SECURE AUDIO TELECONFERENCE

E. F. Brickell, P. J. Lee, Y. Yacobi

Bell Communications Research
435 South Street
Morristown, N.J. 07960, U.S.A.

ABSTRACT

A number of alternative encryption techniques have been suggested for secure audio teleconferencing implementable on public switched network, in which the centralized facility, called *bridge*, does not hold any secret. The role of the bridge is to synchronously add simultaneous encrypted signals, modulo some known number, and then transmit the result to all the participants. Each terminal has a secret key, with which it can decrypt the above modular sum of encrypted signals to obtain the desired ordinary sum of cleartext signals. Secrecy of the systems is analyzed. Some of which are provably secure, assuming the existence of one way functions, and for the others we have partial cryptanalysis.

We also present a N-party identification and signature systems, based on Fiat and Shamir's single party systems, and another N-party signature system based on discrete-log problem. Our systems have communication complexity $2N$ times that of the basic Fiat-Shamir systems (as compared to a factor of N^2 in the direct application of the basic scheme to all pairs).

KEY WORDS: *secrecy*, *identification*, *signature*, *bridging*

C. Pomerance (Ed.): Advances in Cryptology - CRYPTO '87, LNCS 293, pp. 418-426, 1988.

I. INTRODUCTION

When holding a teleconference, it is common for all participants to be connected to a central facility called a *bridge*. The bridge receives the signals from the participants, operates on these signals in an appropriate way, and then broadcasts the result to the participants. Let N be the number of participants in the teleconference. We will assume that the bridge can receive and process N signals in parallel.

In an audio teleconference the bridge could just add the speech signals from all the participants and broadcast the sum. This is not preferable because this would also add the background noise from all the participants and would require unnecessarily large dynamic range. Instead, the bridge typically selects only up to L active speakers, adjusts the volume, adds their signals, and broadcasts the result. L is typically quite small, only 2 or 3, but we will keep it variable for the remainder of this paper. (Certain audio bridges limit the number of broadcastable speaker to be one. But for interactive teleconferencing, which is of our interest, L is required to be at least 2.)

In this paper, we consider the problem of holding a *secure* audio teleconference. The participants will encrypt their digitized speech in the form of linear and non-adaptive PCM or DPCM. The bridge will operate on the encrypted signals without any decryption and broadcast the result. The participants must be able to decrypt and recover the desired signal.

Since from the incoming encrypted signal the bridge cannot determine the activity, the transmitter of each user will determine its own activity and send encrypted speech to the bridge only when they are actually active. (When not active, the transmitter sends a cleartext "idle" signal to the bridge if circuit switched network is used or does not send anything if packet switched network is used.) However if there are more than L participants speaking at a given time, the bridge must decide which L signals to use.

In the initialization phase of the conference, each participant will probably want to be convinced of the identity of each of the other participants. Then they will establish cryptographic keys that will be known to all of the participants, but must be unknown to any outside observer, including the bridge.

We would also like to have a mutual signature of all participants on the contents of the conference. Signatures might also be needed if the participants wish to sign a mutual document during the conference.

In section 2 we present some secure audio teleconferencing systems. Decryption by the participants of the combined signal results in the sum of clear individual signals. The required increase in message size equals (in bits) the logarithm of the maximum number of simultaneously active speakers allowed. The solutions differ in cryptographic strength and system complexity. All the solutions require a mutual synchronous time base in all the terminals and in the bridge.

Secure audio bridges are related to the idea of privacy homomorphisms [RAD]. Privacy homomorphisms would not require synchronous operation of the terminals. However, in [BY], it is shown that two of the additive privacy homomorphisms mentioned in [RAD] are insecure against a ciphertext only attack, and the other two are insecure against a known plaintext attack.

In section 3 we present a N-party identification system and signature system. These systems are based on Fiat and Shamir's 2-party systems, which are much more efficient than using the RSA system for identification or signature. Our systems have communication complexity about 2N times that of the basic systems (compared with N^2 in the direct application of the basic system to all pairs). These protocols may be parallelized easily, unlike the direct application of the basic Fiat-Shamir scheme to all pairs. For example, the identification protocol, when implemented in parallel, takes just twice the time of the basic 2-party protocol, independent of the number of participants.

