Broadcasting Session Keys

Mike Just[*‡] (just@scs.carleton.ca)
Evangelos Kranakis[*§] (kranakis@scs.carleton.ca)
Danny Krizanc[*§] (krizanc@scs.carleton.ca)
Paul Van Oorschot[*†] (paulv@bnr.ca)

February 1994

Abstract

This paper considers true broadcast systems for the secure communication of session keys. These are schemes that provide for parallel rather than serial construction of broadcast messages, while avoiding selective broadcasting. We improve upon a model implemented with Shamir's threshold scheme, and use information theoretic techniques to show that this enhancement is optimal. We then present a general foundation upon which true broadcasting can be realized and illustrate its design with two number theoretic implementations.
1991 AMS Classification: 94A60
CR Categories: D.4.6
Key Words and Phrases: Session key, Secure Broadcasting, True Broadcasting, Broadcast Encryption, Sealed Lock, Secret Sharing.
Carleton University, School of Computer Science: SCS-TR-233
Note: A more recent version of this paper is included in the proceedings of the 2nd Annual ACM Conference on Computer and Communications Security, 1994.

[*]Carleton University, School of Computer Science, Ottawa, ON, Canada. K1S 5B6
[†]Bell Northern Research, P.O. Box 3511, Station C, Ottawa, ON, Canada. K1Y 4H7
[§]Research supported in part by NSERC grant.
[‡]Research supported by NSERC graduate scholarship.

1 Introduction

Consider a system consisting of a set U of users who are registered with a trusted center T. We assume that the registration is done *a priori*. The center wishes to securely exchange a large message M with a set P of these users, known as the *privileged set*. As a practical example, one can think of a local cable company where the users are cable subscribers while P is the set of users who subscribe to additional features offered by the cable company.

There are a number of ways to accomplish this exchange. T can communicate with each member of P over a secure channel. This solution is very impractical due to cost. Therefore, we will concern ourselves only with attempts to communicate over insecure channels.

An obvious solution would be for T to encrypt M separately for each member of P, using a secret key that T shares with each member. Although convenient for members in P, this solution is complex for T. For a large message M, T must perform $|P| = t$ distinct encryptions and distribute $t \log_2 M$ bits, for an integer message M. In addition each message must be addressed to reach the intended receiver. This requires an additional $\log_2 |U|$ bits to be appended to each user's message.

Even though T is sharing M with a subset of users, this solution uses the idea of point-to-point communications, i.e. T sends a different encoding of M to different users in P. A more feasible solution would be to incorporate a point-to-multipoint exchange (see [8]), where one message is *broadcast* from the center to all users, with the intent that only members of P can recover the desired information from it.

A solution using broadcasting would be for T to assign a key for every possible subset P. Each user would be required to store a key for each subset that the user belongs to. To send a message to a privileged set P, T encrypts the message using the key corresponding to the subset P, along with information identifying the set P to allow each user to use the proper key. This solution requires T and each user to store a large number of keys, making it very impractical.

A better way to take advantage of message broadcasting would be for T to initiate an exchange by sharing a common *session key* K with each member of P. This can be done by encrypting and sending K to each user in P individually, using a secret key that T shares with each user. This method is advantageous over encrypting M for each user since K is usually much smaller than M. T need only encrypt M once for P, under K, and

broadcast the result to all users.

To improve on the fact that we are using point-to-point exchanges of K, we can also broadcast K, i.e. send a common message to all users such that only members of P can recover K from it. This paper examines methods of broadcasting session keys, attempting to minimize the storage and computational costs of each member of P, while at the same time allowing T to take advantage of point-to-multipoint transmissions.

1.1 Definitions and Notations

We examine systems where we have a set U of n users $U_1, U_2, \ldots, U_n$, and a trusted center T. T wishes to send a large message to a privileged subset P of users, where $|P| = t$ for some integer $t \leq n$. Every U_i will share a secret s_i with T, where each s_i is an s-bit random string. The secure distribution of s_i can be achieved when the user subscribes with T.

