New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

Abstract—**Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.**

I. Introduction

WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

The development of computer controlled communication networks promises effortless and inexpensive contact between people or computers on opposite sides of the world, replacing most mail and many excursions with telecommunications. For many applications these contacts must be made secure against both eavesdropping and the injection of illegitimate messages. At present, however, the solution of security problems lags well behind other areas of communications technology. Contemporary cryptography is unable to meet the requirements, in that its use would impose such severe inconveniences on the system users, as to eliminate many of the benefits of teleprocessing.

Manuscript received June 3, 1976. This work was partially supported by the National Science Foundation under NSF Grant ENG 10173. Portions of this work were presented at the IEEE Information Theory Workshop, Lenox, MA, June 23–25, 1975 and the IEEE International Symposium on Information Theory in Ronneby, Sweden, June 21–24, 1976.

W. Diffie is with the Department of Electrical Engineering, Stanford University, Stanford, CA, and the Stanford Artificial Intelligence Laboratory, Stanford, CA 94305.

M. E. Hellman is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305.

The best known cryptographic problem is that of privacy: preventing the unauthorized extraction of information from communications over an insecure channel. In order to use cryptography to insure privacy, however, it is currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such as private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channels without compromising the security of the system. In a *public key cryptosystem* enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver is able to decipher it. As such, a public key cryptosystem is a multiple access cipher. A private conversation can therefore be held between any two individuals regardless of whether they have ever communicated before. Each one sends messages to the other enciphered in the receiver's public enciphering key and deciphers the messages he receives using his own secret deciphering key.

We propose some techniques for developing public key cryptosystems, but the problem is still largely open.

Public key distribution systems offer a different approach to eliminating the need for a secure key distribution channel. In such a system, two users who wish to exchange a key communicate back and forth until they arrive at a key in common. A third party eavesdropping on this exchange must find it computationally infeasible to compute the key from the information overheard. A possible solution to the public key distribution problem is given in Section III, and Merkle [1] has a partial solution of a different form.

A second problem, amenable to cryptographic solution, which stands in the way of replacing contemporary busi-

ness communications by teleprocessing systems is authentication. In current business, the validity of contracts is guaranteed by signatures. A signed contract serves as legal evidence of an agreement which the holder can present in court if necessary. The use of signatures, however, requires the transmission and storage of written contracts. In order to have a purely digital replacement for this paper instrument, each user must be able to produce a message whose authenticity can be checked by anyone, but which could not have been produced by anyone else, even the recipient. Since only one person can originate messages but many people can receive messages, this can be viewed as a broadcast cipher. Current electronic authentication techniques cannot meet this need.

Section IV discusses the problem of providing a true, digital, message dependent signature. For reasons brought out there, we refer to this as the one-way authentication problem. Some partial solutions are given, and it is shown how any public key cryptosystem can be transformed into a one-way authentication system.

Section V will consider the interrelation of various cryptographic problems and introduce the even more difficult problem of trap doors.

At the same time that communications and computation have given rise to new cryptographic problems, their offspring, information theory, and the theory of computation have begun to supply tools for the solution of important problems in classical cryptography.

The search for unbreakable codes is one of the oldest themes of cryptographic research, but until this century all proposed systems have ultimately been broken. In the nineteen twenties, however, the "one time pad" was invented, and shown to be unbreakable [2, pp. 398–400]. The theoretical basis underlying this and related systems was put on a firm foundation a quarter century later by information theory [3]. One time pads require extremely long keys and are therefore prohibitively expensive in most applications.

In contrast, the security of most cryptographic systems resides in the computational difficulty to the cryptanalyst of discovering the plaintext without knowledge of the key. This problem falls within the domains of computational complexity and analysis of algorithms, two recent disciplines which study the difficulty of solving computational problems. Using the results of these theories, it may be possible to extend proofs of security to more useful classes of systems in the foreseeable future. Section VI explores this possibility.

Before proceeding to newer developments, we introduce terminology and define threat environments in the next section.

II. Conventional Cryptography

Cryptography is the study of "mathematical" systems for solving two kinds of security problems: privacy and authentication. A privacy system prevents the extraction of information by unauthorized parties from messages transmitted over a public channel, thus assuring the sender of a message that it is being read only by the intended recipient. An authentication system prevents the unauthorized injection of messages into a public channel, assuring the receiver of a message of the legitimacy of its sender.

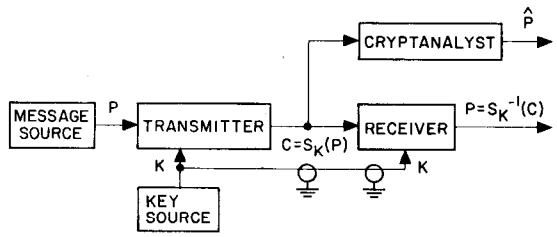

Fig. 1. Flow of information in conventional cryptographic system.

A channel is considered public if its security is inadequate for the needs of its users. A channel such as a telephone line may therefore be considered private by some users and public by others. Any channel may be threatened with eavesdropping or injection or both, depending on its use. In telephone communication, the threat of injection is paramount, since the called party cannot determine which phone is calling. Eavesdropping, which requires the use of a wiretap, is technically more difficult and legally hazardous. In radio, by comparison, the situation is reversed. Eavesdropping is passive and involves no legal hazard, while injection exposes the illegitimate transmitter to discovery and prosecution.

Having divided our problems into those of privacy and authentication we will sometimes further subdivide authentication into message authentication, which is the problem defined above, and user authentication, in which the only task of the system is to verify that an individual is who he claims to be. For example, the identity of an individual who presents a credit card must be verified, but there is no message which he wishes to transmit. In spite of this apparent absence of a message in user authentication, the two problems are largely equivalent. In user authentication, there is an implicit message "I AM USER X," while message authentication is just verification of the identity of the party sending the message. Differences in the threat environments and other aspects of these two subproblems, however, sometimes make it convenient to distinguish between them.

Fig. 1 illustrates the flow of information in a conventional cryptographic system used for privacy of communications. There are three parties: a transmitter, a receiver, and an eavesdropper. The transmitter generates a plaintext or unenciphered message P to be communicated over an insecure channel to the legitimate receiver. In order to prevent the eavesdropper from learning P, the transmitter operates on P with an invertible transformation S_K to produce the ciphertext or cryptogram $C = S_K(P)$. The key K is transmitted only to the legitimate receiver via a secure channel, indicated by a shielded path in Fig. 1. Since the legitimate receiver knows K, he can decipher C by operating with $S_K{}^{-1}$ to obtain $S_K{}^{-1}(C) = S_K{}^{-1}(S_K(P)) = P$, the original plaintext message. The secure channel cannot

be used to transmit P itself for reasons of capacity or delay. For example, the secure channel might be a weekly courier and the insecure channel a telephone line.

