SCIENTIFIC AMERICAN

Cryptography and Computer Privacy

Author(s): Horst Feistel

Source: *Scientific American*, Vol. 228, No. 5 (May 1973), pp. 15-23

Published by: Scientific American, a division of Nature America, Inc.

Stable URL: https://www.jstor.org/stable/10.2307/24923044

JSTOR is a not-for-profit service that helps scholars, researchers, and students discover, use, and build upon a wide range of content in a trusted digital archive. We use information technology and tools to increase productivity and facilitate new forms of scholarship. For more information about JSTOR, please contact support@jstor.org.

Your use of the JSTOR archive indicates your acceptance of the Terms & Conditions of Use, available at https://about.jstor.org/terms

Scientific American, a division of Nature America, Inc. is collaborating with JSTOR to digitize, preserve and extend access to *Scientific American*

This content downloaded from 185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms

Established 1845 SCIENTIFIC AMERICAN May 1973 Volume 228 Number 5

Cryptography and Computer Privacy

Computer systems in general and personal "data banks" in particular need protection. This can be achieved by enciphering all material and authenticating the legitimate origin of any command to the computer

by Horst Feistel

There is growing concern that computers now constitute, or will soon constitute, a dangerous threat to individual privacy. Since many computers contain personal data and are accessible from distant terminals, they are viewed as an unexcelled means of assembling large amounts of information about an individual or a group. It is asserted that it will soon be feasible to compile dossiers in depth on an entire citizenry, where until recently the material for such dossiers was scattered in many separate locations under widely diverse jurisdictions. It will be argued here, however, that a computer system can be adapted to guard its contents from everyone but authorized individuals by enciphering the material in forms highly resistant to cipher-breaking.

Traditionally those most dependent on secrecy have been military men and diplomats. Their work often calls for the element of surprise, and surprise implies secrecy. Whatever need ordinary people have had for secrecy has remained essentially an individual problem and has rarely been of public concern; lovers and thieves have solved their requirements for communications privacy as best they could. This state of affairs changed little until about the middle of the 19th century. At about that time scientific methods and modes of thought were finally enlisted to improve the techniques of cryptography. Nevertheless, the techniques employed in secret communication remained largely pencil-and-paper operations until well into this century.

Cryptographic encipherment can be achieved in two essentially different ways: by ciphers or by codes. A helpful distinction between the two is as follows. A cipher always assigns substitute symbols to some given set of alphabet letters. Being alphabetic in character, the cipher enables one to say anything that one can print on a typewriter in the particular language that can be represented by its keyboard letters. This means that with a cipher one can say things that have never been said before or even been anticipated as needing to be said. A code, on the other hand, is intrinsically semantic in character. A code can convey only meanings thought of in advance and provided for in a secret list such as a code book.

One should perhaps mention that today the word "code" is frequently used in a sense that does not imply cryptography. The word then usually has a broad meaning embracing sets of symbols with special relations. Thus one speaks of error-detection and error-correction codes, data-compression codes, codes for telecommunications purposes, commercial codes and codes involving all kinds of highly intricate electrical signals and wave forms.

In tackling the privacy problem presented by modern computers at the Thomas J. Watson Research Center of the International Business Machines Corporation we have given the central role to cipher techniques. It will not be possible here to cover the entire subject of "data bank" confidentiality and the securing of computer operations. I do hope to show, however, that certain principles underlying data encipherment and authentication of sources are pertinent to these broad problems.

In modern machine-to-machine networks of the type needed for a data bank the notion of secrecy embraces more than mere message concealment. A data bank includes a network of terminal-to-computer communications. The communications lines connecting the terminals and the computer centers are wide open not only to tapping but also to deliberate alteration and corruption of traffic. In addition to the more obvious aspects of data secrecy, one must therefore provide adequate protection against operational deception of the system. Mere error-detection will not do. The system itself must make it extremely unlikely that an unauthorized but clever and sophisticated person can either enter, withdraw or alter commands or data in such a system.

That is vitally important, because the slightest amount of false data entered in a data-bank system, whether by accident or by intent, can make much of the system's operation worthless. Computers without adequate protection are easily deceived, particularly by operators who thoroughly understand their operation. Indeed, the terminals them-

© 1973 SCIENTIFIC AMERICAN, INC

This content downloaded from 185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms

CLEAR	CIPHER 1	CIPHER 2
A	F	○
B	E	□
C	K	#
D	J	△
E	N	♡
F	P	∃
G	O	✳
H	C	▨
I	D	✠
J	Y	⊗
K	U	▽
L	W	☽
M	H	◐
N	M	⊖
O	B	∞
P	Z	⋒
Q	L	⬡
R	V	◪
S	X	◎
T	G	⌂
U	T	Σ
V	R	⊠
W	S	┼
X	I	⧈
Y	Q	☼
Z	A	✕

CLEAR TEXT

S	E	N	D	M	O	N	E	Y

CIPHER 1

X	N	M	J	H	B	M	N	Q

CIPHER 2

◎	♡	⊖	△	◐	∞	⊖	♡	☼

selves can be misused for cleverly designed data insertion. In order to preserve the purity of a data bank it will be necessary to authenticate the legitimate origin and character of any data received and to do so rapidly and with very high margins of safety.

Let us begin with the most elementary fact about ciphers: All cryptography amounts to substitution. In its simplest form a substitution can be defined by a tableau, or table. The left side lists the ordinary letters of the alphabet in "clear text"; the right side lists their cipher equivalents, the substituted values. The amateur is frequently impressed with the enormous number of possibilities of arranging such substitution, or permuting, alphabets. For the English alphabet with 26 letters there are $1 \times 2 \times 3 \ldots \times 26$ ways of writing down unique substitution alphabets. (Such a product is referred to as 26 "factorial," written 26!. Any number n! is the product of all the integers from 1 to n.)

The n! possible permutations of a tableau with n entries constitute the number of possible "keys." Now, 26! is a very large number, more than 4×10^{26}. Even so, any simple alphabetic substitution can be broken easily by frequency analysis. If the letter Q occurs more frequently than any other in a fairly long sample of cipher text, the analyst can be pretty sure that a cipher Q stands for E, the letter that occurs more frequently than any other in English clear text, and so on.