We present a model satisfying the definition of a *true broadcast system*, for broadcasting a session key K. According to [1], this is a scheme "in which the broadcast message contains the same information for each and every listener", yet only members in P can recover K from this message.

A true broadcast system is referred to as a *parallel* construction of the broadcast message since T only performs one encryption of K and each user uses the same broadcast message to recover K. On the other hand, a *serial* construction would involve separate encryptions of K for each user in P, followed by the generation of a broadcast message from the separate encryptions. From the broadcast message, each user would first recover his own encrypted piece and then recover K.

1.2 Previous Work

The concept of *secure broadcasting* was proposed in [5]. Subsequent variations appear in [4, 12]. In this method, associated with each user U_i is a publicized key l_i and a secret key s_i. The sender X of messages could equally be a user or a trusted center. Distribution of a session key assumes the existence of a secure cryptosystem between X and each user in P, hence U_i shares s_i with X. Secure broadcasting uses a serial construction. K is separately encrypted by X with each s_i to produce e_i for $U_i \in P$. A so-called *sealed lock* L is then constructed from these e_i and broadcast to all users.

Using their l_i, a user can recover their e_i and decrypt it with s_i.[1]

One such sealed lock construction proceeds as follows. Suppose that l_i is a prime number, distinct for each U_i. Upon separately encrypting K with each s_i to produce $e_i, \forall i : U_i \in P$, X will use the system of equations $y_i \equiv e_i \pmod{l_i}$ and the Chinese Remainder Theorem to produce a unique sealed lock $0 < L < \prod_{i:U_i \in P} l_i$. Upon receipt of L, each $U_i \in P$ reduces L modulo his l_i to recover e_i. Subsequently, s_i is used to decrypt e_i and recover K.

As described, the term sealed lock is misleading. Since the l_i are made public, anyone can recover a user's piece from L (yet only U_i can subsequently recover K). The purpose of publicizing them is to allow one user to broadcast a single message from which only the privileged group of users can recover K, without having to share an l_i with each of them. The security of the system will depend on the security of the encryption function used.

A major shortcoming of this system arises if it is necessary to update the session key for P. In fact, in [5], one of the intended applications is teleconferencing. Updating the session key requires re-encrypting K for each user, followed by computing a new lock L. We present a method using a parallel construction, which alleviates this problem.

Broadcast encryption was introduced in [6]. Each user shares a number of keys with a trusted center T. The scheme is designed to prevent coalitions of users from conspiring to decrypt the broadcasted message. However, the solution uses *selective broadcasting*. Rather than broadcasting the same message for all users, they require that messages be sent only to specific intended destinations. This is not a characteristic of a true broadcast system. Also, this system requires each user to store a number of keys; the number, and the size of the broadcast message are both dependent upon the size of the anticipated coalition. Our proposed schemes avoid these problems.

Using *secret sharing* to broadcast session keys was introduced by Laih *et. al.* [11], and subsequently by Berkovits [1]. We will examine this model in more detail, later in the paper.

1.3 Summary of Results

Section 2 examines how the concept of a threshold scheme can be applied to broadcasting session keys. We present an implementation from [1] and improve upon it, resulting in an optimal scheme with respect to a lower

[1] Two variations exist. X and U_i can share a common s_i, or U_i can publicize a key e_i and keep a secret key d_i. X would encrypt K with e_i while U_i would use d_i to decrypt (see [5]).

bound on the number of bits transmitted by T. This lower bound corresponds to broadcasting at least t copies of the session key, for a privileged set of size t. This bound is for a more general class of session key broadcasting schemes described in section 3. Section 4 presents a general model for the broadcasting of session keys, and discusses its improvements over previous attempts. Finally, section 5 presents two implementations of the new model. The first is impractical but of theoretical interest. The second achieves the aforementioned lower bound by sacrificing "perfect" security for computational security and practicality.

