bit is encrypted by XORing it with the value of a binary-valued function F applied to the current state. We call this method the *random walk* scheme because the encryptor takes a random walk in the state space.

To help the intended receiver decipher the message, the encryptor also sends with each encrypted message bit a hint about the current state. This hint is the result of applying a ternary-valued function F' to the current state. Thus, for each message bit, the corresponding ciphertext consists of one binary digit and one ternary digit. Decryption is still tricky since the intended receiver cannot always be certain what state transition took place in the encryption unit. One idea behind this scheme is the conjecture that cryptographic security can often be enhanced by making it difficult for even the intended receiver to decipher the message.

Assuming that F' yields each of the three possible output values equally often, it is possible to argue that the receiver will be able to adequately follow the encryptor's progress through state space. In some cases, however, the decoder may have to wait several cycles before being able to decode a given bit. Fortunately, since the transmitter can keep track of the degree of uncertainty in the receiver, the transmitter can avoid buffer overflow problems in the receiver by transmitting hints without ciphertext bits if necessary whenever the receiver happens to arrive at a situation of high ambiguity. It can be shown that binary hints are unsatisfactory for

this scheme. See Figure 10. Again, we urge the reader to attempt to cryptanalyze this scheme.

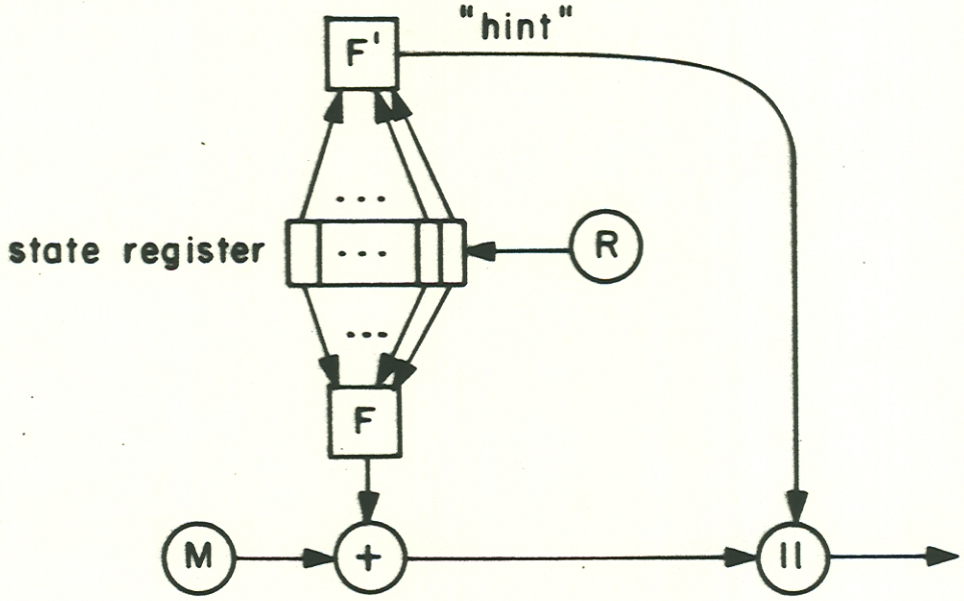


Figure 10. Random walk

IV. Extensions and applications

In this section, we will describe a method for presenting chosen ciphertext attacks and discuss how randomization can be used to combine cryptosystems.

IV.A. Preventing chosen ciphertext attacks

Although most of the schemes presented in section III.B. prevent the enemy from mounting chosen plaintext attacks against the underlying cryptographic functions, many of the schemes do not prevent chosen ciphertext attacks. For example, a chosen plaintext attack against the $(M \oplus R) \,||\, ER$ method amounts only to a known plaintext attack against the encryption function E, but a chosen ciphertext attack against the method amounts to a chosen ciphertext attack against E. Similarly, a chosen plaintext attack against the $(M \oplus FR) \,||\, R$ scheme amounts to a known plaintext attack against F, but a chosen ciphertext attack against the scheme amounts to a chosen plaintext against F. These chosen ciphertext attacks against the encryption schemes can be prevented by the standard technique of constructing the decryption unit so that it will output only properly constructed messages. For example, the decryption unit could be built so that it would output only messages that have valid cryptographic checksums appended to them, as illustrated in Figure 11. Because the enemy does not know the key to F, he should be unable to construct ciphertexts that would decrypt to messages with valid checksums, and should be unable to get the decryption unit to produce outputs. While chosen plaintext attacks can be prevented by adding random bits to the message, chosen ciphertext attacks can be avoided by adding redundancy to the message.