A *cryptographic system* is a single parameter family $\{S_K\}_{K\in\{K\}}$ of invertible transformations

$$S_K:\{P\} \rightarrow \{C\} \tag{1}$$

from a space $\{P\}$ of plaintext messages to a space $\{C\}$ of ciphertext messages. The parameter K is called the key and is selected from a finite set $\{K\}$ called the keyspace. If the message spaces $\{P\}$ and $\{C\}$ are equal, we will denote them both by $\{M\}$. When discussing individual cryptographic transformations S_K, we will sometimes omit mention of the system and merely refer to the transformation K.

The goal in designing the cryptosystem $\{S_K\}$ is to make the enciphering and deciphering operations inexpensive, but to ensure that any successful cryptanalytic operation is too complex to be economical. There are two approaches to this problem. A system which is secure due to the computational cost of cryptanalysis, but which would succumb to an attack with unlimited computation, is called *computationally secure*; while a system which can resist any cryptanalytic attack, no matter how much computation is allowed, is called *unconditionally secure*. Unconditionally secure systems are discussed in [3] and [4] and belong to that portion of information theory, called the Shannon theory, which is concerned with optimal performance obtainable with unlimited computation.

Unconditional security results from the existence of multiple meaningful solutions to a cryptogram. For example, the simple substitution cryptogram *XMD* resulting from English text can represent the plaintext messages: now, and, the, etc. A computationally secure cryptogram, in contrast, contains sufficient information to uniquely determine the plaintext and the key. Its security resides solely in the cost of computing them.

The only unconditionally secure system in common use is the *one time pad*, in which the plaintext is combined with a randomly chosen key of the same length. While such a system is provably secure, the large amount of key required makes it impractical for most applications. Except as otherwise noted, this paper deals with computationally secure systems since these are more generally applicable. When we talk about the need to develop provably secure cryptosystems we exclude those, such as the one time pad, which are unwieldly to use. Rather, we have in mind systems using only a few hundred bits of key and implementable in either a small amount of digital hardware or a few hundred lines of software.

We will call a task *computationally infeasible* if its cost as measured by either the amount of memory used or the runtime is finite but impossibly large.

Much as error correcting codes are divided into convolutional and block codes, cryptographic systems can be divided into two broad classes: *stream ciphers* and *block ciphers*. Stream ciphers process the plaintext in small chunks (bits or characters), usually producing a pseudorandom sequence of bits which is added modulo 2 to the bits of the plaintext. Block ciphers act in a purely combinatorial fashion on large blocks of text, in such a way that a small change in the input block produces a major change in the resulting output. This paper deals primarily with block ciphers, because this *error propagation* property is valuable in many authentication applications.

In an authentication system, cryptography is used to guarantee the authenticity of the message to the receiver. Not only must a meddler be prevented from injecting totally new, authentic looking messages into a channel, but he must be prevented from creating apparently authentic messages by combining, or merely repeating, old messages which he has copied in the past. A cryptographic system intended to guarantee privacy will not, in general, prevent this latter form of mischief.

To guarantee the authenticity of a message, information is added which is a function not only of the message and a secret key, but of the date and time as well; for example, by attaching the date and time to each message and encrypting the entire sequence. This assures that only someone who possesses the key can generate a message which, when decrypted, will contain the proper date and time. Care must be taken, however, to use a system in which small changes in the ciphertext result in large changes in the deciphered plaintext. This intentional error propagation ensures that if the deliberate injection of noise on the channel changes a message such as "erase file 7" into a different message such as "erase file 8," it will also corrupt the authentication information. The message will then be rejected as inauthentic.

The first step in assessing the adequacy of cryptographic systems is to classify the threats to which they are to be subjected. The following threats may occur to cryptographic systems employed for either privacy or authentication.

A *ciphertext only attack* is a cryptanalytic attack in which the cryptanalyst possesses only ciphertext.

A *known plaintext attack* is a cryptanalytic attack in which the cryptanalyst possesses a substantial quantity of corresponding plaintext and ciphertext.

A *chosen plaintext attack* is a cryptanalytic attack in which the cryptanalyst can submit an unlimited number of plaintext messages of his own choosing and examine the resulting cryptograms.

In all cases it is assumed that the opponent knows the general system $\{S_K\}$ in use since this information can be obtained by studying a cryptographic device. While many users of cryptography attempt to keep their equipment secret, many commercial applications require not only that the general system be public but that it be standard.

A ciphertext only attack occurs frequently in practice. The cryptanalyst uses only knowledge of the statistical properties of the language in use (e.g., in English, the letter e occurs 13 percent of the time) and knowledge of certain "probable" words (e.g., a letter probably begins "Dear Sir:"). It is the weakest threat to which a system can be subjected, and any system which succumbs to it is considered totally insecure.

A system which is secure against a known plaintext attack frees its users from the need to keep their past messages secret, or to paraphrase them prior to declassification. This is an unreasonable burden to place on the system's users, particularly in commercial situations where product announcements or press releases may be sent in encrypted form for later public disclosure. Similar situations in diplomatic correspondence have led to the cracking of many supposedly secure systems. While a known plaintext attack is not always possible, its occurrence is frequent enough that a system which cannot resist it is not considered secure.

A chosen plaintext attack is difficult to achieve in practice, but can be approximated. For example, submitting a proposal to a competitor may result in his enciphering it for transmission to his headquarters. A cipher which is secure against a chosen plaintext attack thus frees its users from concern over whether their opponents can plant messages in their system.

For the purpose of certifying systems as secure, it is appropriate to consider the more formidable cryptanalytic threats as these not only give more realistic models of the working environment of a cryptographic system, but make the assessment of the system's strength easier. Many systems which are difficult to analyze using a ciphertext only attack can be ruled out immediately under known plaintext or chosen plaintext attacks.

As is clear from these definitions, cryptanalysis is a system identification problem. The known plaintext and chosen plaintext attacks correspond to passive and active system identification problems, respectively. Unlike many subjects in which system identification is considered, such as automatic fault diagnosis, the goal in cryptography is to build systems which are difficult, rather than easy, to identify.