2 True Broadcasting Using Secret Sharing

Recall that a (t, n) *threshold scheme* implies a method for sharing a secret key K among a finite set U of n users such that a subset P of at least $t \leq n$ users from U can recover K while no group of size less than t can do so (see [2, 13, 15]). A trusted center T randomly selects a key K and uses a *concealer function* to produce n *shares* or *shadows* on input K. The n shares are then secretly distributed to the n users, each user receiving exactly one share. To recover K, the users in the priviliged set P, where $|P| \geq t$, combine their shares and input them to a *revealer function* to produce K. For every set X, where $|X| < t$, if inputting less than t shares to the revealer function will give the users in X no advantage in obtaining K, the threshold scheme is said to be *perfect*.

The major problem with using secret sharing to directly broadcast session keys is that it requires the shares of all members of the set P to be input to the revealer function. In reality, this requires either the physical presence of each user or some way of securely communicating their shares. We proceed to alter the method slightly to allow the broadcast of session keys.

Let each user U_i continue to share a long-term secret s_i with T. T computes a random session key K and will share it with a privileged set P where $|P| = t$. T proceeds to compute the following:

$$\mathcal{B} = g(K, s_{j_1}, s_{j_2}, \ldots, s_{j_t}) \tag{1}$$

for an appropriate share-generating function g, where s_{j_k} is the shared secret of $U_{j_k} \in P$. $\mathcal{B}$ is a set of t shares distinct from the shares input to the function g, and is broadcast to all users.

Notice that we are broadcasting t shares which were formed from the session key K and the shares of each of the members of P. The intent is that a perfect $(t+1, n)$ threshold scheme for sharing the secret K is implied by g, where t shares from $\mathcal{B}$ plus any one share s_{j_k} for $U_{j_k} \in P$ reaches the threshold. Recall that in a perfect threshold scheme, $(t+1)-1$ shares reveal no information about the intended secret.

This general idea was considered by Laih *et. al.* [11], and subsequently by Berkovits [1]. In the following sections, we present one of the schemes from [1] and subsequently show how to decrease the size of the broadcast message and the amount of work required by each user to recover K, while maintaining the "perfect" nature of the threshold scheme.

2.1 Implementation Using Shamir Interpolation

Berkovits [1] uses the threshold scheme from Shamir [13] to allow a trusted center T to broadcast messages to a privileged set P as follows.[2] Each user $U_i \in U$ shares a secret point $(x_i, y_i) \in Z_p^* \times Z_p^*$ with T, where $x_i \neq x_j$ for $i \neq j$. To share a session key with members of P, T first selects a random $K \in Z_p^*$. T then inputs t points $(x_i, y_i), \forall i : U_i \in P$ and $(0, K)$ to g in (1). The function g will first produce a polynomial of degree t,

$$p(x) = K + a_1 x^1 + \cdots + a_t x^t. \tag{2}$$

Note that $t+1$ points are required to uniquely define $p(x)$. $\mathcal{B}$ from (1) will be made up of t points in $Z_p^* \times Z_p^*$ on the polynomial, distinct from any shares of a user $U_i \in P$. Note that this fact requires that $p \geq n \geq 2t+1$.

For this scheme, the size of $\mathcal{B}$ is $2t \log_2 p$ bits. From $\mathcal{B}$ and his own share, each $U_i \in P$ has enough points to reconstruct $p(x)$, and compute $p(0) = K$. A user in possession of only $\mathcal{B}$ does not have enough points to reconstruct $p(x)$ and hence has no advantage in recovering K.

The reconstruction of $p(x)$ by each user is relatively expensive. Given t points, $O(t^2)$ multiplications in Z_p^* are required.

2.1.1 Improving on Implementation

To improve on this technique, we employ an idea used in [10] (in another context). Observe that after constructing $p(x)$, user $U_i \in P$ recovers one of the $t+1$ polynomial coefficients, namely the constant, as the session key K.

[2] We simplify the scheme somewhat for our purposes.