Nothing is gained, of course, by replacing the letters of a substitution alphabet with mysterious-looking symbols. An analyst could not care less how complicated the substitute symbols may look. He simply replaces them with a normal alphabet or number substitute of his own and proceeds with his frequency analysis. The mysterious-symbol approach does, however, demonstrate the flexibility with regard to substitute symbols. Since substitution tableaus, in spite of their well-known weaknesses, are of basic importance in cryptographic design, let us consider the possibility of introducing genuinely useful substitution symbols.

This can indeed be done. The binary number system, consisting of just two symbols, 0 and 1, is ideally suited for cryptographic processing by computers. With n binary digits one can generate 2^n distinct binary codes. Thus with a "block" of five binary digits one can generate 2^5, or 32, distinct combinations, or more than enough to encode the 26 letters of the alphabet [*see top illustration on opposite page*]. If we wish to name or label more items, we can increase our reservoir of distinct binary numbers merely by increasing the size of the digit block. Every time we increase the block size by one digit position we double the number of possible codes. Hence a six-digit code would provide 2^6, or 64, distinct codes, or enough to include numerals, punctuation marks and so on.

In addition to the ease with which binary digits can be represented in electronic circuitry (by on and off signals, for example) they have all the advantages of ordinary decimal numbers. Thus binary numbers can be added, subtracted, multiplied and so on, just as ordinary decimal numbers can. As we shall see, arithmetic manipulation plays a vital role in cryptographic techniques designed for computers.

Now we can introduce complexity in a fundamentally different way. Instead of using just one substitution table we can use several, in some disarranged but prearranged order. The ordering pattern constitutes a key. If we use only two tables, tagged 0 and 1, then a typical key might be 1101101. We are now dealing with multialphabetic substitution. With this new source of complexity available the question arises: Can one not now simplify the substitution tables, perhaps by making them smaller? The simplest binary substitution one can perform is on single message digits, in which case there are only two possible distinct substitution tables. Let us therefore set up two substitution tables, one for each of the two basic types of key [*see bottom illustration on opposite page*]. The table marked Key 1 simply interchanges the 0's and 1's; the table marked Key 0 preserves their identity. These are the only two possibilities. Now, it happens that the same effect can be obtained by the operation known as addition modulo 2: two digits of the same kind add up to 0, two digits of the opposite kind add up to 1. And so, in this special case, the pattern key can be called an addition key.

Before proceeding further let me explain that from here on we shall tacitly assume that the messages we wish to handle cryptographically have first been translated into a sequence of binary digits. Any kind of information, be it letters, musical sounds or television sig-

SUBSTITUTION, the basic operation in cryptography, is illustrated by two tables in which clear-text letters of the alphabet are replaced by cipher equivalents: other letters (*middle column*) or arbitrary symbols.

© 1973 SCIENTIFIC AMERICAN, INC

This content downloaded from
185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms

nals, can be represented by binary coding. We shall no longer worry about the original meaning of the messages. We shall deal only with their binary representation.

We are now ready to see what happens when we encipher a sequence of binary digits, let us say 0001010, into a new sequence using the two keys, Key 1 and Key 0, in some arbitrary sequence: 1101101. Remembering the rule that Key 1 interchanges 0's and 1's and that Key 0 leaves digits unchanged, we get the following:

Message: 0001010
Key: + 1101101
Cipher: 1100111

This represents addition modulo 2. It has the convenient property that subtraction is the same as addition, so that the message can be recovered simply by adding the sequence of digits in the key (which is known to the intended recipient) to the sequence in the cipher:

Cipher: 1100111
Key: − 1101101
Message: 0001010

The question immediately arises: Is this simple cipher of any practical value? Since the cipher in effect employs only two substitution tables of minimal size it is perhaps obvious that we must switch from one to the other frequently and in random fashion, that is, add a random sequence of key digits. Suppose we do. The astonishing answer is that we then have a potentially undecipherable cipher. From the viewpoint of information theory, what this cipher does is to add to each bit of message information one bit of key information (misinformation!). This is enough to completely destroy any kind of structure the original message may have had, provided that the key digits are picked at random, say by flipping a coin, and that the key sequence has the same length as the message and does not ever repeat.

Why is this system genuinely undecipherable? Actually it is no "system" at all. The basic cryptographic transformation amounts to no more than random addition of single digits. It is as trivial as that. The method derives its strength solely from the fact that for each message digit we completely and randomly change the key. This is the only class of ciphers that one can prove to be undecipherable in the absolute sense.

Even if an opponent made a brute-force attempt to break the system, for

CLEAR	BINARY EQUIVALENT	BINARY CIPHER
A	00000	00100
B	00001	01001
C	00010	10001
D	00011	01010
E	00100	00011
F	00101	10011
⋮	⋮	⋮
X	10111	11101
Y	11000	01110
Z	11001	10110

TRANSLATION TO BINARY SYSTEM is required for computer cryptography. A binary substitution-table cipher is constructed, for example, by first translating each letter of the alphabet into a five-digit binary number and then enciphering each binary equivalent.