The chosen plaintext attack is often called an IFF attack, terminology which descends from its origin in the development of cryptographic "identification friend or foe" systems after World War II. An IFF system enables military radars to distinguish between friendly and enemy planes automatically. The radar sends a time-varying challenge to the airplane which receives the challenge, encrypts it under the appropriate key, and sends it back to the radar. By comparing this response with a correctly encrypted version of the challenge, the radar can recognize a friendly aircraft. While the aircraft are over enemy territory, enemy cryptanalysts can send challenges and examine the encrypted responses in an attempt to determine the authentication key in use, thus mounting a chosen plaintext attack on the system. In practice, this threat is countered by restricting the form of the challenges, which need not be unpredictable, but only nonrepeating.

There are other threats to authentication systems which cannot be treated by conventional cryptography, and which require recourse to the new ideas and techniques introduced in this paper. The *threat of compromise of the receiver's authentication data* is motivated by the situation in multiuser networks where the receiver is often the system itself. The receiver's password tables and other authentication data are then more vulnerable to theft than those of the transmitter (an individual user). As shown later, some techniques for protecting against this threat also protect against the *threat of dispute*. That is, a message may be sent but later repudiated by either the transmitter or the receiver. Or, it may be alleged by either party that a message was sent when in fact none was. Unforgeable digital signatures and receipts are needed. For example, a dishonest stockbroker might try to cover up unauthorized buying and selling for personal gain by forging orders from clients, or a client might disclaim an order actually authorized by him but which he later sees will cause a loss. We will introduce concepts which allow the receiver to verify the authenticity of a message, but prevent him from generating apparently authentic messages, thereby protecting against both the threat of compromise of the receiver's authentication data and the threat of dispute.

III. Public Key Cryptography

As shown in Fig. 1, cryptography has been a derivative security measure. Once a secure channel exists along which keys can be transmitted, the security can be extended to other channels of higher bandwidth or smaller delay by encrypting the messages sent on them. The effect has been to limit the use of cryptography to communications among people who have made prior preparation for cryptographic security.

In order to develop large, secure, telecommunications systems, this must be changed. A large number of users n results in an even larger number, $(n^2 - n)/2$ potential pairs who may wish to communicate privately from all others. It is unrealistic to assume either that a pair of users with no prior acquaintance will be able to wait for a key to be sent by some secure physical means, or that keys for all $(n^2 - n)/2$ pairs can be arranged in advance. In another paper [5], the authors have considered a conservative approach requiring no new development in cryptography itself, but this involves diminished security, inconvenience, and restriction of the network to a starlike configuration with respect to initial connection protocol.

We propose that it is possible to develop systems of the type shown in Fig. 2, in which two parties communicating solely over a public channel and using only publicly known techniques can create a secure connection. We examine two approaches to this problem, called public key cryptosys-

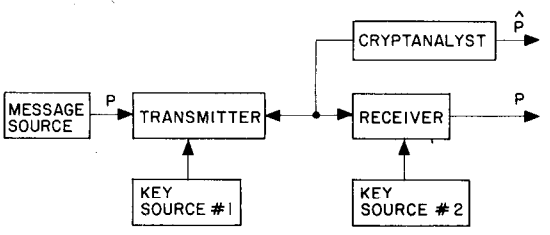

Fig. 2. Flow of information in public key system.

tems and public key distribution systems, respectively. The first are more powerful, lending themselves to the solution of the authentication problems treated in the next section, while the second are much closer to realization.

A *public key cryptosystem* is a pair of families $\{E_K\}_{K \in \{K\}}$ and $\{D_K\}_{K \in \{K\}}$ of algorithms representing invertible transformations,

$$E_K: \{M\} \rightarrow \{M\} \tag{2}$$

$$D_K: \{M\} \rightarrow \{M\} \tag{3}$$

on a finite message space $\{M\}$, such that

1) for every $K \in \{K\}$, E_K is the inverse of D_K,
2) for every $K \in \{K\}$ and $M \in \{M\}$, the algorithms E_K and D_K are easy to compute,
3) for almost every $K \in \{K\}$, each easily computed algorithm equivalent to D_K is computationally infeasible to derive from E_K,
4) for every $K \in \{K\}$, it is feasible to compute inverse pairs E_K and D_K from K.

Because of the third property, a user's enciphering key E_K can be made public without compromising the security of his secret deciphering key D_K. The cryptographic system is therefore split into two parts, a family of enciphering transformations and a family of deciphering transformations in such a way that, given a member of one family, it is infeasible to find the corresponding member of the other.

The fourth property guarantees that there is a feasible way of computing corresponding pairs of inverse transformations when no constraint is placed on what either the enciphering or deciphering transformation is to be. In practice, the cryptoequipment must contain a true random number generator (e.g., a noisy diode) for generating K, together with an algorithm for generating the $E_K - D_K$ pair from its outputs.

Given a system of this kind, the problem of key distribution is vastly simplified. Each user generates a pair of inverse transformations, E and D, at his terminal. The deciphering transformation D must be kept secret, but need never be communicated on any channel. The enciphering key E can be made public by placing it in a public directory along with the user's name and address. Anyone can then encrypt messages and send them to the user, but no one else can decipher messages intended for him. Public key cryptosystems can thus be regarded as *multiple access ciphers*.

It is crucial that the public file of enciphering keys be protected from unauthorized modification. This task is made easier by the public nature of the file. Read protection is unnecessary and, since the file is modified infrequently, elaborate write protection mechanisms can be economically employed.

A suggestive, although unfortunately useless, example of a public key cryptosystem is to encipher the plaintext, represented as a binary n-vector $\boldsymbol{m}$, by multiplying it by an invertible binary $n \times n$ matrix E. The cryptogram thus equals $E\boldsymbol{m}$. Letting $D = E^{-1}$ we have $\boldsymbol{m} = D\boldsymbol{c}$. Thus, both enciphering and deciphering require about n^2 operations. Calculation of D from E, however, involves a matrix inversion which is a harder problem. And it is at least conceptually simpler to obtain an arbitrary pair of inverse matrices than it is to invert a given matrix. Start with the identity matrix I and do elementary row and column operations to obtain an arbitrary invertible matrix E. Then starting with I do the inverses of these same elementary operations in reverse order to obtain $D = E^{-1}$. The sequence of elementary operations could be easily determined from a random bit string.