Define the shares created by $\mathcal{B}$ to be the coefficients $a_1, \ldots, a_t$ of $p(x)$. To recover $p(x)$, $U_i \in P$ will create the polynomial in (2) missing only K. Using his share (a point on $p(x)$), U_i can easily solve for the constant of the polynomial, K. This can be done with a simple substitution, followed by an application of Horner's rule with $O(t)$ multiplications. Note the size of $\mathcal{B}$ is now $t \log_2 p$ bits where the session key K is $\log_2 p$ bits.

3 Finding a Lower Bound

We recall some standard information theoretic properties that are used in the upcoming section. For further reference, consult [9, 14].

Given a finite set X of size n and a probability distribution $\{p(x_i)\}_{x_i \in X}$, the *entropy of* X is defined to be

$$H(X) = \sum_{i=1}^{n} p(x_i) \log_b \left(\frac{1}{p(x_i)} \right)$$

where $\log_b(\frac{1}{0})$ is defined to be 0. The subscript b refers to our base of reference. For our purposes, we will be using $b = 2$ and hence computing *bits* of entropy.

Entropy has some useful interpretations. $H(X)$ defines one's average uncertainty in X, i.e. uncertainty about which element of X has been chosen, given the probability distribution. $H(X)$ is also useful for approximating the minimum number of bits required to encode elements of X.

The range of $H(X)$ is $0 \leq H(X) \leq \log |X|$, where the lower bound is obtained when $p(x_i) = 1$ for some i, while $p(x_j) = 0, \forall j \neq i$. The upper bound is achieved when $p(x_i) = \frac{1}{|X|}, \forall i$.

The *equivocation* of X given Y is defined to be

$$H(X|Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i) p(x_i|y_j) \log \left(\frac{1}{p(x_i|y_j)} \right)$$

where Y is a finite set of size m. $H(X|Y)$ can be thought of as the uncertainty in X, given that Y has been observed *a priori*. Note that $H(X|Y) \geq 0$.

The *mutual information* or *transinformation* of X and Y is defined as

$$I(X|Y) = H(X) - H(X|Y)$$

and can be thought of as the amount of information that Y reveals about X. Note that $I(X,Y)$ and $I(X;Y)$ are also equivalent notations. If X and Y are independent, then $H(X|Y) = H(X)$ and $I(X|Y) = 0$. In other words, Y contributes no information about X. Similarly,

$$I(X|YZ) = H(X) - H(X|YZ),$$

for finite sets X, Y and Z. Transinformation has the properties that

$$\begin{aligned} I(X|Y) &= I(Y|X), \\ I(X|Y) &\geq 0. \end{aligned}$$

From the latter statement, note that $H(X) \geq H(X|Y)$.

The *conditional transinformation* of the pair X, Y given Z is defined as

$$I_Z(X|Y) = H(X|Z) - H(X|YZ),$$

and can be interpreted as the amount of information that Y provides about X, given that Z has already been observed. From this we obtain

$$I(X|YZ) = I(X|Y) + I_Y(X|Z).$$

3.1 Properties of Secret Sharing Model

In this section, we take the general idea for broadcasting session keys by secret sharing, as given in section 2, and add several conditions to define a specific model. We then prove a lower bound on the size of the broadcast message required to broadcast a session key and note that the scheme given in section 2.1.1 meets this bound. Let $\mathcal{B}$ and K be the respective broadcast message and session key from section 2 and s_i be the share of any $U_i \in P$. Let $\mathcal{S}_P = (s_{i_1}, \ldots, s_{i_t})$ be the t shares of the users in P. The following are assumptions for the model.

A1 $H(s_i|\mathcal{B}) = H(s_i)$. The broadcast message does not decrease the uncertainty in a user's share.

A2 $H(K|\mathcal{B}) = H(K)$. The broadcast message does not decrease the uncertainty in the session key.

A3 $H(s_i|s_j) = H(s_i), i \neq j$. The share of one user does not decrease the uncertainty in the share of another user, i.e. the shares are independant of one another.

A4 $H(K|\mathcal{B}s_i) = 0$. The session key is uniquely defined by the broadcast message and the share of any member of the privileged set.