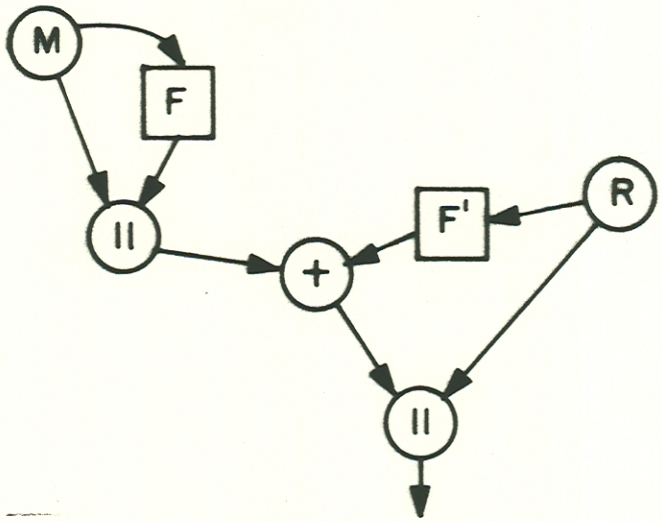


Figure 11. $((M \parallel FM) \oplus F'R) \parallel R$

IV.B. Combining Cryptosystems

Randomized encryption techniques provide powerful tools for combining cryptosystems. For example, Asmuth and Blakely's [AsB82] ingenious method for combining two cryptosystems E and E' depends crucially on randomized encryption. In our notation, their system is $(M \oplus R \oplus R') \parallel ER \parallel E'R'$, which may be viewed as a generalization of the $(M \oplus R) \parallel ER$ method. See Figure 12. Asmuth and Blakely conjecture that, provided the keys for the two component cryptosystems E and E' are independently selected, the resulting randomized cryptosystem is at least as hard to break as either E or E'. Unfortunately, Asmuth and Blakely were unable to rigorously prove their conjecture. We consider it an interesting open question to provide a more formal statement and proof of the Asmuth/Blakely conjecture.

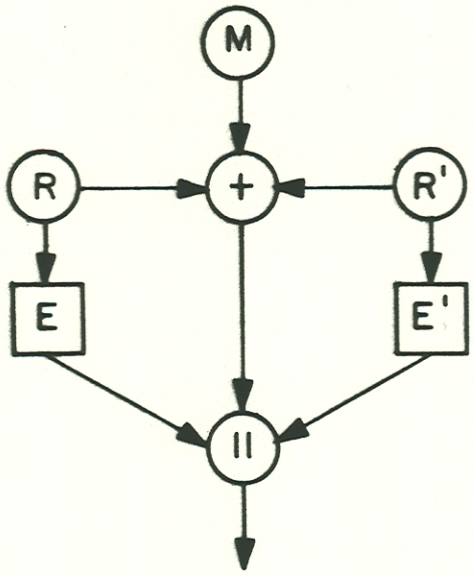


Figure 12. $(M \oplus R \oplus R') \,||\, ER \,||\, E'R'$

V. Directions for further research

We suggest several avenues for further investigation. First, the effect of randomized encryption techniques on specific cryptographic functions merits further study. Second, schemes involving multiple encryption functions and multiple random sources could be more thoroughly investigated. Third, it would be interesting to determine what operations other than XOR and || would be good methods for combining messages with random sequences. Among other operations, threshold schemes [Bla80] might be useful tools. Finally, it would be interesting to explore two-party randomized encryption systems. Known examples include Merkle's puzzle system [Mer78], Rabin's signature scheme [Rab78], Shamir, Rivest, and Adleman's mental poker protocols [SRA79], Rabin's oblivious transfer [Rab81], and Blum's protocol for exchanging secrets [Blu82].

VI. Summary

In this paper we have described many patterns in which randomization can be used in encryption for the purpose of enhancing cryptographic security. In the process, we have attempted to provide a unified framework against which recently discovered randomized cryptographic systems can be viewed. Although the schemes presented here represent only a small fraction of the available techniques, many of the methods not mentioned in this paper can be constructed by combining and

extending ideas discussed here. We hope that this paper will stimulate the reader to think about additional ways in which randomization can be used.

While this paper has attempted to provide a rough taxonomy of the available techniques, the use of randomization itself deserves much further careful study. In particular, it would be interesting to determine for various encryption functions how much cryptographic security is gained by different randomization techniques. This is a difficult question, however, because, to answer it, we must first thoroughly understand what cryptographic security means. Of course, whether or not the enhanced security of randomized encryption is worth its associated cost of bandwidth expansion will depend on the particular application. Fortunately, advances in communications technologies (*e.g.* fiber-optics) may help alleviate the drawbacks of increased bandwidth. We believe that randomization has been and will continue to be an important tool in cryptographic design.

VII. Acknowledgments

We would like to thank David Gifford for reading the manuscript and making several useful comments. In addition, we are grateful to Joseph Shipman and several of the CRYPTO 82 attendees who pointed out some examples of randomized encryption systems.

VIII. References

[AsB82] Asmuth, C. A., and G. R. Blakely. An efficient algorithm for constructing a cryptosystem which is harder to break than two other cryptosystems. *Comp. & Maths. with Appls.*, 7 (1981), 447–450.