Unfortunately, matrix inversion takes only about n^3 operations. The ratio of "cryptanalytic" time (i.e., computing D from E) to enciphering or deciphering time is thus at most n, and enormous block sizes would be required to obtain ratios of 10^6 or greater. Also, it does not appear that knowledge of the elementary operations used to obtain E from I greatly reduces the time for computing D. And, since there is no round-off error in binary arithmetic, numerical stability is unimportant in the matrix inversion. In spite of its lack of practical utility, this matrix example is still useful for clarifying the relationships necessary in a public key cryptosystem.

A more practical approach to finding a pair of easily computed inverse algorithms E and D; such that D is hard to infer from E, makes use of the difficulty of analyzing programs in low level languages. Anyone who has tried to determine what operation is accomplished by someone else's machine language program knows that E itself (i.e., what E does) can be hard to infer from an algorithm for E. If the program were to be made purposefully confusing through addition of unneeded variables and statements, then determining an inverse algorithm could be made very difficult. Of course, E must be complicated enough to prevent its identification from input–output pairs.

Essentially what is required is a one-way compiler: one which takes an easily understood program written in a high level language and translates it into an incomprehensible program in some machine language. The compiler is one-way because it must be feasible to do the compilation, but infeasible to reverse the process. Since efficiency in size of program and run time are not crucial in this application, such compilers may be possible if the structure of the machine language can be optimized to assist in the confusion.

Merkle [1] has independently studied the problem of distributing keys over an insecure channel. His approach is different from that of the public key cryptosystems suggested above, and will be termed a *public key distribution system*. The goal is for two users, A and B, to securely exchange a key over an insecure channel. This key is then used by both users in a normal cryptosystem for both enciphering and deciphering. Merkle has a solution whose cryptanalytic cost grows as n^2 where n is the cost to the legitimate users. Unfortunately the cost to the legitimate users of the system is as much in transmission time as in computation, because Merkle's protocol requires n

potential keys to be transmitted before one key can be decided on. Merkle notes that this high transmission overhead prevents the system from being very useful in practice. If a one megabit limit is placed on the setup protocol's overhead, his technique can achieve cost ratios of approximately 10 000 to 1, which are too small for most applications. If inexpensive, high bandwidth data links become available, ratios of a million to one or greater could be achieved and the system would be of substantial practical value.

We now suggest a new public key distribution system which has several advantages. First, it requires only one "key" to be exchanged. Second, the cryptanalytic effort appears to grow exponentially in the effort of the legitimate users. And, third, its use can be tied to a public file of user information which serves to authenticate user A to user B and vice versa. By making the public file essentially a read only memory, one personal appearance allows a user to authenticate his identity many times to many users. Merkle's technique requires A and B to verify each other's identities through other means.

The new technique makes use of the apparent difficulty of computing logarithms over a finite field $GF(q)$ with a prime number q of elements. Let

$$Y = \alpha^X \bmod q, \qquad \text{for } 1 \leq X \leq q-1, \tag{4}$$

where α is a fixed primitive element of $GF(q)$, then X is referred to as the logarithm of Y to the base α, mod q:

$$X = \log_\alpha Y \bmod q, \qquad \text{for } 1 \leq Y \leq q-1. \tag{5}$$

Calculation of Y from X is easy, taking at most $2 \times \log_2 q$ multiplications [6, pp. 398–422]. For example, for $X = 18$,

$$Y = \alpha^{18} = (((\alpha^2)^2)^2)^2 \times \alpha^2. \tag{6}$$

Computing X from Y, on the other hand can be much more difficult and, for certain carefully chosen values of q, requires on the order of $q^{1/2}$ operations, using the best known algorithm [7, pp. 9, 575–576], [8].

The security of our technique depends crucially on the difficulty of computing logarithms mod q, and if an algorithm whose complexity grew as $\log_2 q$ were to be found, our system would be broken. While the simplicity of the problem statement might allow such simple algorithms, it might instead allow a proof of the problem's difficulty. For now we assume that the best known algorithm for computing logs mod q is in fact close to optimal and hence that $q^{1/2}$ is a good measure of the problem's complexity, for a properly chosen q.

Each user generates an independent random number X_i chosen uniformly from the set of integers $\{1,2,\cdots,q-1\}$. Each keeps X_i secret, but places

$$Y_i = \alpha^{X_i} \bmod q \tag{7}$$

in a public file with his name and address. When users i and j wish to communicate privately, they use

$$K_{ij} = \alpha^{X_i X_j} \bmod q \tag{8}$$

as their key. User i obtains K_{ij} by obtaining Y_j from the public file and letting

$$K_{ij} = Y_j^{X_i} \bmod q \tag{9}$$

$$= (\alpha^{X_j})^{X_i} \bmod q \tag{10}$$

$$= \alpha^{X_j X_i} = \alpha^{X_i X_j} \bmod q. \tag{11}$$

User j obtains K_{ij} in the similar fashion

$$K_{ij} = Y_i^{X_j} \bmod q. \tag{12}$$

Another user must compute K_{ij} from Y_i and Y_j, for example, by computing

$$K_{ij} = Y_i^{(\log_\alpha Y_j)} \bmod q. \tag{13}$$

We thus see that if logs mod q are easily computed the system can be broken. While we do not currently have a proof of the converse (i.e., that the system is secure if logs mod q are difficult to compute), neither do we see any way to compute K_{ij} from Y_i and Y_j without first obtaining either X_i or X_j.

If q is a prime slightly less than 2^b, then all quantities are representable as b bit numbers. Exponentiation then takes at most $2b$ multiplications mod q, while by hypothesis taking logs requires $q^{1/2} = 2^{b/2}$ operations. The cryptanalytic effort therefore grows exponentially relative to legitimate efforts. If $b = 200$, then at most 400 multiplications are required to compute Y_i from X_i, or K_{ij} from Y_i and X_j, yet taking logs mod q requires 2^{100} or approximately 10^{30} operations.

IV. One-Way Authentication

The problem of authentication is perhaps an even more serious barrier to the universal adoption of telecommunications for business transactions than the problem of key distribution. Authentication is at the heart of any system involving contracts and billing. Without it, business cannot function. Current electronic authentication systems cannot meet the need for a purely digital, unforgeable, message dependent signature. They provide protection against third party forgeries, but do not protect against disputes between transmitter and receiver.

In order to develop a system capable of replacing the current written contract with some purely electronic form of communication, we must discover a digital phenomenon with the same properties as a written signature. It must be easy for anyone to recognize the signature as authentic, but impossible for anyone other than the legitimate signer to produce it. We will call any such technique *one-way authentication*. Since any digital signal can be copied precisely, a true digital signature must be recognizable without being known.