a

KEY 0

CLEAR	CIPHER
0	0
1	1

KEY 1

CLEAR	CIPHER
0	1
1	0

b

KEY	MESSAGE 0	MESSAGE 1
0	0	1
1	1	0

c

MESSAGE		KEY		CIPHER
0	+	0	=	0
0	+	1	=	1
1	+	0	=	1
1	+	1	=	0

d

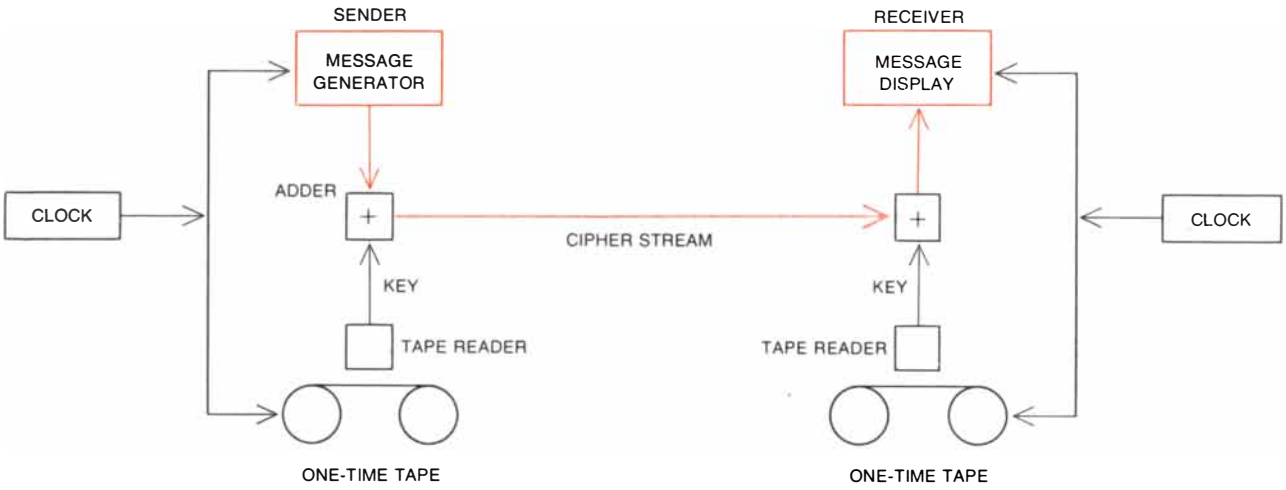
F = 00101
KEY = 10110
CIPHER = 10011

CIPHER = 10011
KEY = 10110
00101 = F

BINARY SUBSTITUTION in its simplest form allows only two possibilities (*a*): in one table (*Key 0*) there is no change from clear to cipher; in the other table (*Key 1*) the cipher is the opposite of the clear. In this case of binary substitution the two tables can be represented by a single addition table modulo 2 (*b*); in this case binary substitution is equivalent to binary addition (*c*). The cipher key for addition modulo 2 is an arbitrary sequence of 1's or 0's. To encipher the binary equivalent of a message letter, one adds a key digit to each message digit (*d*). To decipher, one subtracts the key (the same as adding modulo 2).

© 1973 SCIENTIFIC AMERICAN, INC

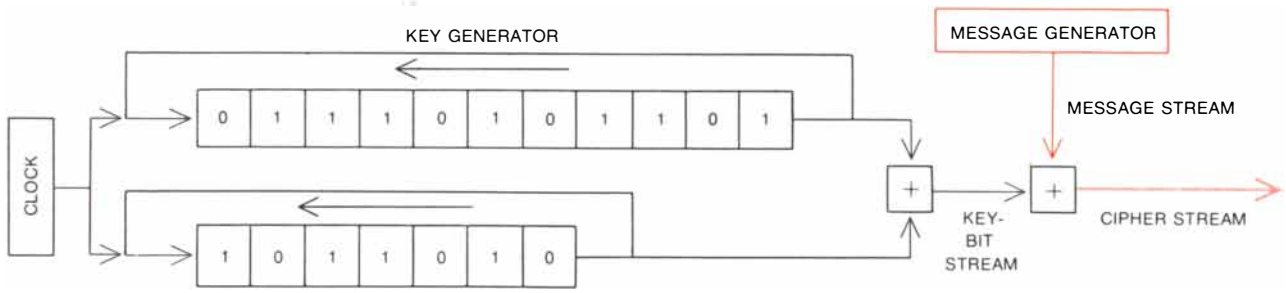
This content downloaded from
185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms



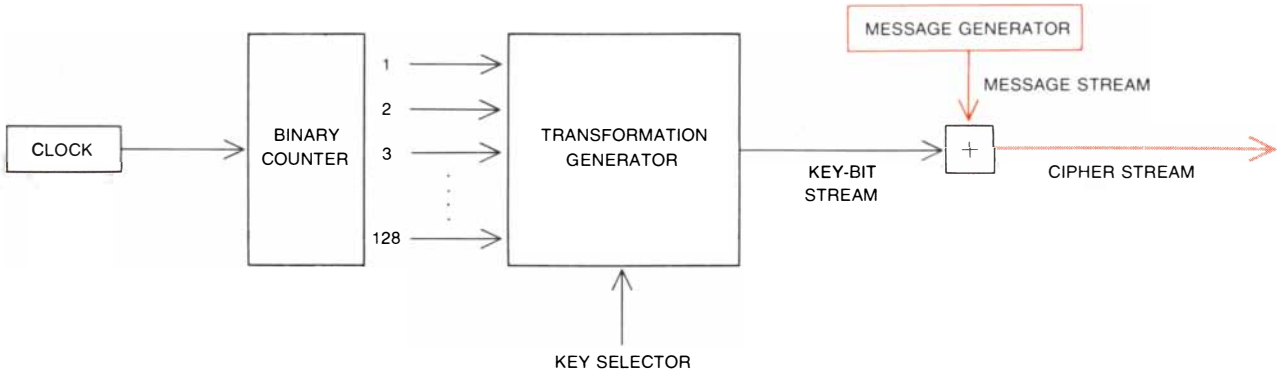
"ONE-TIME-TAPE" SYSTEM requires that the sender and the receiver have tapes carrying the same key, synchronized by clock devices. The digits of the key and of the message are added, the resulting cipher stream is transmitted and the key is subtracted (added modulo 2). For large traffic volumes a vast store of key digits must be conveyed securely to the receiver and filed there.

example by trying all the possible addition keys (2^6, or 64, in the case of our six-digit message), he would get all possible clear texts, including the one we had actually enciphered. Thus if we had enciphered the name "Tom" (which would actually take a minimum of 15 binary digits), the analyst would find among his trial decipherments all English three-letter names, such as Joe, Jim, Job and so on, including Tom, but no clue as to which name was correct. Not even a god or demon who could try all possible keys in an instant could ever establish any consistency. The system is well known and is in actual use by all major governments under various names, such as the Vernam system or "one-time pad."

In a practical system two identical tapes of random key digits are prepared; the tapes can be of any type—printed, punched, magnetic or whatever. One tape is held by the sender and the other is transmitted by "noninterceptible means," say by guarded courier, to the legitimate receiver. When the sender wants to transmit a message, he converts it first to binary form and places it in an apparatus that adds a digit, modulo 2,



PSEUDORANDOM-TAPE SYSTEM incorporates two tape loops of key digits, or bits, that are added together to generate a "pseudorandom" key-bit stream that is in turn added to the message stream as in the one-time-tape method and is deciphered by an identically generated key at the receiver. Short tapes thus simulate long ones, but the resulting internal periodicities may be useful to an analyst.