A5 $H(K|s_i) = H(K)$. A user's share alone provides no information about the session key.

A6 $H(\mathcal{B}|\mathcal{S}_P K) = 0$. The shares of the users in P, and the session key, define the broadcast message.

We begin by establishing some technical lemmas required for the proof of Theorem 1.

Lemma 1 $H(\mathcal{B}) = I(\mathcal{B}|\mathcal{S}_P K)$.

Proof

$$\begin{aligned} I(\mathcal{B}|\mathcal{S}_P K) &= I(\mathcal{B}|s_{i_1}) + I_{s_{i_1}}(\mathcal{B}|s_{i_2}) + I_{s_{i_1}s_{i_2}}(\mathcal{B}|s_{i_3}) + \cdots + I_{\mathcal{S}_P}(\mathcal{B}|K) \\ &= H(\mathcal{B}) - H(\mathcal{B}|s_{i_1}) + H(\mathcal{B}|s_{i_1}) - H(\mathcal{B}|s_{i_1}s_{i_2}) \\ &\quad + H(\mathcal{B}|s_{i_1}s_{i_2}) - H(\mathcal{B}|s_{i_1}s_{i_2}s_{i_3}) + \cdots + H(\mathcal{B}|\mathcal{S}_P) \\ &\quad - \underbrace{H(\mathcal{B}|\mathcal{S}_P K)}_{0\ by\ A6} \\ &= H(\mathcal{B}) \ \blacksquare \end{aligned}$$

On its own, lemma 2 implies that if the entropy (uncertainty) of a privileged user's share s_i is equal to the entropy of the session key K, then given the broadcast message, any privileged user(s) have no uncertainty in s_i.

Lemma 2 *Let $\mathcal{D} \subset P$ be a non-empty subset of privileged participants such that $|\mathcal{D}| \leq (t-1)$, and let $\mathcal{S}_\mathcal{D}$ be the set of shares of participants in $\mathcal{D}$. Also let s_i be the share of $U_i \in P$ such that $U_i \notin \mathcal{D}$. Given a session key K and broadcast message $\mathcal{B}$ from (1), $H(s_i) - H(s_i|\mathcal{B}\mathcal{S}_\mathcal{D}) \geq H(K)$.*

Proof The term $H(s_i K|\mathcal{B}\mathcal{S}_\mathcal{D})$ simplifies to either $H(s_i|\mathcal{B}\mathcal{S}_\mathcal{D}) + H(K|\mathcal{B}\mathcal{S}_\mathcal{D}s_i)$ or $H(K|\mathcal{B}\mathcal{S}_\mathcal{D}) + H(s_i|\mathcal{B}K\mathcal{S}_\mathcal{D})$ (cf. [3, Lemma 3.3]). Therefore, we can write

$$\begin{aligned} H(s_i|\mathcal{B}\mathcal{S}_\mathcal{D}) + \overbrace{H(K|\mathcal{B}\mathcal{S}_\mathcal{D}s_i)}^{0\ by\ A4} &= \overbrace{H(K|\mathcal{B}\mathcal{S}_\mathcal{D})}^{0\ by\ A4} + H(s_i|\mathcal{B}K\mathcal{S}_\mathcal{D}) \\ H(s_i|\mathcal{B}\mathcal{S}_\mathcal{D}) &= H(s_i|\mathcal{B}K\mathcal{S}_\mathcal{D}) \end{aligned} \tag{3}$$

We also have,

$$\begin{array}{rcl} I_{\mathcal{B}}(s_i|K) & = & I_{\mathcal{B}}(K|s_i) \\ \overbrace{H(s_i|\mathcal{B})}^{H(s_i)\ by\ A1} - H(s_i|\mathcal{B}K) & = & \overbrace{H(K|\mathcal{B})}^{H(K)\ by\ A2} - \overbrace{H(K|\mathcal{B}s_i)}^{0\ by\ A4} \\ H(s_i) - H(s_i|\mathcal{B}K) & = & H(K) \end{array} \tag{4}$$