Consider the "login" problem in a multiuser computer system. When setting up his account, the user chooses a password which is entered into the system's password directory. Each time he logs in, the user is again asked to provide his password. By keeping this password secret from all other users, forged logins are prevented. This,

however, makes it vital to preserve the security of the password directory since the information it contains would allow perfect impersonation of any user. The problem is further compounded if system operators have legitimate reasons for accessing the directory. Allowing such legitimate accesses, but preventing all others, is next to impossible.

This leads to the apparently impossible requirement for a new login procedure capable of judging the authenticity of passwords without actually knowing them. While appearing to be a logical impossibility, this proposal is easily satisfied. When the user first enters his password PW, the computer automatically and transparently computes a function $f(PW)$ and stores this, not PW, in the password directory. At each successive login, the computer calculates $f(X)$, where X is the proffered password, and compares $f(X)$ with the stored value $f(PW)$. If and only if they are equal, the user is accepted as being authentic. Since the function f must be calculated once per login, its computation time must be small. A million instructions (costing approximately \$0.10 at bicentennial prices) seems to be a reasonable limit on this computation. If we could ensure, however, that calculation of f^{-1} required 10^{30} or more instructions, someone who had subverted the system to obtain the password directory could not in practice obtain PW from $f(PW)$, and could thus not perform an unauthorized login. Note that $f(PW)$ is not accepted as a password by the login program since it will automatically compute $f(f(PW))$ which will not match the entry $f(PW)$ in the password directory.

We assume that the function f is public information, so that it is not ignorance of f which makes calculation of f^{-1} difficult. Such functions are called one-way functions and were first employed for use in login procedures by R. M. Needham [9, p. 91]. They are also discussed in two recent papers [10], [11] which suggest interesting approaches to the design of one-way functions.

More precisely, a function f is a *one-way function* if, for any argument x in the domain of f, it is easy to compute the corresponding value $f(x)$, yet, for almost all y in the range of f, it is computationally infeasible to solve the equation $y = f(x)$ for any suitable argument x.

It is important to note that we are defining a function which is not invertible from a computational point of view, but whose noninvertibility is entirely different from that normally encountered in mathematics. A function f is normally called "noninvertible" when the inverse of a point y is not unique, (i.e., there exist distinct points x_1 and x_2 such that $f(x_1) = y = f(x_2)$). We emphasize that this is not the sort of inversion difficulty that is required. Rather, it must be overwhelmingly difficult, given a value y and knowledge of f, to calculate any x whatsoever with the property that $f(x) = y$. Indeed, if f is noninvertible in the usual sense, it may make the task of finding an inverse image easier. In the extreme, if $f(x) \equiv y_0$ for all x in the domain, then the range of f is $\{y_0\}$, and we can take any x as $f^{-1}(y_0)$. It is therefore necessary that f not be too degenerate. A small degree of degeneracy is tolerable and, as discussed later, is probably present in the most promising class of one-way functions.

Polynomials offer an elementary example of one-way functions. It is much harder to find a root x_0 of the polynomial equation $p(x) = y$ than it is to evaluate the polynomial $p(x)$ at $x = x_0$. Purdy [11] has suggested the use of sparse polynomials of very high degree over finite fields, which appear to have very high ratios of solution to evaluation time. The theoretical basis for one-way functions is discussed at greater length in Section VI. And, as shown in Section V, one-way functions are easy to devise in practice.

The one-way function login protocol solves only some of the problems arising in a multiuser system. It protects against compromise of the system's authentication data when it is not in use, but still requires the user to send the true password to the system. Protection against eavesdropping must be provided by additional encryption, and protection against the threat of dispute is absent altogether.

A public key cryptosystem can be used to produce a true one-way authentication system as follows. If user A wishes to send a message M to user B, he "deciphers" it in his secret deciphering key and sends $D_A(M)$. When user B receives it, he can read it, and be assured of its authenticity by "enciphering" it with user A's public enciphering key E_A. B also saves $D_A(M)$ as proof that the message came from A. Anyone can check this claim by operating on $D_A(M)$ with the publicly known operation E_A to recover M. Since only A could have generated a message with this property, the solution to the one-way authentication problem would follow immediately from the development of public key cryptosystems.

One-way message authentication has a partial solution suggested to the authors by Leslie Lamport of Massachusetts Computer Associates. This technique employs a one-way function f mapping k-dimensional binary space into itself for k on the order of 100. If the transmitter wishes to send an N bit message he generates $2N$, randomly chosen, k-dimensional binary vectors $x_1, X_1, x_2, X_2, \cdots, x_N, X_N$ which he keeps secret. The receiver is given the corresponding images under f, namely $y_1, Y_1, y_2, Y_2, \cdots, y_N, Y_N$. Later, when the message $\boldsymbol{m} = (m_1, m_2, \cdots, m_N)$ is to be sent, the transmitter sends x_1 or X_1 depending on whether $m_1 = 0$ or 1. He sends x_2 or X_2 depending on whether $m_2 = 0$ or 1, etc. The receiver operates with f on the first received block and sees whether it yields y_1 or Y_1 as its image and thus learns whether it was x_1 or X_1, and whether $m_1 = 0$ or 1. In a similar manner the receiver is able to determine $m_2, m_3, \cdots, m_N$. But the receiver is incapable of forging a change in even one bit of $\boldsymbol{m}$.

This is only a partial solution because of the approximately 100-fold data expansion required. There is, however, a modification which eliminates the expansion problem when N is roughly a megabit or more. Let g be a one-way mapping from binary N-space to binary n-space where n is approximately 50. Take the N bit message $\boldsymbol{m}$

and operate on it with g to obtain the n bit vector $\boldsymbol{m}'$. Then use the previous scheme to send $\boldsymbol{m}'$. If $N = 10^6$, $n = 50$, and $k = 100$, this adds $kn = 5000$ authentication bits to the message. It thus entails only a 5 percent data expansion during transmission (or 15 percent if the initial exchange of $y_1, Y_1, \cdots, y_N, Y_N$ is included). Even though there are a large number of other messages (2^{N-n} on the average) with the same authentication sequence, the one-wayness of g makes them computationally infeasible to find and thus to forge. Actually g must be somewhat stronger than a normal one-way function, since an opponent has not only $\boldsymbol{m}'$ but also one of its inverse images $\boldsymbol{m}$. It must be hard even given $\boldsymbol{m}$ to find a different inverse image of $\boldsymbol{m}'$. Finding such functions appears to offer little trouble (see Section V).

