Multiuser cryptographic techniques*

by WHITFIELD DIFFIE and MARTIN E. HELLMAN
Stanford University
Stanford, California

ABSTRACT

This paper deals with new problems which arise in the application of cryptography to computer communication systems with large numbers of users. Foremost among these is the key distribution problem. We suggest two techniques for dealing with this problem. The first employs current technology and requires subversion of several separate key distribution nodes to compromise the system's security. Its disadvantage is a high overhead for single message connections. The second technique is still in the conceptual phase, but promises to eliminate completely the need for a secure key distribution channel, by making the sender's keying information public. It is also shown how such a public key cryptosystem would allow the development of an authentication system which generates an unforgeable, message dependent digital signature.

INTRODUCTION

In a computer network with a large number of users, cryptography is often essential for protecting stored or transmitted data. While this application closely resembles the age old use of cryptography to protect military and diplomatic communications, there are several important differences which require new protocols and new types of cryptosystems. This paper addresses the multiuser aspect of computer networks and presents ways to preserve privacy of communication despite the large number of user connections which are possible.

In a system with n users there are n^2-n pairs who may wish to hold private conversations. The straightforward way to achieve this is to give each pair of users a key in common which they share with no one else. Each user will then have n-1 keys, one for communicating with each other user. Unfortunately, the cost of distributing these keys is prohibitive. A new user must send keys to all other users. Unfortunately, the network cannot be used for this purpose, and an external secure channel is required. This procedure is comparable to requiring each new telephone subscriber to send a registered letter to everyone else in the phonebook.

Military communications suffer less from this problem for several reasons. Among these are the limitations imposed by the chain of command and the fact that stations change allegiance infrequently. In a computer network designed for business communication, on the other hand, users will regard each other as friends on one matter and as opponents on another. Firms A and B may cooperate on one venture in competition with C, while simultaneously, A and C compete with B on a different endeavor. A must therefore use different keys for communicating with B and C.

One approach to this problem is to assume that the users trust the network. Each user remembers only one key which is used to communicate with a local node. From there the message is relayed from node to node, each of which decrypts it, then reencrypts it in a different key for the next leg of its journey. This process is known as link encryption.[1] When the message reaches the network node closest to its destination, it is sent on to the addressee encrypted in a key shared only by the addressee and that node.

Although this technique requires each user to remember only one key, it has the disadvantage that a message is compromised if any one of the nodes in its path is subverted. In this paper we examine two other ways of allowing secure communication between any pair of users without assuming the integrity of all nodes in the network and without requiring the users to distribute or store large numbers of keys.

The first technique requires no new technology, but imposes a complex initial connection protocol. This is the subject of the second section of this paper. We call the second technique public key cryptography, since most of the secrecy traditionally required for the keys has been removed. This is discussed in section three and represents a radical departure from past cryptographic practices. While it requires further work before it becomes implementable, its simplicity of operation makes it extremely attractive. If a suc-

* This work was supported by the National Science Foundation under NSF Grant ENG 10173.

cessful implementation can be developed it should find wide use in both military and civilian applications.

The fourth section shows how public key cryptography can be used to provide a time and message dependent digital signature which cannot be forged even when past signatures have been seen. This is an example of the general problem of authentication discussed in greater detail in Reference 2, which provides a more general perspective in which public key cryptography can be viewed.

A PROTFCTIVE PROTOCOL

As indicated earlier, a message protected by link encryption will be compromised if any node in the path it follows from the sender to the receiver is subverted. In this section we describe a protocol which guarantees to protect the message unless a large number of nodes are compromised. While many variations are possible, the basic technique is as follows.

A small number m of the network's nodes will function as "key distribution nodes." Each user has m keys, one for communicating with each of these m nodes. These keys vary from user to user, so while each user must remember only m keys, each of the key distribution nodes remembers n, one for each user of the net. When users A and B wish to establish a secure connection they contact the m key distribution nodes and receive one randomly chosen key from each. These keys are sent in encrypted form using the keys which the users share with the respective nodes. Upon receiving these keys, the conversants each compute the exclusive or of the m keys received to obtain a single key which is then used to secure a private conversation. None of the nodes involved can violate this privacy individually. Only if all m nodes are compromised will the security of this connection fail.

It might be objected that any key distribution node acting alone can prevent all communication by mischievously sending out different keys to each of the parties, thus bringing network operations to a halt. The users, however, can easily protect themselves against this threat. If communication using the composite key fails, its use as a key is abandoned, and the components are exchanged one by one, in clear, for comparison. If any key fails to agree, the node which issued it is blacklisted. Finally, on conclusion of this process, the users repeat the request for keys to the nodes which passed the previous test.

Alternatively, the component keys can be compared by the use of one way functions[2,3,5] without ever being transmitted in clear. Loosely speaking, a function f is called a one-way function if it is easy to compute in the forward direction, but given any output, it is computationally infeasible to find an input which produces it. In referring to a task as computationally infeasible, we have in mind that it cannot be done in fewer than a finite but astronomical number of operations, say 2↑100. For practical purposes, this is equivalent to being incomputable. As shown in Reference 2, a one way function can easily be obtained from a secure cryptosystem.

