

KEY EXCHANGE USING 'KEYLESS CRYPTOGRAPHY' *

Bowen ALPERN and Fred B. SCHNEIDER

Department of Computer Science, Cornell University, Ithaca, NY 14853, U.S.A.

Communicated by Ken C. Sevcik

Received 24 August 1982

Revised 1 November 1982

Protocols to generate and distribute secret keys in a computer network are described. They are based on *keyless cryptography*, a new cryptographic technique where information is hidden by only keeping the originator of a message, and not its contents, secret.

Keywords: Cryptography, key distribution, encryption

1. Introduction

In a classical cryptosystem, as opposed to a public-key cryptosystem, communication between users is based on a secret key known to them, but no others. A *key-distribution protocol* is employed to distribute copies of this secret key to the users. One simple way to perform key-distribution makes use of out-of-band communication – a diplomatic courier, for example. More desirable for a computer network, however, is a key-distribution protocol that uses the network itself for all communication. Protocols that do this are described in [2,1]

In this note, we describe key-distribution protocols that are based on *keyless cryptography*, a new cryptographic technique where information is hidden by keeping the *originator* of a message, not the *contents* of the message, secret. Our protocols allow two users to agree on the value of a secret key, while preventing passive wiretappers and other users from determining its value.

2. General approach

For users A and B to agree on a secret key k_{AB} , each first chooses a secret bit string. Let $k_A[1]k_A[2] \cdots k_A[2n]$ be k_A , the bit string chosen by A, and $k_B[1]k_B[2] \cdots k_B[2n]$ be k_B , the bit string chosen by B. Assume that user names are totally ordered so that each can independently compute the same name k_{AB} for the key on which they will agree. A then constructs $2n$ messages, one for each bit in k_A , where each message has the form:

“Concerning k_{AB} : my bit i is $k_A[i]$ ”

and B constructs $2n$ messages, one for each bit in k_B , where each message has the form:

“Concerning k_{AB} : my bit i is $k_B[i]$ ”.

Finally, each of these messages is made available to other users (and any passive wiretappers) in a way that leaves its contents readable, but makes it impossible to determine its origin.

By reading these messages, user A can determine k_B : A knows the messages it constructed, thus the other value for each bit position can be attributed to B. Similarly, B can determine k_A .

* This work is supported in part by NSF Grant MCS-81-03605.
B. Alpern is supported by an IBM Graduate Fellowship.

Now, consider some other user of passive wiretapper U. U can read the contents of all messages. This allows U to deduce values only for those positions of k_A and k_B that are the same. On the average, U will learn n bits in this manner. However, A and B can determine which values U will deduce. So, A can delete the bits known to U from k_A to form k'_A , and B can form k'_B similarly. Then, by convention,

$$k_{AB} = k'_A = \text{complement}(k'_B).$$

If necessary, this procedure can be repeated to obtain a larger key.

3. Implementation

Three ways to implement the key distribution scheme described above are now outlined. In each, passive wiretappers and users are able to read the contents of messages, but are unable to determine their origin.

3.1. Central key distribution facility

Assume the existence of a trusted process S.¹ S maintains a database, called the *blackboard*, which can be read by any user at any time. In addition, each user A is assumed to have a secure channel to S. This channel allows a user A to transmit to S the $2n$ messages it constructs containing the bits of k_A .

Upon receipt of such a message, S posts it on the blackboard. So no user can determine the origin of a message simply by its position on the blackboard, S should post messages in random positions on the blackboard; further, S should wait until receiving a number of messages from both A and B before posting any of them, so no user can determine the origin of a message by observing communications traffic.

By reading the blackboard, A can determine k_B and B can determine k_A . Now consider another user or passive wiretapper U. By assumption, U

will be unable to intercept messages sent to S by A and B. Reading the blackboard allows U to deduce values only for those positions of k_A and k_B that are the same. As before, A and B can determine which values U will deduce, so those bits are deleted when forming k_{AB} .

3.2. Broadcast network

Assume users are linked by a broadcast network – either a ring or an Ethernet – where messages are labeled with a destination address, but not the sender's address. Users A and B each transmit in the clear to a fictitious process called *blackboard* the $2n$ messages they construct for their secret key.