There is another partial solution to the one-way user authentication problem. The user generates a password X which he keeps secret. He gives the system $f^T(X)$, where f is a one-way function. At time t the appropriate authenticator is $f^{T-t}(X)$, which can be checked by the system by applying $f^t(X)$. Because of the one-wayness of f, past responses are of no value in forging a new response. The problem with this solution is that it can require a fair amount of computation for legitimate login (although many orders of magnitude less than for forgery). If for example t is incremented every second and the system must work for one month on each password then $T = 2.6$ million. Both the user and the system must then iterate f an average of 1.3 million times per login. While not insurmountable, this problem obviously limits use of the technique. The problem could be overcome if a simple method for calculating $f^{(2^n)}$, for $n = 1,2, \cdots$ could be found, much as $X^8 = ((X^2)^2)^2$. For then binary decompositions of $T - t$ and t would allow rapid computation of f^{T-t} and f^t. It may be, however, that rapid computation of f^n precludes f from being one-way.

V. Problem Interrelations and Trap Doors

In this section, we will show that some of the cryptographic problems presented thus far can be reduced to others, thereby defining a loose ordering according to difficulty. We also introduce the more difficult problem of trap doors.

In Section II we showed that a cryptographic system intended for privacy can also be used to provide authentication against third party forgeries. Such a system can be used to create other cryptographic objects, as well.

A cryptosystem which is secure against a known plaintext attack can be used to produce a one-way function.

As indicated in Fig. 3, take the cryptosystem $\{S_K : \{P\} \rightarrow \{C\}\}_{K \in \{K\}}$ which is secure against a known plaintext attack, fix $P = P_0$ and consider the map

$$f : \{K\} \rightarrow \{C\} \tag{14}$$

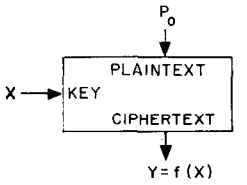

Fig. 3. Secure cryptosystem used as one-way function.

defined by

$$f(X) = S_X(P_0). \tag{15}$$

This function is one-way because solving for X given $f(X)$ is equivalent to the cryptanalytic problem of finding the key from a single known plaintext–cryptogram pair. Public knowledge of f is now equivalent to public knowledge of $\{S_K\}$ and P_0.

While the converse of this result is not necessarily true, it is possible for a function originally found in the search for one-way functions to yield a good cryptosystem. This actually happened with the discrete exponential function discussed in Section III [8].

One-way functions are basic to both block ciphers and key generators. A key generator is a pseudorandom bit generator whose output, the keystream, is added modulo 2 to a message represented in binary form, in imitation of a one-time pad. The key is used as a "seed" which determines the pseudorandom keystream sequence. A known plaintext attack thus reduces to the problem of determining the key from the keystream. For the system to be secure, computation of the key from the keystream must be computationally infeasible. While, for the system to be usable, calculation of the keystream from the key must be computationally simple. Thus a good key generator is, almost by definition, a one-way function.

Use of either type of cryptosystem as a one way function suffers from a minor problem. As noted earlier, if the function f is not uniquely invertible, it is not necessary (or possible) to find the actual value of X used. Rather any X with the same image will suffice. And, while each mapping S_K in a cryptosystem must be bijective, there is no such restriction on the function f from key to cryptogram defined above. Indeed, guaranteeing that a cryptosystem has this property appears quite difficult. In a good cryptosystem the mapping f can be expected to have the characteristics of a randomly chosen mapping (i.e., $f(X_i)$ is chosen uniformly from all possible Y, and successive choices are independent). In this case, if X is chosen uniformly and there are an equal number of keys and messages (X and Y), then the probability that the resultant Y has $k + 1$ inverses is approximately $e^{-1}/k!$ for $k = 0,1,2,3, \cdots$. This is a Poisson distribution with mean $\lambda = 1$, shifted by 1 unit. The expected number of inverses is thus only 2. While it is possible for f to be more degenerate, a good cryptosystem will not be too degenerate since then the key is not being well used. In the worst case, if $f(X) \equiv Y_0$ for some Y_0, we have $S_K(P_0) \equiv C_0$, and encipherment of P_0 would not depend on the key at all!

While we are usually interested in functions whose domain and range are of comparable size, there are exceptions. In the previous section we required a one-way function mapping long strings onto much shorter ones. By using a block cipher whose key length is larger than the blocksize, such functions can be obtained using the above technique.

Evans *et al.* [10] have a different approach to the problem of constructing a one-way function from a block cipher. Rather than selecting a fixed P_0 as the input, they use the function

$$f(X) = S_X(X). \tag{16}$$

This is an attractive approach because equations of this form are generally difficult to solve, even when the family S is comparatively simple. This added complexity, however, destroys the equivalence between the security of the system S under a known plaintext attack and the one-wayness of f.

Another relationship has already been shown in Section IV.

A public key cryptosystem can be used to generate a one-way authentication system.

The converse does not appear to hold, making the construction of a public key cryptosystem a strictly more difficult problem than one-way authentication. Similarly, a public key cryptosystem can be used as a public key distribution system, but not conversely.

Since in a public key cryptosystem the general system in which E and D are used must be public, specifying E specifies a complete algorithm for transforming input messages into output cryptograms. As such a public key system is really a set of *trap-door one-way functions*. These are functions which are not really one-way in that simply computed inverses exist. But given an algorithm for the forward function it is computationally infeasible to find a simply computed inverse. Only through knowledge of certain *trap-door information* (e.g., the random bit string which produced the E-D pair) can one easily find the easily computed inverse.

Trap doors have already been seen in the previous paragraph in the form of *trap-door one-way functions*, but other variations exist. A *trap-door cipher* is one which strongly resists cryptanalysis by anyone not in possession of *trap-door information* used in the design of the cipher. This allows the designer to break the system after he has sold it to a client and yet falsely to maintain his reputation as a builder of secure systems. It is important to note that it is not greater cleverness or knowledge of cryptography which allows the designer to do what others cannot. If he were to lose the trap-door information he would be no better off than anyone else. The situation is precisely analogous to a combination lock. Anyone who knows the combination can do in seconds what even a skilled locksmith would require hours to accomplish. And yet, if he forgets the combination, he has no advantage.

A trap-door cryptosystem can be used to produce a public key distribution system.