And,

$$\begin{array}{rcl} I_{\mathcal{B},K}(s_i|\mathcal{S}_{\mathcal{D}}) & = & H(s_i|\mathcal{B}K) - H(s_i|\mathcal{B}K\mathcal{S}_{\mathcal{D}}) \geq 0 \\ & \Rightarrow & H(s_i|\mathcal{B}K) \geq H(s_i|\mathcal{B}K\mathcal{S}_{\mathcal{D}}) = H(s_i|\mathcal{B}\mathcal{S}_{\mathcal{D}}). \end{array} \tag{5}$$

The last equality in (5) is obtained from (3). The result follows by applying (5) to (4). ■

Notice that for the scheme in section 2.1.1, we have that $H(s_i) = 2\log_2 p$ for a randomly chosen point (x_i, y_i) on $p(x)$. Given the broadcast message $\mathcal{B}$ and the point (x_j, y_j) on $p(x)$ for user $U_j \in P$, we have $H(s_i|\mathcal{B}\mathcal{S}_{\mathcal{D}}) = \log_2 p$, where $\mathcal{S}_{\mathcal{D}}$ need only consist of s_j. This results from the fact that user j has the ability to recover $p(x)$ and has thus reduced his uncertainty in user i's share from two dimensions to one. Lemma 2 holds, since the uncertainty in the session key K is only $\log p$ bits.

It is well-known (see [15]) that in a perfect secret sharing scheme, the size of the shares must be at least as large as the secret key. The following corollary highlights the fact that the number of bits used to encode a user's secret must be at least as large as those used to encode the session key, for the specified broadcasting scheme.

Corollary 1 $H(s_i) \geq H(K)$.

Proof From lemma 2, we have

$$H(s_i) \geq H(K) + H(s_i|\mathcal{B}\mathcal{S}_{\mathcal{D}}).$$

Since $H(s_i|\mathcal{B}\mathcal{S}_{\mathcal{D}}) \geq 0$, the result follows. ■

The following theorem will show that for our model, the minimum number of bits required to encode the broadcast message $\mathcal{B}$ is at least as large as t copies of the session key K, where $|P| = t$.

Theorem 1 $H(\mathcal{B}) \geq tH(K)$.

Proof Recall that $\mathcal{S}_P$ is the set of t shares of users in P.

$$\begin{aligned} H(\mathcal{B}) &= I(\mathcal{B}|\mathcal{S}_P K) && (\text{ by Lemma 1 }) \\ &= I(\mathcal{B}|\mathcal{S}_P) + I_{\mathcal{S}_P}(\mathcal{B}|K) \\ &= I(\mathcal{S}_P|\mathcal{B}) + I_{\mathcal{S}_P}(K|\mathcal{B}) \\ &= H(\mathcal{S}_P) - H(\mathcal{S}_P|\mathcal{B}) + \overbrace{H(K|\mathcal{S}_P)}^{H(K)\ by\ A5} - \overbrace{H(K|\mathcal{B}\mathcal{S}_P)}^{0\ by\ A4} \end{aligned} \tag{6}$$

Now, $H(\mathcal{S}_P) = H(s_{i_1}) + H(s_{i_2}) + \cdots + H(s_{i_t})$ (by A3)

$$\begin{aligned} H(\mathcal{S}_P|\mathcal{B}) &= H(s_{i_1} s_{i_2} \ldots s_{i_t}|\mathcal{B}) \\ &= H(s_{i_1}|\mathcal{B}) + \sum_{j=2}^{t} H(s_{i_j}|\mathcal{B} s_{i_1} \cdots s_{i_j - 1}) \end{aligned}$$

So, completing (6), we have

$$\begin{aligned} H(\mathcal{B}) &= \overbrace{H(s_{i_1}) - H(s_{i_1}|\mathcal{B})}^{0\ by\ A1} + \sum_{j=2}^{t} (H(s_{i_j}) - H(s_{i_j}|\mathcal{B} s_{i_1} \cdots s_{i_j - 1})) \\ &\quad + H(K) \\ &\geq (t-1)H(K) + H(K) && (\text{ by Lemma 2 }) \\ &= tH(K) \ \blacksquare \end{aligned}$$