Users monitoring the network can read the contents of a message, but will not be able to determine its origin. As required, A will be able to determine k_B ; B will be able to determine k_A ; other users U will only be able to deduce values for those positions of k_A and k_B that are the same.²

3.3. Special communications channels

Assume users A and B are linked by a communication channel consisting of 2 wires³ named 0 and 1, each characterized by:

- (C1) Both users can write a 1 on either wire.
- (C2) Both users and any wiretapper can sense the state of either wire.
- (C3) If a wire has value 1, then no passive wiretapper can determine which of A and/or B is writing a 1 on that wire.

Such a wire can be approximated by using low-impedance cable and high-impedance receivers – that way, hardly any current will flow when a 1 is written on the wire. The approximation will be even closer if a wiretapper is restricted to a single-point tap only.

² A passive wiretapper might be able to determine the origin of a message by simultaneously monitoring several points on the network. Thus, this implementation may not protect the key from a passive wiretapper; it will protect the key from users monitoring the network.

³ One multiplexed wire would do.

¹ An *authentication server* in the terminology of Needham and Schroeder [2].

Assume A and B have perfectly synchronized clocks.⁴ At the i th clock tick, A writes a 1 on the wire whose name is represented by bit $k_A[i]$ and B writes a 1 on the wire whose name is represented by bit $k_B[i]$. Then, each of them reads both wires and records which ones have a 1 written on them.

As before, each user can determine the sequence of bits sent by the other, but a wiretapper can determine only some of the values sent.

4. Discussion

Although perhaps impractical, keyless cryptography provides a way to hide information. By keeping the origin of a message secret, instead of its content, it is possible for two users to exchange messages and have on the average $\frac{1}{2}$ the bits hidden from others. We have shown how this can be used to allow users to agree on a common key; the technique also can be used to exchange other messages.

In addition to the implementations described above, others are possible. An obvious one uses a satellite repeater. A highly directional antenna (or laser) is used for the up-link, so that there is a secure channel between each user and the satellite; messages received by the satellite are broadcast down to all users, after some random delay.

Unfortunately, all of the keyless cryptography protocols we have been able to devise appear to be vulnerable to sabotage by an active wiretapper. The wiretapper might pretend to be A or B (or both) and send bogus messages. In the implementations described in Sections 3.1 and 3.2, as long as the wiretapper is unable to delete messages sent by users, its existence will be revealed by the presence of more than 2 messages concerning the same bit-position. The wiretapper can nevertheless prevent the key exchange from occurring. In the implementation described in Section 3.3, an active wiretapper cannot be detected with certainty, since it implicitly deletes messages sent by A and B whenever it changes the state of a wire.

It would be misleading to view keyless cryptography as a bona fide cryptosystem. While it can disguise data during transmission, we have been unable to devise a scheme using keyless cryptography to encrypt data for secure storage.

Appendix A. Writing values without synchronized clocks

Assume users A and B have clocks that drift apart at rate ϵ , where $\epsilon \ll 1$. A and B use their clocks to measure intervals of K seconds (apparent time). Each will write its value for k seconds starting from the beginning of an interval and will sample the wires at the midpoint of the interval. In order for each user to read the value sent by the other, the clocks must be reset every $k/(2\epsilon)$ seconds. Whenever there is a transition on the wire, the users resynchronize their clocks. If no transition occurs for nearly $k/(2\epsilon)$ seconds, a transition is forced – the value corresponding to this transition is ignored by both since they know it is only for purposes of clock synchronization.

Acknowledgment

A number of people have been subjected to our early formulations of this work, including: Alan Demers, Steve Gurevitz, Dean Krafft, Kathleen Krafft, Steve Worona and Dave Wright. Their comments and patience are appreciated. Dorothy Denning, David Gries and Leslie Lamport provided very helpful comments on an earlier draft of this paper; John Hopcroft furnished helpful advice. We would also like to thank editor Ken Sevcik and the referees for their prompt and helpful responses.

References

- [1] D.E. Denning and G.M. Sacco, Timestamps in key distribution protocols, *Comm. ACM* 24(8) (1981) 533–536.
- [2] R.M. Needham and M. Schroeder, Using encryption for authentication in large networks of computers, *Comm. ACM* 21(12) (1978) 993–999.

⁴ This assumption is relaxed in Appendix A.