PSEUDORANDOM STREAM can be generated in more complex forms. In this generalized representation a binary counter provides the input numbers for the transformation generator, which supplies the key-stream digits to be added to the message stream.

© 1973 SCIENTIFIC AMERICAN, INC

This content downloaded from
185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms

from the key tape to each message digit [*see top illustration on opposite page*]. At the receiver the coded message is recorded and run through a machine similar to the encoder that adds (subtracts) a digit, modulo 2, from the key tape to each digit of the coded message, thereby yielding the clear text. Naturally the key tape must be stepped with absolute synchronization to duplicate the encoding operation.

The fundamental drawback to the Vernam system is that for every "bit" of information transmitted the recipient needs to have in his possession in advance one bit of key information. Moreover, these bits must be in a random sequence that cannot be used a second time. For large traffic volumes that is a severe restriction. Because of this requirement the Vernam system has usually been reserved for top-secret messages.

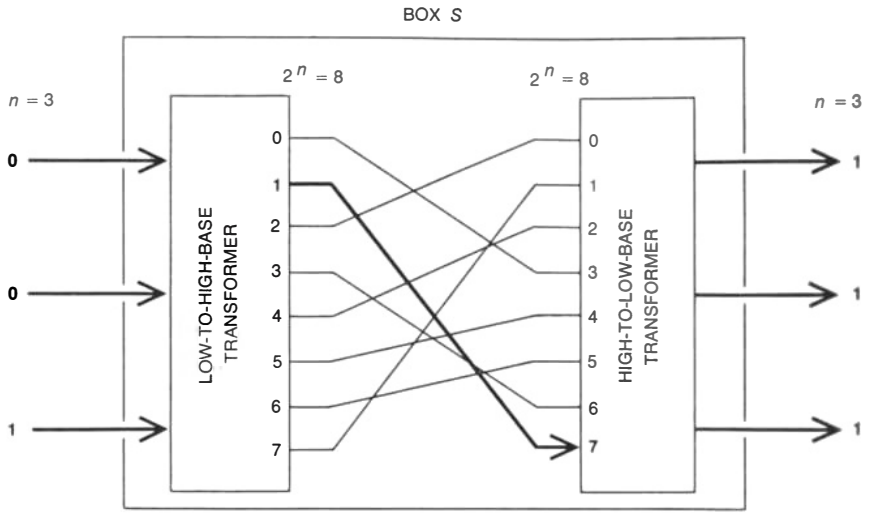
To get around the problem of supplying the recipient with large volumes of random key digits in advance, designers and inventors have devised many ingenious schemes for generating very long pseudorandom streams of digits from several short streams according to some algorithm. The recipient of a coded message can then be provided with a generator that operates exactly like the one used to add pseudorandom digits to the original message. An algorithm of course implies systematic routines, which in turn introduce regularities for which an analyst may search.

One basic method for building such a generator is to use two or more key-bit tapes that are added, bit by bit, to yield a compounded stream. For example, the simple one-time tape can be replaced with two tape loops of prime or relative-prime lengths. In this situation the lengths have no common factors and the compounded stream has the length of the product of the component streams: two tapes of lengths 1,000 and 1,001 respectively produce a compounded stream that will not repeat for 1,000 × 1,001, or 1,001,000, digits. The tapes are circulated through an adder that makes a modulo-2 addition of the digits sampled, one from each tape [*see middle illustration on opposite page*]. The output of the adder serves as the key digit for encoding the message. It is therefore important that the compounded stream exceed the length of all the messages one expects to send over a reasonable period of time.

Since any bit-by-bit adder is linear, it is inherently weak, but it can be cryptographically strengthened in any number of ways. One can pile complication on complication by introducing intricate feedback arrangements, perhaps linked in some fashion to the message, or by introducing such nonlinear mathematical operations as substitutions on digit blocks of suitable size. The nonsecret cryptographic literature contains many fascinating designs for pseudorandom-stream generators, all of which can in principle be reduced to one basic scheme [*see bottom illustration on opposite page*]. One way or another they generate a pseudorandom number by performing complex mathematical operations on some ordered sequence of input numbers, thereby transforming it in a way that is supposed to confound an analyst.

The reader will be surprised and disappointed to learn that the Vernam class of ciphers—the only class that can be proved to be undecipherable in the absolute sense—is not perfect at all in another sense: it does not in itself offer protection against clever deception on nonredundant traffic. Whether a message is encoded by the use of truly random digits or by pseudorandom-stream digits, in bit-by-bit encipherment a single error that arises during transmission is confined to a single digit position; the error does not "propagate," or spread, through the rest of the message. The cipher does not introduce intersymbol dependence. When the message itself is in a "natural" language such as English text, the normal redundancy context makes it easy for a human reader to spot an occasional error. Thus if some of the five binary digits representing the letter *E* were corrupted to turn it into the binary sequence for *F* (so that, for example, SECRET came out SECRFT), a human interpreter would detect the error immediately.

The situation is quite different in a



IN	OUT
000	011
001	**111**
010	000
011	110
100	010
101	100
110	101
111	001

SUBSTITUTION BOX, unlike the stream devices, is general, and includes both linear and nonlinear transformations: it does not merely add a 0 or 1 to input digits but can substitute for any input digit block any output digit block. It consists essentially of two switches. One converts a binary number of n digits into a one-digit number base 2^n; the other reconverts. The box therefore provides 2^n internal terminals that can be wired in $n!$ (n factorial, or $1 \times 2 \times 3 \ldots \times n$) different ways; in the case of this box with $n = 3$, that means 40,320 different wirings, or different tableaus like the one illustrated. A box of this kind with $n = 128$ would defeat analysis, but it is not possible to realize technologically.