If communication fails using the composite key, the users send the images of the individual keys under a public one-way function. If the image received does not agree with that computed by applying f to the key, the node which issued it is guilty of compromise. Since the valid keys have not been publicly revealed in this process, there is no need to request new ones from the uncompromised nodes. Instead the invalid ones are omitted and the remainder xored.

To sum up, this technique requires each user to remember m keys and each key distribution node to remember n keys. Unless all m key distribution nodes are subverted, any two users can establish a private link through use of a set-up protocol usually requiring 2m exchanges (more are required if a key distribution node has been subverted). The next section describes a concept which eliminates much of this overhead and does not require the user to trust any node. This new concept, if successfully implemented, will make the technique described above obsolete.

PUBLIC KEY CRYPTOGRAPHY

In this section we propose that it is possible to eliminate most of the secrecy surrounding the key used in a communication, and yet to preserve the secrecy of the communication. This is accomplished by giving each user a pair of keys E and D. E is an enciphering key and is public information. D is the corresponding deciphering key, and while this must be kept secret, it need never be communicated, eliminating the need for a secure key distribution channel. Although D is determined by E, it is infeasible to compute D from E.

For reasons of security, generation of this E-D pair is best done at the user's terminal which is assumed to have some computational power. The user then keeps the deciphering key D secret but makes the enciphering key E public by placing it in a central file along with his name and address. Anyone can then encrypt a message and send it to the user, but only the intended receiver can decipher it. Public key cryptosystems can therefore be regarded as multiple access ciphers.

By regularly checking the file of enciphering keys the user can guard against any attempt to alter it surreptitiously. Any such mischief is reported and settled by other authentication means, such as personal appearance.

The crucial feature of a public key system is that it is relatively easy to generate an E-D pair, preferably automatically through a publicly available transformation from a random bit string to E-D, and yet it is computationally infeasible to compute D from E.

At present we have neither a proof that public key systems exist, nor a demonstration system. We hope to have a demonstration E-D pair in the near future, and expect that if the demonstration pair successfully resists attack then we will be able to design an algorithm for automatically generating E-D pairs of a similar kind. In the meantime, the following reasoning is given to help dispel any doubts the reader may have.

A suggestive example is to let the cryptogram, represented as a binary n-vector c equal E m; where m is the message also represented as a binary n-vector, and E is an arbitrary n-by-n invertible matrix. Letting D=E↑-1 we have m=D c. Thus both enciphering and deciphering are easily accomplished with 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 easily be generated from a random bit string.

Unfortunately, matrix inversion takes only about n↑3 operations even without knowledge of the sequence of elementary operations. The ratio of "cryptanalytic" time (i.e., computing D from E) to enciphering or deciphering time is thus at most n. To obtain ratios of 10↑6 or greater would thus require enormous block sizes. 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 of no consequence in the matrix inversion. In spite of its lack of practical utility, this matrix oriented example is still useful for clarifying the relationships necessary in a public key system.

A more practical direction uses the observation that we are really seeking a pair of easily computed inverse algorithms E and D, but that D must be hard to infer from E. This is not as impossible as it may sound. 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 E (i.e., a listing of E). If the program were to be made purposefully confusing through addition of unneeded variables, statements and outputs, then determining an inverse algorithm could be made very difficult indeed. Of course, E must be complicated enough to prevent its identification from input-output pairs.

Another idea appears more promising. Suppose we start with a schematic of a 100 bit input, 100 bit output circuit which merely is a set of 100 wires implementing the identity mapping. Select 4 points in the circuit at random, break these wires, and insert AND, OR and NOT gates which implement a randomly chosen 4 bit to 4 bit invertible mapping (a 4 bit S box in Feistel's notation).[4] Then repeat this insertion operation approximately 100 times to obtain an enciphering circuit E. Knowing the sequence of operations which led to the final E circuit allows one to easily design an inverse circuit D. If however the gates are now randomly moved around on the schematic of E to hide their associations into S boxes, an opponent would have great difficulty in reconstructing the simple description of E in terms of S boxes, and therefore would have great difficulty in constructing a simple version of D. His task could be further complicated by using reduction techniques (e.g. Carnaugh maps) or expansion techniques (e.g. $\sim(AB)=\sim A$ or $\sim B$, or expressing a logical variable in terms of previous variables), and by adding additional, unneeded S boxes and outputs.

For ease of exposition, we have described the implementation of a specific key in hardware. In practice, a special purpose simulator is obviously of most interest. The hardware description is also valuable in exemplifying a generally useful idea. To build a good public key cryptosystem one needs easily inverted elementary building blocks and a general framework for describing the concatenation of these elementary blocks. Here the elementary building blocks are S boxes and the general framework is the schematic diagram. The general framework must also hide the sequence of elementary building blocks so that no one other than the designer can easily implement the sequence of inverse elementary operations. Examination will show that the matrix example had a similar structure, except there the general class of transformations obtainable was too small.