II. SECURE AUDIO TELECONFERENCE SYSTEMS

2.1 System Descriptions

The following notations all refer to the message (quantized analog value), cryptogram etc., at instant (sample time) t, therefore we omit the indication t. Let the message space M be the integers $0,1,2,\cdots,B$. The cipher space C will be the integers mod P for some $P > L \cdot B$. Let M_i and C_i denote the message and cryptogram of participant i (at time t). To simplify notation, we assume that participants $1,2,\ldots,Q$ are the Q active (allowed) speakers ($Q \leq L$). f denotes an easily computable function, which is hard to invert. f will produce a random integer *mod* P from a key and a sync word (t may be used as a sync word).

(a) Distinct key / Common Sync Additive System

Encryption of message M_i is $C_i = f_{K_i}(t)+M_i \mod P$. The bridge computes $C_T = \sum_{i=1}^{Q} C_i \mod P$ and broadcasts it to all the participants. Decryption is done by subtracting the corresponding sum $\sum_{i=1}^{Q} f_{K_i}(t) \ mod \ P$, and the result is $M_T = \sum_{i=1}^{Q} M_i \ mod \ P$. The bridge needs to notify the receivers of the user IDs of the Q active speakers. This can be very infrequent (say once every 10 ms). The set of keys of all the participants must be known to each of them.

(b) Common Key / Common Sync Additive System.

For this system, the running keys for all the participants are the same since they use a common key K and common sync. The bridge broadcasts C_T as well as (infrequently) the value of Q. The receiver subtracts $Q \cdot f_K(t)$ from C_T to obtain M_T. This system is not as secure as the previous one as we will show shortly, but it greatly simplifies the hardware of the receiver, reduces the amount of side information (just Q, not IDs), and reduces the initial start-up duration since it needs to distribute only one key.

(c) Common Key / Distinct Sync Additive System

If the transmitter uses its ID as a part of the sync word then the resulting running keys are all different. Hence, under some assumptions on the function $f(\cdot)$, this system is as secure as the distinct key system. The bridge broadcasts C_T as well as the active users IDs (infrequently). The receiver subtracts $\sum_{i=1}^{Q} f_K(i;t)$ from C_T to get M_T. Its hardware is more complex than that of system (b), but in some implementations may be less complex than that of system (a).

(d) Common Key / Common Sync Multiplicative System

In this system $C_i = M_i \cdot f_K(t) \mod P$. We will assume that $f_K(t)$ has an inverse modulo P. The bridge sums the cryptograms mod P, and decryption is done by multiplying the total cryptogram by $f_K(t)^{-1} \ mod \ P$, to result M_T.

The following Lemma holds for all systems.

Lemma 2.1 : $M_T = \sum_{i=1}^{L} M_i$.

Proof : $M_T = \sum_{i=1}^{L} M_i \ \ mod \ P$, and $\sum_{i=1}^{L} M_i \leq L \cdot (B) < P$.

2.2 The Security of the systems

We assume that the function $f_K(t)$ is a pseudo random function chosen from a poly-random collection [GGM]. This means that Knowing the output of f on polynomially many inputs one cannot infer, using polynomially bounded resources, the output for some other input with probability significantly higher than guessing. Such functions exist if one way functions exist (i.e., easily computable functions, which are hard to invert on nonnegligible portion of their target) [GGM]. In real applications we believe that using DES as the function $f_K(t)$ is sufficient, however, to prove the following three theorems we need the assumption that f is a pseudo random function.

Theorem 2.2.1 : All the systems are secure for a single speaker.

Sketch of Proof : Using Shannon theory, they are secure for a single speaker if the key sequence is truly random. If we replace the truly random key sequence with a pseudo random function then the system is secure because a proof of its insecurity will establish an efficient way to distinguish between the pseudo random function and a truly random function [GGM].

Theorem 2.2.2 : Systems (a) and (c) are secure for multiple speakers.

Sketch of Proof : The outputs of the pseudo random functions for systems (a) and (c) are all *distinct* and unpredictable (within polynomial resources) for all the users. Hence, if there is cryptanalysis for these systems with multiple speakers, then there is also cryptanalysis for a single speaker, which contradicts the previous theorem.

The outputs of the pseudo random function for systems (b) and (d) are not distinct. At each time instance all the N generators produce the same output. Therefore we cannot prove a theorem similar to 2.2.2 for these systems. In fact Theorem 2.2.3 states the contrary.

Theorem 2.2.3 : Systems (b) and (d) are partially insecure for multiple speakers.

Sketch of Proof : Recall that $B<P/L$. For a given cryptogram C, the probability for a pseudo random key R to decipher C into a value in M is B/P. Therefore, the probability for a R to decipher Q different cryptograms simultaneously into values in M is $(B/P)^{-Q}$. Since there are P pseudo random keys to try, the number of possible solutions is $P \cdot L^{-Q}$ on the average. In addition, in system (b), if there are more than two speakers, then the cryptograms which correspond to the largest and the smallest M's can be identified. Hence with probability $\approx 3/4$, the most significant bits of the two extreme messages can be cryptanalyzed.