© 1973 SCIENTIFIC AMERICAN, INC

This content downloaded from 185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms

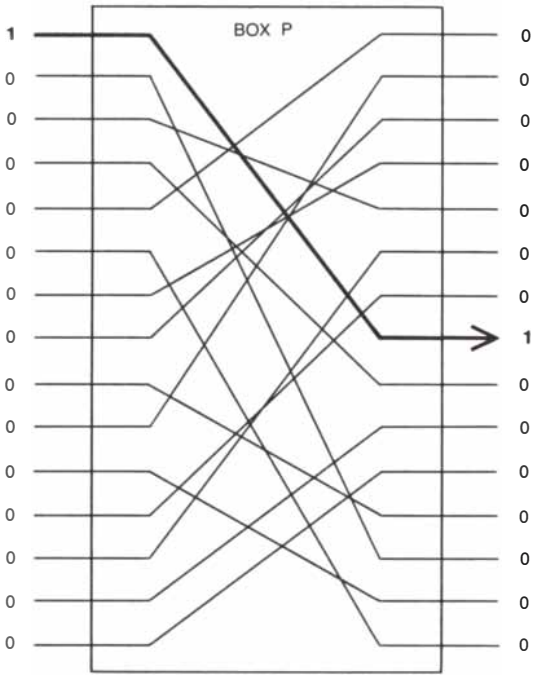
computer environment. Here the data transmitted may often be nonredundant if, for example, they are purely numerical, and an error in a single digit can cause an avalanche of computational errors. Study of the problem has shown that simple error-detecting codes are inadequate for guarding the integrity of computer data against possible tampering by a human expert. What is required is not mere error-detection but cryptographically protected authentication. Surprisingly, this is best achieved by relying on certain principles inherent in the cipher structure itself. Rather than trying to modify the stream concept, let us take a fresh look at the basis of all cryptography: substitution on blocks of message digits.

We shall refer to any cipher that converts n message digits into n cipher digits as a block cipher. For example, a block cipher would be one that turns 00000, standing for a clear-text *A*, into, say, 11001, the cipher equivalent of *A* according to some permutation key, exactly as a tableau does. To see how such a binary transformation is performed by an electronic device let us consider a substitution on only three binary digits [*see illustration on preceding page*].

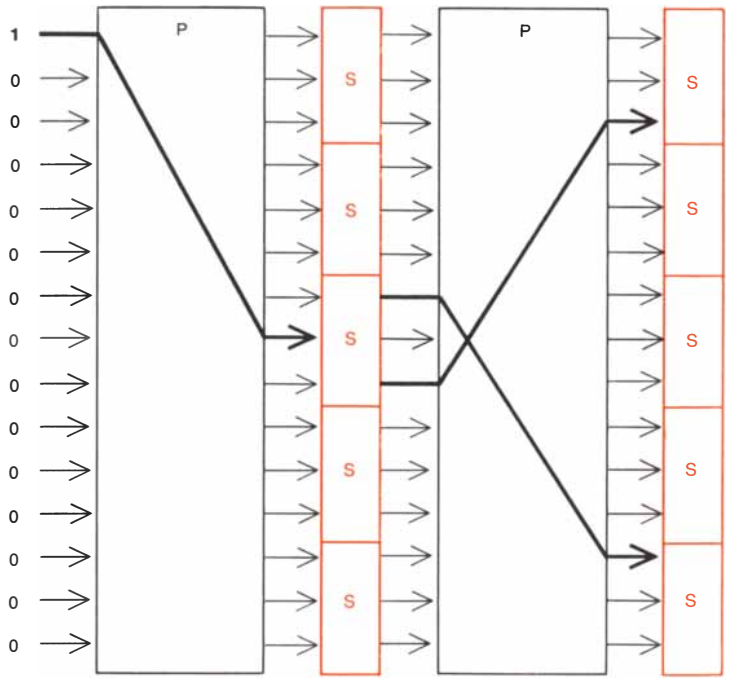
Three binary digits can represent eight items: 2^3 equals eight. The substitution device consists of two switches. The first converts a sequence of three binary digits into its corresponding value to the base eight, thereby energizing any one of eight output lines. These eight lines can be connected to the second switch in any one of 8!, or 40,320, ways. We are at liberty to decide which one of these 40,320 distinct connection patterns, or wire permutations, is to be made between the first switch and the second switch. The role of the second switch is to convert the input, presented as one digit to the base eight, back into a three-digit binary output.

If the substitution device were built to handle a five-digit binary input, it could be used to encipher an alphabet of 32 letters. The number of possible connection patterns between the two switches would then be 32!. That would seem to be an incredibly large number of keys, but the cipher produced must still be regarded as glaringly weak: it could not resist letter-frequency analysis. The weakness is not intrinsic; the device described is mathematically the most general possible. It includes, for any given input-output dimension, any possible reversible cipher that has been or ever could be invented; mathematicians would say it represents the full symmetric group. It is completely "nonsystematic": one permutation connection tells an opponent nothing at all about any other connection. The problem is not intrinsic, then, but is related to size. In spite of the large number of keys, the "catalogue" of possible inputs and outputs is too small: only 32. What is required is a catalogue so large that it is impractical for any opponent to record it. If we had a box with 128 inputs and outputs, for example, an analyst would have to cope with 2^{128} (or more than 10^{38}) possible digit blocks, a number so vast that frequency analysis would no longer be feasible. Unfortunately a substitution device with 128 inputs would also require 2^{128} internal terminals between the first and the second switch, a technological impossibility. This is a fundamental dilemma in cryptography. We know what would be ideal but we cannot achieve the ideal in practice.

Perhaps one could find a device that is easy to realize for a large number of inputs. One might, for example, build a box with, say, 128 input and 128 output terminals that are connected internally by ordinary wire crossings [*see illustration at left below*]. Such a "permutation box" with $n!$ terminals would have $n!$ possible wire crossings, each of which could be set by a different key. It could be built easily for $n = 128$. Although this provides a usefully large number of keys (128!), we are now faced with a new difficulty. By the use of special trick messages it is possible to read out the complete key to such a system in only $n - 1$ (in this case 127) trials. The trick



PERMUTATION BOX can handle very many terminals but it only shuffles positions of digits. An opponent can learn its wiring by feeding in inputs with single 1's and seeing where 1's come out.



PRODUCT-CIPHER SYSTEM combines *P* boxes and *S* boxes. The *P* boxes have a large number of inputs (represented by 15 in the illustration) and the *S* boxes a number that is manageable for such

© 1973 SCIENTIFIC AMERICAN, INC

This content downloaded from
185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms

is to introduce a series of messages containing a single 1 at $n-1$ positions; the position of the 1 in the output betrays the particular wire crossing used in the box. The flaw in the simple permutation box is again that it is a linear system.