While the above arguments only provide plausibility as opposed to proof, we hope they will stimulate additional work on this promising area of research.

PUBLIC KEY AUTHENTICATION

The purpose of a cryptographic system is to prevent the unauthorized extraction of information from a public (i.e., insecure) channel. The dual problem of authentication is to prevent unauthorized injection of messages into a public channel.

In conventional paper oriented business transactions, signatures provide a generally accepted level of authentication. As electronic communication replaces mail service the need for a digital signature will be strongly felt.

Various types of authentication are now possible,[2] but the development of public key cryptosystems would allow an entirely new dimension.

Currently, most message authentication consists of appending an authenticator pattern, known only to the transmitter and intended receiver, to each message

and encrypting the combination. This protects against an eavesdropper being able to forge new, properly authenticated messages unless he has also stolen the key being used. There is no protection against such an eavesdropping thief or against the threat of dispute. That is, the transmitter may transmit a properly authenticated message, later deny this action, and falsely blame the receiver for taking unauthorized action. Or, conversely, the receiver may take unauthorized action, forge a message to itself and then falsely blame the transmitter for these actions. For example, a dishonest stockbroker may try to cover up unauthorized buying and selling for personal gain by forging orders from clients. Or a client may disclaim an order, actually authorized by him, but which is later seen to cause a loss. We will introduce concepts which would allow the receiver to easily verify the authenticity of a message, but which prevent him from generating apparently authenticated messages, thereby protecting against both the threat of eavesdropping thieves and the threat of dispute. Note that these techniques thus provide stronger protection than signatures, voiceprints, etc. which can be forged once seen and are not message dependent.

To obtain an unforgeable digital signature from a public key cryptosystem, the protocol would be as follows: Assume user A wishes to send a message M to user B. The transformed message C=EbDa(M) is sent, where Eb represents the transformation effected by use of B's public enciphering key and Da represents the transformation effected by use of A's secret deciphering key. Upon receipt of C, user B operates first with his secret operation Db and then with the public operation Ea thereby obtaining EaDb(C) = EaDbEbDa (M) = M. No one else can extract M because of the need to know Db. By saving the intermediate result Db(C) = Da(M) user B (and only user B) can prove that he received the specific message M from user A. There must be some structure to the message (e.g., it could include a date and time field) to prevent injection of random bit patterns for C, with the hope that the resultant decoded "message", EaDb(C), might cause random mischief such as deletion of files.

Note that since there is no need for a secure channel for distribution of authentication information, we have a public key authentication system. This system protects against, "eavesdropping thieves" and against a dispute as to whether or not an action taken by the receiver was authorized by the transmitter. Similarly, a public key cryptosystem can be used to protect against the other type of dispute in which the transmitter A claims to have issued an order which was not carried out by the receiver B. The transmitter requests that the receiver B send EaDb(M) as a receipt for the message M. By operating on this receipt with his secret operation Da, the transmitter obtains Db (M), which could only have been generated by the receiver B. Only user A can generate this receipt since it requires knowledge of Da.

While the above discussion centered on message authentication it also applies to user authentication. The implicit message becomes "I am user X and the time is T." Inclusion of the time field prevents an eavesdropper from using old authentication signals to pose as someone else. For reasons noted in Reference 2, such a system deserves to be called a one-way IFF system.

We thus see that public key cryptosystems developed for ensuring the privacy of communications, could also be used to ensure their authenticity. They could therefore be used to fill the need for a digital equivalent of a signature. This need is currently a major barrier to the use of electronic mail for business communications, and provides additional motivation for study of public key cryptosystems.

ACKNOWLEDGMENT

The authors wish to thank Leslie Lamport of Massachusetts Computer Associates for several valuable discussions. In particular, the technique described in Section 2 was discovered during one of these conversations.

REFERENCES

1. Baran, Paul, *On Distributed Communications: IX. Security, Secrecy, and Tamper-Free Considerations*, Santa Monica, CA: The Rand Corporation August 1964, (RM-3765-PR).
2. Diffie, Whitfield and Martin E. Hellman, forthcoming paper to be submitted to the *IEEE Transactions on Information Theory.*
3. Evans, Arthur, Jr., William Kantrowitz and Edwin Weiss, "A User Authentication System not Requiring Secrecy in the Computer," *Communications of the ACM*, Vol. 17 No. 8, August 1974, pp. 437-442.
4. Feistel, Horst, "Cryptography and Computer Privacy," *Scientific American*, Vol. 228, No. 5, May 1973, pp. 15-23.
5. Purdy, George B., "A High Security Log-in Procedure," *Communications of the ACM*, Vol. 17, No. 8, August 1974, pp. 442-445.