III. IDENTIFICATION AND SIGNATURE IN TELECONFERENCING

3.1 Introduction

Fiat and Shamir [FS] invented a single-party identification scheme and a signature scheme, based on the ideas of *zero knowledge proofs* These schemes are secure if factoring is hard. The implementation is more efficient than RSA.

For an N-party teleconference we present similar systems with communication complexity about $2N$ times that of the basic systems (instead of N^2 in the direct application of the basic systems to all pairs).

These protocols can be implemented with high parallelism, and therefore, if computation time is negligible compared to communication time, the identification process of an N-party conference takes about twice the time of 2-party identification protocol, independent of N. Recall that in our model, we assume that the bridge can receive and process N signals in parallel.

Our protocols do not solve the simultaneity problem, i.e. Some parties may quit the process after receiving the signatures of others, and before delivering theirs. If these signature protocols are used to sign the conference itself (say each second), then simultaneity is not important, because the honest parties will quit the conference, as soon as anybody stops cooperating. However, simultaneity is important for documents signature, hence for this purpose our protocols are inappropriate.

In [KO] Koyama and Ohta propose a N-Party identity based key distribution system. Their system provides identification and key-distribution, while our system provides identification and signature.

In addition, we give an N-party signature protocol, based on discrete-log problem. (N-party discrete-log identification protocols were published by Chaum and Van de Graaf [CH].)

3.2 Detailed Description

Each user u has some unique identifying data I_u associated with him (e.g. name, address, social security number). This data is not secret.

Let h be any cryptographically secure pseudo random function. For a more precise definition see [GGM]. Let k be an integer. Everybody can compute $v_{uj}=h(I_u,j)$, $j=1,2,3,..,k$. Without loss of generality we assume that $v_{u1},...,v_{uk}$ all have Jacobi symbol $+1$ (rename the first k v_{uj}, which have Jacobi symbol $+1$). Everybody can efficiently compute the Jacobi symbol of every number.

In this section all congruences are modulo n, the product of two large primes, which are congruent to 3 modulo 4. (Such n is known as "Blum-integer".) The nice property of such n is that for every a, if a has Jacobi symbol $+1$ modulo n, then exactly one of a or $-a$ is a quadratic residue modulo n. We assume that only the central authority knows the factorization of n. Recall that N is the number of users. For user u, the central authority gives u $s_{u1},.....,s_{uk}$ such that for j=1,...,k, $s_{uj}^2 \equiv \pm v_{uj}^{-1}$. It is the knowledge of the

factorization of n, which enables the central authority to compute square roots modulo n.

We will need to have a one way function g which will map very long messages into a relatively short sequence of bits. Let M be the message to be signed. (M could be the concatenation of all encrypted signals during the last second, or a document.) Let Γ be the list of participants.

3.2.1 The quadratic residue signature protocol

(1) Each participant $1 \leq u \leq N$ picks random $0 \leq r_u \leq n$, computes $x_u \equiv r_u^2$, and sends (x_u) to the bridge,

(2) The bridge computes $X \equiv \prod_{u=1}^{N} x_u$ and broadcasts it.

(3) Each participant u computes $g(M,X,\Gamma) = (e_1,..e_k)$,

(4) Each participant u computes $y_u \equiv r_u \cdot \prod_{j=1}^{k} s_{uj}^{e_j}$, and transmits y_u to the bridge,

(5) The bridge computes $Y \equiv \prod_{u=1}^{N} y_u$, and broadcasts it ,

(6) Each participant computes $Z \equiv Y^2 \prod_{j=1}^{k} V_j^{e_j}$, where $V_j \equiv \prod_{u=1}^{N} v_{uj}$, and $v_{uj}^{-1} \equiv s_{uj}^2$,

(7) Y is a valid signature if and only if $Z \equiv \pm X$, and the list Γ of parties matches the list of parties signing the documents.

[ ]

Remarks:

(1) If nobody cheats the protocol terminates positively.

(2) To show that this signature scheme is secure, we need to show that it is difficult for a set R of users to produce a document that is signed by a set $R \cup S$ of users. We can show that this problem is essentially the same problem faced by a forger in the original FS signature scheme (except for simultaneity).

(3) Communication Complexity: 4N communications of length log n, but the time required is only the time for four communications of length log n.