We need a compromise that will at least approximate the features of the general system. We are led to the notion of a product cipher in which two or more ciphers are combined in such a way that the resulting system is stronger than either of the component systems alone. Even before World War I various cumbersome ciphers using several stages of encipherment were studied. The first genuinely successful example was probably the one devised by the Germans that was known as the ADFGVX system. We need only observe here that it coupled "fractionation" with "transposition." By that procedure a message was broken into segments and the segments were transposed. The important fact to note here is that the result of a product cipher is again a block cipher; the goal, of course, is that the cipher behave as much as possible as if it were a general substitution cipher.

Between World War I and World War II interest in product ciphers almost totally disappeared because of the successful development of rotor, or wired-wheel, machines, which belong to the general class of pseudorandom-stream generators. A typical rotor machine has a keyboard resembling that of a typewriter. Each letter is enciphered by the operation of several wheels in succession, the wheels being given a new alignment for each new letter according to an irregular and keyed stepping algorithm. The message is decoded by an identical machine with an identical key setting.

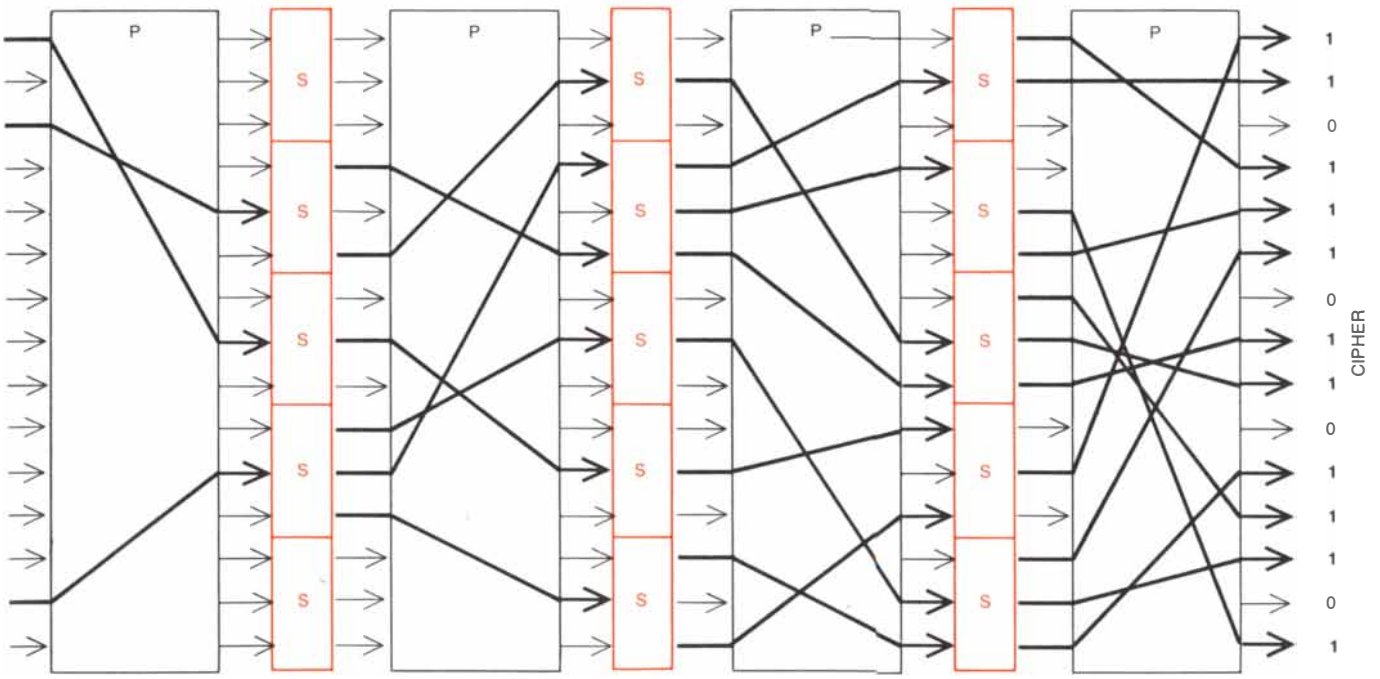
The modern interest in product systems was stimulated by a paper by Claude E. Shannon titled "Communication Theory of Secrecy Systems," published in the *Bell System Technical Journal* in 1949. In a section on practical cipher design Shannon introduced the notion of "mixing transformation," which involved a special way of using products of transformations. In addition to outlining intuitive guides that he believed would lead to strong ciphers, he introduced the concepts of "confusion" and "diffusion." The paper opened up almost unlimited possibilities to invention, design and research.

The manner in which the principles of confusion and diffusion interact to provide cryptographic strength can be described as follows. We have seen that general substitution cannot be realized for large values of n, say $n = 128$, and so we must settle for a substitution scheme of practical size. In the IBM system named Lucifer we have chosen $n = 4$ for the substitution box. Even though 4 may seem to be a small number, it can be quite effective if the substitution key, or wire-crossing pattern, is properly chosen. In Lucifer nonlinear substitution effectively provides the element of confusion.

We have also seen that a linear permutation box is easy to build even for $n = 128$. The number of input and output terminals is simply equal to n. Being a pure digit-shuffler, a device that merely moves digits around without altering the number of 1's in the data, the permutation box is a natural spreader of confusion, that is, it can provide optimal diffusion.

In the Lucifer system the input data pass through alternating layers of boxes that we can label P and S. P stands for permutation boxes in which n is a large number (64 or 128) and S stands for substitution boxes in which n is small (4). Whereas either P boxes alone or S boxes alone would make a weak system, their strength in combination is considerable.

One measure of strength is depicted in a device in which for simplicity the P boxes have $n = 15$ and the S boxes have $n = 3$ [*see illustration on these two pages*]. If we imagine this sandwich of boxes being "tickled" by addressing it with a specially selected input, which might consist of a number made up of



devices—three in this case. The *P* boxes shuffle the digits, providing "diffusion." The *S* boxes provide nonlinear substitution and thus "confusion." In this simplified example the input includes a single 1 and 14 0's. Because the *S* boxes are nonlinear, they can potentially increase the number of 1's; meanwhile the *P* boxes move the 1's around. The result can be an unpredictable avalanche of 1's.