From this theorem, we obtain a lower bound for the size of the broadcast message $\mathcal{B}$ from (1) in section 2. We note that the scheme from section 2.1.1 both satisfies the conditions for this model and realizes the lower bound, and is thus optimal in this regard. Notice that we satisfy A3 for this scheme by defining the share of each user to simply be the y-component. In this way, the shares of each user are independent of one another.

This adjustment will also meet the condition of lemma 2. Consider that since only the secrecy of the y-component is maintained, we would now have that $H(s_i) = \log_2 p$ for a randomly chosen point (x_i, y_i) on $p(x)$. Given the broadcast message $\mathcal{B}$ and the point (x_j, y_j) on $p(x)$ for user $U_j \in P$, we have $H(s_i|\mathcal{B}\mathcal{S}_\mathcal{D}) = 0$, where $\mathcal{S}_\mathcal{D}$ need only consist of s_j. This results from the fact that user j has the ability to recover $p(x)$ and the corresponding y-component of user i's share, since the secrecy of the x-component is not maintained. Since the uncertainty in the session key K is $\log p$ bits, we have $H(s_i) - H(s_i|\mathcal{B}\mathcal{S}_\mathcal{D}) \geq H(K)$, i.e. $\log_2 p - 0 \geq \log_2 p$.

4 General Broadcasting Model

In this section we present a general model for broadcasting a session key K. This model encompasses both the new schemes proposed in sections 5.1 and 5.2 as well as previous schemes (e.g. sections 2 and 3). It presents a foundation for the parallel construction of a broadcast message, such that only members of the privileged set can recover K from this message.

Once the privileged set P has been defined, T generates a broadcast encryption key K_P for that set, where

$$K_P = f(s_{j_1}, s_{j_2}, \ldots, s_{j_t}), \tag{7}$$

for an appropriate function f. T then computes

$$C = E_{K_P}(K), \tag{8}$$

where C is the broadcast message. E is an "encryption function" parameterized by the "key" K_P. f and E are designed such that only knowledge of a single s_{j_k} is required to recover K from C. This will become more meaningful with the schemes presented in sections 5.1 and 5.2.

Notice the relation that (7) and (8) have with (1) of section 2. If f from (7) simply returns its parameters as output, then E from (8) and g from (1) are equivalent. This suggests the use of secret sharing to essentially achieve encryption.

An important difference from previous methods is the ease with which one can update K. By this general approach, one need only recalculate and broadcast (8) to establish a new session key K, whereas [5] and others require separately re-encrypting the new K for each user, followed by recalculation of the sealed lock. In fact, our model could more suitably be called a sealed lock, with master key K_P and equally effective keys s_i for each user $U_i \in P$.

5 True Broadcasting Using General Model

The following sections discuss two implementations of the model presented in section 4. The security of each relies on certain number theoretic assumptions. The first is based on the assumption that it is difficult to determine the quadratic residuosity of an integer. The second is based on the assumption that it is difficult to compute the inverse of an integer, without knowledge of the modulus.

Recall that given a prime number p, q is a *quadratic residue mod p* (denoted $q \in QR_p$) if $\exists x \in Z_p^*$ such that $x^2 \equiv q \pmod{p}$ for $q \in Z_p^*$. If q is not a quadratic residue, then q is a *quadratic non-residue* (denoted $q \in QNR_p$).

Now given an integer $N = p_1 p_2, \ldots, p_t$, where each p_i is a distinct prime,

$$q \in QR_N \Leftrightarrow q \in QR_{p_1} \cap QR_{p_2} \cap \cdots \cap QR_{p_t} \tag{9}$$

whereas

$$q \in QNR_N \Leftrightarrow q \in QNR_{p_1} \cup QNR_{p_2} \cup \cdots \cup QNR_{p_t} \tag{10}$$

Notice that in (9), all of $q \in QR_{p_i}$ must be true, while in (10), only one of $q \in QNR_{p_i}$ need be true.