For A and B to establish a common private key, A chooses a key at random and sends an arbitrary plaintext–cryptogram pair to B. B, who made the trap-door cipher public, but kept the trap-door information secret, uses the plaintext-cryptogram pair to solve for the key. A and B now have a key in common.

There is currently little evidence for the existence of trap-door ciphers. However they are a distinct possibility and should be remembered when accepting a cryptosystem from a possible opponent [12].

By definition, we will require that a trap-door problem be one in which it is computationally feasible to devise the trap door. This leaves room for yet a third type of entity for which we shall use the prefix "quasi." For example a *quasi one-way function* is not one-way in that an easily computed inverse exists. However, it is computationally infeasible even for the designer, to find the easily computed inverse. Therefore a quasi one-way function can be used in place of a one-way function with essentially no loss in security.

Losing the trap-door information to a trap-door one-way function makes it into a quasi one-way function, but there may also be one-way functions not obtainable in this manner.

It is entirely a matter of definition that quasi one-way functions are excluded from the class of one-way functions. One could instead talk of one-way functions in the wide sense or in the strict sense.

Similarly, a quasi secure cipher is a cipher which will successfully resist cryptanalysis, even by its designer, and yet for which there exists a computationally efficient cryptanalytic algorithm (which is of course computationally infeasible to find). Again, from a practical point of view, there is essentially no difference between a secure cipher and a quasi secure one.

We have already seen that public key cryptosystems imply the existence of trap-door one-way functions. However the converse is not true. For a trap-door one-way function to be usable as a public key cryptosystem, it must be invertible (i.e., have a unique inverse.)

VI. Computational Complexity

Cryptography differs from all other fields of endeavor in the ease with which its requirements may appear to be satisfied. Simple transformations will convert a legible text into an apparently meaningless jumble. The critic, who wishes to claim that meaning might yet be recovered by cryptanalysis, is then faced with an arduous demonstration if he is to prove his point of view correct. Experience has shown, however, that few systems can resist the concerted attack of skillful cryptanalysts, and many supposedly secure systems have subsequently been broken.

In consequence of this, judging the worth of new systems has always been a central concern of cryptographers. During the sixteenth and seventeenth centuries, mathematical arguments were often invoked to argue the strength of cryptographic methods, usually relying on counting methods which showed the astronomical number

of possible keys. Though the problem is far too difficult to be laid to rest by such simple methods, even the noted algebraist Cardano fell into this trap [2, p. 145]. As systems whose strength had been so argued were repeatedly broken, the notion of giving mathematical proofs for the security of systems fell into disrepute and was replaced by certification via crypanalytic assault.

During this century, however, the pendulum has begun to swing back in the other direction. In a paper intimately connected with the birth of information theory, Shannon [3] showed that the one time pad system, which had been in use since the late twenties offered "perfect secrecy" (a form of unconditional security). The provably secure systems investigated by Shannon rely on the use of either a key whose length grows linearly with the length of the message or on perfect source coding and are therefore too unwieldy for most purposes. We note that neither public key cryptosystems nor one-way authentication systems can be unconditionally secure because the public information always determines the secret information uniquely among the members of a finite set. With unlimited computation, the problem could therefore be solved by a straightforward search.

The past decade has seen the rise of two closely related disciplines devoted to the study of the costs of computation: computational complexity theory and the analysis of algorithms. The former has classified known problems in computing into broad classes by difficulty, while the latter has concentrated on finding better algorithms and studying the resources they consume. After a brief digression into complexity theory, we will examine its application to cryptography, particularly the analysis of one-way functions.

A function is said to belong to the complexity class ***P*** (for polynomial) if it can be computed by a deterministic Turing Machine in a time which is bounded above by some polynomial function of the length of its input. One might think of this as the class of easily computed functions, but it is more accurate to say that a function not in this class must be hard to compute for at least some inputs. There are problems which are known not to be in the class ***P*** [13, pp. 405–425].

There are many problems which arise in engineering which cannot be solved in polynomial time by any known techniques, unless they are run on a computer with an unlimited degree of parallelism. These problems may or may not belong to the class ***P***, but belong to the class ***NP*** (for nondeterministic, polynomial) of problems solvable in polynomial time on a "nondeterministic" computer (i.e., one with an unlimited degree of parallelism). Clearly the class ***NP*** includes the class ***P***, and one of the great open questions in complexity theory is whether the class ***NP*** is strictly larger.

Among the problems known to be solvable in ***NP*** time, but not known to be solvable in ***P*** time, are versions of the traveling salesman problem, the satisfiability problem for propositional calculus, the knapsack problem, the graph coloring problem, and many scheduling and minimization problems [13, pp. 363–404], [14]. We see that it is not lack of interest or effort which has prevented people from finding solutions in ***P*** time for these problems. It is thus strongly believed that at least one of these problems must not be in the class ***P***, and that therefore the class ***NP*** is strictly larger.

Karp has identified a subclass of the ***NP*** problems, called ***NP*** complete, with the property that if any one of them is in ***P***, then all ***NP*** problems are in ***P***. Karp lists 21 problems which are ***NP*** complete, including all of the problems mentioned above [14].

While the ***NP*** complete problems show promise for cryptographic use, current understanding of their difficulty includes only worst case analysis. For cryptographic purposes, typical computational costs must be considered. If, however, we replace worst case computation time with average or typical computation time as our complexity measure, the current proofs of the equivalences among the ***NP*** complete problems are no longer valid. This suggests several interesting topics for research. The ensemble and typicality concepts familiar to information theorists have an obvious role to play.

We can now identify the position of the general cryptanalytic problem among all computational problems.

The cryptanalytic difficulty of a system whose encryption and decryption operations can be done in ***P*** *time cannot be greater than* ***NP.***

To see this, observe that any cryptanalytic problem can be solved by finding a key, inverse image, etc., chosen from a finite set. Choose the key nondeterministically and verify in ***P*** time that it is the correct one. If there are M possible keys to choose from, an M-fold parallelism must be employed. For example in a known plaintext attack, the plaintext is encrypted simultaneously under each of the keys and compared with the cryptogram. Since, by assumption, encryption takes only ***P*** time, the cryptanalysis takes only ***NP*** time.

We also observe that the general cryptanalytic problem is ***NP*** complete. This follows from the breadth of our definition of cryptographic problems. A one-way function with an ***NP*** complete inverse will be discussed next.