[AvT82] Avis, G. M., and S. E. Tavares. A microprocessor based cryptosystem for secure message exchange. *Advances in Cryptography: Proceedings of CRYPTO 82*, Plenum Press, (New York, 1983).

[BMT78] Berlekamp, E. R., R. J. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. on Info. Theory*, **IT-24** (1978), 384–386.

[Bla80] Blakely, G. R. The Vernam one-time pad is a key safeguarding scheme, not a cryptosystem. *Proceedings of the 1980 IEEE Symposium on Security and Privacy*, (1980), 447–450.

[BlM82] Blum, Manuel, and Silvio Micali. How to generate cryptographically strong sequences of pseudo random bits. *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, (November, 1982), 112–117.

[Blu82] Blum, Manuel. How to exchange (secret) keys. *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, (May 1983), to appear.

[DiH79] Diffie, Whitfield, and Martin E. Hellman. Privacy and authentication: an introduction to cryptography. *Proceedings of the IEEE*, **67** (March 1979), 397–427.

[FIP77] FIPS Publication 46. Specifications for the Data Encryption Standard. U.S. Dept. of Commerce, National Bureau of Standards, (January 15, 1977).

[FIP80] FIPS Publication 81. DES modes of operation. U.S. Dept. of Commerce, National Bureau of Standards, (December 2, 1980).

[Gal68] Gallager, R. G. *Information Theory and Reliable Communication*, John Wiley, (New York, 1968).

[Gif82] Gifford, David K. Early experience with natural random bits. Seminar talk, MIT Laboratory for Computer Science, (May 11, 1982).

[GoM81] Goldwasser, Shafi, and Silvio Micali. A bit by bit secure public-key cryptosystem. Technical memo UCB/ERL M81/88, Univ. of California, Berkeley, (December 1981).

[GoM82] Goldwasser, Shafi, and Silvio Micali. Probabilistic encryption & how to play mental poker keeping all partial information secret. *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, (May 5-7, 1982), 365–377.

[Kah67] Kahn, David. *The Codebreakers: The Story of Secret Writing*, Macmillan, (New York, 1967).

[Kle60] Kleinrock, L. A program for testing sequences of random numbers. MIT Lincoln Laboratory Report 51G-0018, (October 25, 1960).

[Kru81] Kruh, Louis. The Genesis of the Jefferson/Bazeries Cipher Device. *Cryptologia*, **5** (October 1981), 193–208.

[Lem79] Lempel, Abraham. Cryptology in transition. *ACM Computing Surveys*, **11** (December 1979), 285–303.

[Mad72] Maddocks, R. S. *et al.* A compact and accurate generator for truly random binary digits. *Journal of Physics E: Scientific Instruments*, **5** (1972), 542-544.

[McE78] McEliece, R. J. A public-key cryptosystem based on algebraic coding theory. Deep Space Network Progress Report 42-22, Pasadena Jet Propulsion Labs., (January-February 1978), 114–116.

[Mer78] Merkle, Ralph C. Secure communications over insecure channels. *CACM*, **21** (April 1978), 294–299.

[Nic82] Nicolai, Carl R. Non deterministic cryptography. *Advances in Cryptography: Proceedings of CRYPTO 82*, Plenum Press, (New York, 1983).

[NiZ80] Niven, Ivan, and H. S. Zuckerman. *An Introduction to the Theory of Numbers*, John Wiley, (New York, 1980).

[Rab78] Rabin, Michael O. Digitalized signatures. *Foundations of Secure Computation*, (edited by DeMillo, *et al.*). Academic Press, (New York, 1978), 155–168.

[Rab79] Rabin, Michael O. Digitalized signatures and public-key functions as intractable as factorization. Technical report no. TR–212, MIT Lab. for Computer Science, (January 1979).

[Rab81] Rabin, Michael O. How to exchange secrets by oblivious transfer. Technical memo TR–81, Harvard Center for Research in Computing, (1981).

[SRA79] Shamir, Adi, Ronald Rivest, and Leonard Adleman. Mental poker. *The Mathematical Gardner*, (edited by D. Klarner), Prindle, Weber, and Schmidt, (Boston, 1981), 37–43.

[Sha49] Shannon, Claude E. Communication theory of secrecy systems. *Bell System Technical Journal*, **28** (October 1949), 659–715.

[Sim82] Simmons, Gustavus J., and Diane Holdridge. Forward search as a cryptanalytic tool against a public key privacy channel. Presented at the Symposium on Computer Security and Privacy, (Oakland, April 1982).

[Slo82] Sloane, N. J. A. Error-correcting codes and cryptography—part I. *Cryptologia*, **6** (April 1982), 128–153.

[Wyn75] Wyner, A. D. The wire-tap channel. *The Bell System Technical Journal*, **54** (October 1975), 1355–1387.