(4) The list Γ of parties signing the document is used to overcome a problem that arises in mutual signatures that does not occur in single signatures. A participant $N+1$ who did not sign the original document may at some later date wish to have his signature on the document. He could take the string of bits $e_1, \cdots, e_k$ and form $Y_{n+1} = \prod_{j=1}^{k} S_{n+1,j}^{e_j}$. (He is essentially using $r_{n+1}=1$.) Then $Y' = Y_{n+1}$ would be accepted as a valid signature of $N+1$ parties if there was no requirement concerning the list Γ.

(5) k is the security parameter. This protocol requires each user u to have a large number $k \approx 100$ of identifiers $S_{u1}, \ldots, S_{uk}$. This number can be reduced to k/τ by a modification that would also increase the communication complexity by a factor of τ.

3.2.2 The quadratic residue identification protocol

One way to achieve identification is to have all participants use the signature protocol to sign the empty message. The function g is used only as a way of generating the bits $e_1, \ldots, e_k$. Each participant wants to be convinced that even if the bridge and all $N-1$ other participants colluded, there is still some randomness in these bits. There are other ways of generating such a string of bits for an identification protocol. One method is to have each participant send in δ bits and then use the concatenation of all of these bits as the string $\bar{e}$. For this $\delta \approx 20$ would probably be sufficient.

3.2.3 Discrete-log N-party signature protocol

We show that discrete-log problem can be used efficiently for signature in N-party teleconference. We could use discrete log either over finite integer fields, or over finite polynomial fields. Operations in GF(2^{1000}) are more efficient then in GF(P), where P $\approx 2^{500}$, while they have about the same security.

Comparing the polynomial discrete-log signature systems, to the quadratic residue signature system, taking in account the current difference between shift register technology and CPU technology, we conclude that the polynomial version discrete-log signature system is almost an order of magnitude faster than the quadratic residue signature system in computation time. (The above comparison is true for ten users, and security parameter $k=50$. The security parameter determines the error probability $=2^{-k}$.) On the other hand it is twice slower in communication time (because the messages are twice longer).

Let T be a prime power, and, as before, let Γ be the list of parties, and M the message to be signed. $\equiv$ denotes congruence in $GF(T)$. Each user u, $1 \leq u \leq N$ has secrets s_{uj}, $1 \leq j \leq k$. $w_{uj} \equiv \alpha^{-s_{uj}}$ is public. To avoid the use of a "telephone book" for public keys, each party can deliver his claimed public key certified by the trusted center. α is a generator of the multiplicative group of the field.

The protocol

(1) u picks random $r_u \in [0, T)$, computes $a_u \equiv \alpha^{r_u}$, and transmits it to the bridge.

(2) The bridge computes $A \equiv \prod_{u=1}^{N} a_u$, and broadcasts it,

(3) Let $(e_1, \ldots, e_k)$ be the first k bits of α^{σ} in $GF(T)$, where σ is the concatenation (M, A, Γ). u computes $y_u \equiv r_u + \sum_{e_j=1} s_{uj} \mod T-1$, and transmits it to the bridge.

(4) The bridge computes $Y \equiv \sum_{u=1}^{N} y_u \mod T-1$, and broadcasts it.

(5) u computes $W \equiv \prod_{u=1}^{N} \cdot \prod_{e_j=1} w_{uj}$, and then $Z \equiv \alpha^{Y} \cdot W$.

(6) Iff $Z \equiv A$, then o.k.

□

The signature is composed of all the above communications.

REFERENCES:

[BY] E. Brickell and Y. Yacobi, "On Privacy Homomorphisms", **Eurocrypt-87,**

[CH] D. Chaum, J. van de Graaf :" An Improved Protocol for Demonstrating Possession of a Discrete Logarithm and Some Generalizations", **Eurocrypt-87** ,

[FS] A. Fiat and A. Shamir :"How to prove yourself: Practical Solutions to Identification and Signature Problems", to appear in Proc. **Crypto-86.**

[GGM]O. Goldreich, S. Goldwasser and S. Micali:"How to Construct Random Functions", **JACM** , Vol.33, No.4, Oct. 1986, pp.792-807.

[KO] K. Koyama and K. Ohta :"Identity based conference key distribution systems", to appear in **Crypto-87** .

[RAD] Rivest, Adleman and Dertouzos :" On data banks and privacy homomorphisms", in Foundations of secure computation, edited by Demillo et al. **Academic Press** 1978.

[Y] A.C. Yao, "Theory and Applications of Trapdoor Functions", Proc. **Foundations of Computer Science**, 1982, pp.80-91.