© 1973 SCIENTIFIC AMERICAN, INC

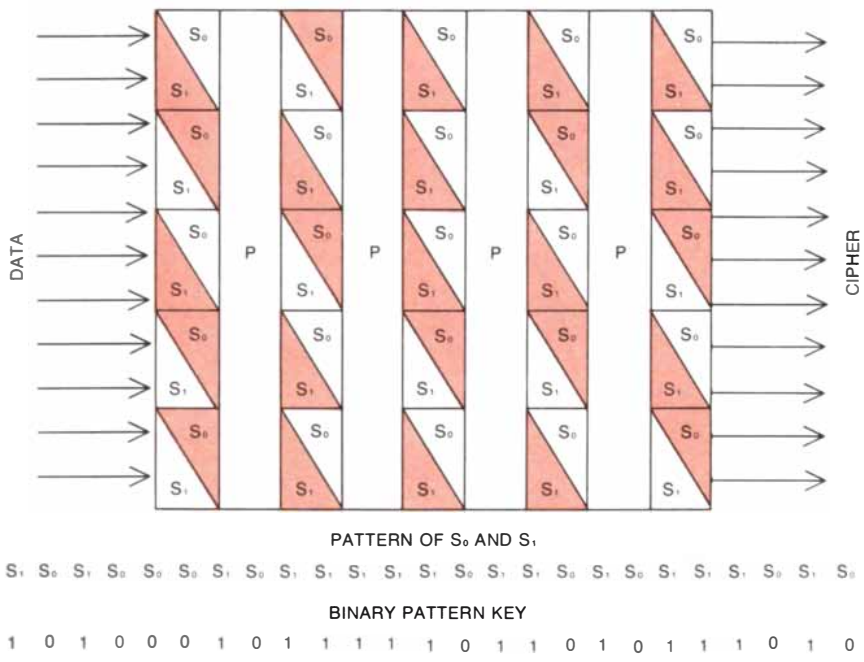
This content downloaded from
185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms

14 0's and only a single 1, it is easy to see confusion and diffusion at work. The first box, *P*, moves the single 1 to some box *S*, which being a nonlinear device may convert the 1 to a three-digit output having as many as three 1's. In the diagram it actually generates two 1's. The next box, *P*, shuffles the two 1's and feeds them into two separate *S* boxes, which together have the potential of generating as many as six 1's. From here on the diagram is self-explanatory. As the input moves through successive layers the pattern of 1's generated is amplified and results in an unpredictable avalanche. In the end the final output will have, on the average, half 0's and half 1's, depending on the permutation keys chosen for the various *P* and *S* boxes.

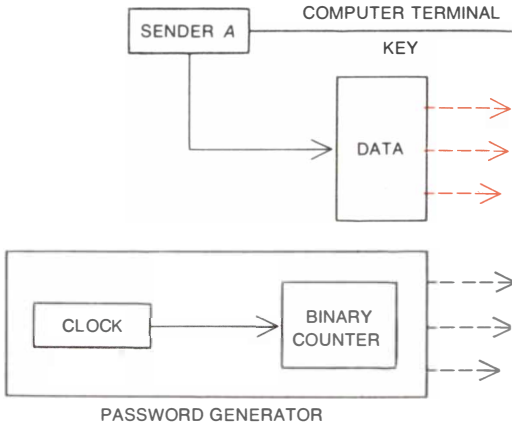
The important fact is that all output digits have potentially become very involved functions of all the input digits. Since all the boxes are independently keyed, the avalanche and its final effect cannot be predicted. The goal of the designer, of course, is to make it very difficult for an opponent to trace the pattern back and thus to reconstruct the permutation keys on *S* and *P*.

In the actual Lucifer system we found it advisable to introduce a few important changes. For example, box *S*, being perfectly general, could be accidentally keyed to behave exactly like a *P* box, in which case the entire system would be no better than a single layer of *P*, which could be exposed by tickling. To avoid this result both *S* and *P* boxes are permanently keyed with permutations that are considered strong; these permanent keys will be known to anyone owning the boxes. Hence we need another keying capability, preferably one that can be represented by a binary number.

This can be accomplished by building a sandwich in which each *S* box has two different permanent keys and can thus be present in two different possible states, S_0 and S_1. The sequence of these states in any one sandwich constitutes a keyed pattern unknown to a potential opponent. The pattern can be represented by a binary key that in effect says which of two substitution tables is to be used, just as in the case of the two-table substitution discussed earlier [*see illustration below*]. (In the diagram there are 25 *S* boxes, and so the key is 25 digits long; in an actual device there would be many more *S* boxes and a correspondingly longer key.) The key digits can be loaded into a key register in the cryptographic device and possibly recorded magnetically in the key card assigned to the authorized user of the system. When two states of the *S* box are used in this way, the resulting cryptogram exhibits a sensitive intersymbol dependence that makes all output digits complicated functions not only of all message digits but also of all the digits in the key. The concept has so far proved to be resistant to penetration by mathematical analysis.



INDIVIDUAL KEYING CAPABILITY is required because the keys built into the *S* and *P* boxes are known to anyone with access to the same system. Individual keying capability can be provided as shown here. At each *S* position in the system there are two possible states, S_1 or S_0. The pattern in which they are intermixed is established by a binary key as shown.



COMPLETE SYSTEM combines a password generator, a cryptographic product system such as the *P* and *S* sandwich in the preceding illustrations and error-correction. The password generator provides a fresh pass-