Cryptography can draw directly from the theory of ***NP*** complexity by examining the way in which ***NP*** complete problems can be adapted to cryptographic use. In particular, there is an ***NP*** complete problem known as the knapsack problem which lends itself readily to the construction of a one-way function.

Let $y = f(x) = \boldsymbol{a} \cdot \boldsymbol{x}$ where $\boldsymbol{a}$ is a known vector of n intergers $(a_1, a_2, \cdots, a_n)$ and $\boldsymbol{x}$ is a binary n-vector. Calculation of y is simple, involving a sum of at most n integers. The problem of inverting f is known as the knapsack problem and requires finding a subset of the $\{a_i\}$ which sum to y.

Exhaustive search of all 2^n subsets grows exponentially and is computationally infeasible for n greater than 100 or so. Care must be exercised, however, in selecting the parameters of the problem to ensure that shortcuts are not possible. For example if $n = 100$ and each a_i is 32 bits long, y is at most 39 bits long, and f is highly degenerate; re-

quiring on the average only 2^{38} tries to find a solution. Somewhat more trivially, if $a_i = 2^{i-1}$ then inverting f is equivalent to finding the binary decomposition of y.

This example demonstrates both the great promise and the considerable shortcomings of contemporary complexity theory. The theory only tells us that the knapsack problem is probably difficult in the worst case. There is no indication of its difficulty for any particular array. It appears, however, that choosing the $\{a_i\}$ uniformly from $\{0,1,2,\cdots,2^{n-1}\}$ results in a hard problem with probability one as $n \to \infty$.

Another potential one-way function, of interest in the analysis of algorithms, is exponentiation mod q, which was suggested to the authors by Prof. John Gill of Stanford University. The one-wayness of this functions has already been discussed in Section III.

VII. Historical Perspective

While at first the public key systems and one-way authentication systems suggested in this paper appear to be unportended by past cryptographic developments, it is possible to view them as the natural outgrowth of trends in cryptography stretching back hundreds of years.

Secrecy is at the heart of cryptography. In early cryptography, however, there was a confusion about what was to be kept secret. Cryptosystems such as the Caesar cipher (in which each letter is replaced by the one three places further on, so A is carried to D, B to E, etc.) depended for their security on keeping the entire encryption process secret. After the invention of the telegraph [2, p. 191], the distinction between a general system and a specific key allowed the general system to be compromised, for example by theft of a cryptographic device, without compromising future messages enciphered in new keys. This principle was codified by Kerchoffs [2, p. 235] who wrote in 1881 that the compromise of a cryptographic system should cause no inconvenience to the correspondents. About 1960, cryptosystems were put into service which were deemed strong enough to resist a known plaintext cryptanalytic attack, thereby eliminating the burden of keeping old messages secret. Each of these developments decreased the portion of the system which had to be protected from public knowledge, eliminating such tedious expedients as paraphrasing diplomatic dispatches before they were presented. Public key systems are a natural continuation of this trend toward decreasing secrecy.

Prior to this century, cryptographic systems were limited to calculations which could be carried out by hand or with simple slide-rule-like devices. The period immediately after World War I saw the beginning of a revolutionary trend which is now coming to fruition. Special purpose machines were developed for enciphering. Until the development of general purpose digital hardware, however, cryptography was limited to operations which could be performed with simple electromechanical systems. The development of digital computers has freed it from the limitations of computing with gears and has allowed the search for better encryption methods according to purely cryptographic criteria.

The failure of numerous attempts to demonstrate the soundness of cryptographic systems by mathematical proof led to the paradigm of certification by cryptanalytic attack set down by Kerchoffs [2, p. 234] in the last century. Although some general rules have been developed, which aid the designer in avoiding obvious weaknesses, the ultimate test is an assault on the system by skilled cryptanalysts under the most favorable conditions (e.g., a chosen plaintext attack). The development of computers has led for the first time to a mathematical theory of algorithms which can begin to approach the difficult problem of estimating the computational difficulty of breaking a cryptographic system. The position of mathematical proof may thus come full circle and be reestablished as the best method of certification.

The last characteristic which we note in the history of cryptography is the division between amateur and professional cryptographers. Skill in production cryptanalysis has always been heavily on the side of the professionals, but innovation, particularly in the design of new types of cryptographic systems, has come primarily from the amateurs. Thomas Jefferson, a cryptographic amateur, invented a system which was still in use in World War II [2, pp. 192–195], while the most noted cryptographic system of the twentieth century, the rotor machine, was invented simultaneously by four separate people, all amateurs [2, pp. 415, 420, 422–424]. We hope this will inspire others to work in this fascinating area in which participation has been discouraged in the recent past by a nearly total government monopoly.

References

[1] R. Merkle, "Secure communication over an insecure channel," submitted to *Communications of the ACM*.
[2] D. Kahn, *The Codebreakers, The Story of Secret Writing*. New York: Macmillan, 1967.
[3] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, pp. 656–715, Oct. 1949.
[4] M. E. Hellman, "An extension of the Shannon theory approach to cryptography," submitted to *IEEE Trans. Inform. Theory*, Sept. 1975.
[5] W. Diffie and M. E. Hellman, "Multiuser cryptographic techniques," presented at National Computer Conference, New York, June 7–10, 1976.
[6] D. Knuth, *The Art of Computer Programming, Vol. 2, Semi-Numerical Algorithms*. Reading, MA.: Addison–Wesley, 1969.
[7] ——, *The Art of Computer Programming, Vol. 3, Sorting and Searching*. Reading, MA.: Addison–Wesley, 1973.
[8] S. Pohlig and M. E. Hellman, "An improved algorithm for computing algorithms in $GF(p)$ and its cryptographic significance," submitted to *IEEE Trans. Inform. Theory*.
[9] M. V. Wilkes, *Time-Sharing Computer Systems*. New York: Elsevier, 1972.
[10] A. Evans, Jr., W. Kantrowitz, and E. Weiss, "A user authentication system not requiring secrecy in the computer," *Communications of the ACM*, vol. 17, pp. 437–442, Aug. 1974.
[11] G. B. Purdy, "A high security log-in procedure," *Communications of the ACM*, vol. 17, pp. 442–445, Aug. 1974.
[12] W. Diffie and M. E. Hellman, "Cryptanalysis of the NBS data encryption standard" submitted to *Computer*, May 1976.
[13] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, MA.: Addison–Wesley, 1974.
[14] R. M, Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. R. E. Miller and J. W. Thatcher, Eds. New York: Plenum, 1972, pp. 85–104.