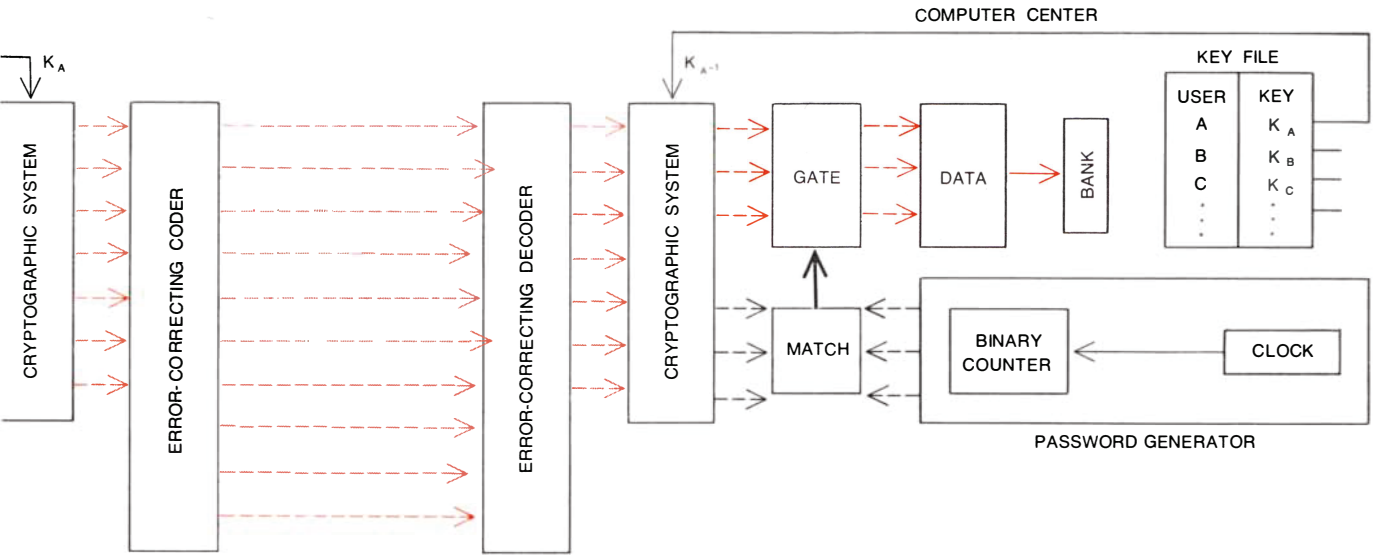
Although the strong intersymbol dependence is a necessary (but not sufficient) index of cryptographic strength, it has a dark side: it implies great sensitivity to noise or interference during transmission. The corruption of a single digit position can, through the inverse-avalanche effect, result in complete garble at the receiver. Modern communications techniques make the problem less serious, however, at least for nonmilitary uses.

Moreover, the strong interdependence among digits provides a surprising and unexpected windfall. Because the system is so sensitive and so violent in its response to changes, it is automatically an ideal agent for detecting such changes, whether generated by accident or by clever intent. And so we have now simultaneously combined high message secrecy with a tamper-proof alarm for error-detection.

To make use of this automatic feature we need only reserve space for a "password" within a given block of message digits. The password is a sequence of digits fed into the message stream automatically by the sending apparatus and does not concern the person using the system. The role of the password is to tell the receiving apparatus that the message has not been intentionally tampered with or seriously degraded by noise during transmission. The enciphering process keeps the opponent from knowing how message bits and password

© 1973 SCIENTIFIC AMERICAN, INC

This content downloaded from
185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms



word block (*black dashes*) for each data block. The sender, using his individual key, introduces his data (*colored dashes*). Password and data digits are no longer traceable as such after being enciphered according to the key. Error-correcting digits are introduced and removed after transmission. The cryptographic system at the computer center deciphers the transmission according to the inverse of the sender's key, which has been selected from a secure file in the center, and extracts the password digits. If these match the password generated in the computer, the gate opens and the input data are admitted to the bank as being of legitimate origin.

bits are reflected in the cryptogram. If the password digits cannot be cleanly recovered by the decoder at the receiving end, the message is rejected.

The decisive role in this scheme is played by the password generator, which is required at both the transmitter and the receiver [*see illustration above*]. The password generator is really nothing more than a binary clock or counter that indicates the time or order number of the message in binary notation and combines a sequence of time digits with each block of data digits transmitted. We assume that at some initial time, say 8:00 A.M., the clocks at the two ends of the transmission channel are synchronized to the same stepping frequency.

Now let us see how the "password authentication scheme" provides privacy for the members of a centralized data-bank community who have access to a large central computer. Each member has his own private key, perhaps in the form of a sequence of binary digits recorded magnetically on a card. The keys of all members are listed in a suitably protected form at the central computer. Let us suppose that a member with Key *A* wishes to send a message to the computer. He inserts the card bearing his key in a typewriterlike terminal sitting on his desk, waits a second or two for a signal that the line is free and begins typing his message.

The message is automatically broken up into blocks of data digits (say 64) and combined at each tick of the binary clock with the password (which might also be 64 digits) that corresponds to the output of the clock at that moment. The resulting block of 128 digits is enciphered by its passing through the sandwich of *S* and *P* boxes that thoroughly mix password digits and data digits.

Since the resulting cryptogram is sensitive to transmission errors, it is fortified with an appropriate error-correcting code, which might be matched to the noise characteristics of the telephone line being used. The addition of the code lengthens the password-plus-message block by a few digits. The resulting cipher block is preceded by the address of the sender in the clear and is transmitted to the central computer. When the message arrives, the key *A* belonging to Member *A* is looked up in the listing and the inverse of Key *A* is fed into the decoder to decipher the cryptogram.

The decisive test now is whether or not the cryptogram as received will yield the same password as the one generated locally by the binary clock at the receiver. In the absence of interference, and if the cryptogram was indeed enciphered with Key *A*, the output of the decoder will consist of the data digit block and the password digit block. This is taken as sufficient proof that the cryptogram did in fact originate with Member *A*. The data are accepted as being of legitimate origin.

What happens if there is interference? If it is nothing more than sporadic bursts of noise on the transmission line, the error-correcting code will eliminate it and the message will pass the authentication test. If, however, the errors are of such a nature that they are not eliminated by the error-correcting code, even one false digit will produce an avalanche effect in the decoder and will garble the output. The passwords will no longer match. The system regards the cryptogram as being of suspicious origin and rejects the communication.

The password test would function just as reliably if someone were to record an intercepted communication and were to retransmit it when the password was no longer valid. The use of a false key, of course, is an instant cause for rejection. The system appears to be proof against any imaginable attempt to deceive it. Each binary digit invested in the password provides one bit of authentication information. If the password is n digits long, an opponent has only one chance in 2^n (or one chance in 2^{64} if $n = 64$) of generating by any means he may select a cryptogram that on decipherment will yield a password that is accidentally correct. The number 2^{64} is equal to about 10^{19}. It is not possible to authenticate more efficiently.

There is no reason why the security system described for a single link could not be expanded to provide security for all users of a network. Other cryptographic approaches are still being studied in our laboratory and elsewhere. It would be surprising if cryptography, the traditional means of ensuring confidentiality in communication, could not provide privacy for a community of data-bank users.

© 1973 SCIENTIFIC AMERICAN, INC

This content downloaded from
185.206.249.34 on Sun, 27 Apr 2025 15:27:10 UTC
All use subject to https://about.jstor.org/terms