Given an integer q, one can determine whether or not q is a quadratic residue modulo a prime p, by performing the following test

$$\begin{aligned} q^{\frac{p-1}{2}} &\equiv 1 \pmod{p} \Rightarrow q \in QR_p \\ &\equiv -1 \pmod{p} \Rightarrow q \in QNR_p \end{aligned} \tag{11}$$

We make use of these results in section 5.1.

For the same prime p, one can also compute an *inverse* modulo p. Given an integer $k \in Z_p^*$, there exists a unique inverse k^{-1} for k such that $kk^{-1} \equiv 1 \pmod{p}$. Using N from above, we can obtain a similar result. Given an integer $k \in Z_N^* = \{x | 1 \leq x \leq N-1, gcd(x, N) = 1\}$, there also exists a unique inverse $k^{-1} \in Z_N^*$ for k such that $kk^{-1} \equiv 1 \pmod{N}$. We make use of these results in section 5.2.

5.1 True Broadcasting Using Quadratic Residues

T shares a distinct, independent prime number p_i with each of n users. Let $K = k_0 k_1 \cdots k_{l-1}$ be the binary representation of the session key. T will form the broadcast key K_P in (7) and use it to encrypt the session key K as in (8). C will then be broadcast to all users (including possible eavesdroppers), such that only members of P are able to recover K.

K_P is constructed as follows. First calculate

$$N = p_{j_1} p_{j_2} \cdots p_{j_t} \tag{12}$$

where p_{j_k} is the secret prime that T shares with user $U_{j_k} \in P$. T will produce a y such that $y \in QNR_N$. We require that $y \in QNR_{p_i}$ for each p_i

in (12), which may be done by choosing random $y_i \in Z_p^*$ and using (11) to determine if $y_i \in QNR_{p_i}$. Alternatively, each U_i could share a pre-defined y_i with T. T can then use the Chinese Remainder Theorem with each y_i and p_i to solve for y such that $y \in QNR_N$. Here, $K_P = (y, N)$ serves as the broadcast key.

For the encryption function E in (8), we use a variant of the method of *probabilistic encryption* from [7]. Sending one bit of K will require broadcasting a $\log_2 N$ bit integer. For T to broadcast a bit k_b such that only $U_i \in P$ can recover k_b, T first selects a random $x \in Z_N^*$ and computes $x^2 \bmod N$. Then T computes $C_b = x^2 y^{k_b} \bmod N$ and broadcasts C_b to all users.

Now consider the following two possibilities. If $k_b = 0$ then $C_b \equiv x^2 \pmod N$ and thus $C_b \in QR_N$. From (9), this implies $C_b \in QR_{p_i}$ for each prime p_i. By (11), each U_i who possesses a prime divisor p_i of N can determine that $C_b \in QR_N$ and conclude that $k_b = 0$. If $k_b = 1$, then $C_b \equiv x^2 y \pmod N$ and thus $C_b \in QNR_N$. Since y was chosen such that $y \in QNR_{p_i}$ for each p_i in the factorization of N, again by (11) each privileged user can determine that $C_b \in QNR_N$ and conclude that $k_b = 1$. Users $U_j \notin P$ cannot recover k_b as they lack the appropriate primes p_i.

The size of the broadcast message is $l \log_2 N$ bits, where N is encoded with $t \log_2 p$ bits for some integer $p = max(p_i)$ and an l-bit session key K. This exceeds the lower bound on the size of the broadcast message of section 3 by a factor of l. This clearly is not a practical method of broadcasting. As previously mentioned, this scheme is presented for its theoretical interest.

Once a suitable y has been selected, the amount of work performed by the center to produce C_b is at most 2 modular multiplications in Z_N^*. To broadcast all of K, this process is repeated l times. Each user i using only their prime p_i to recover K (in the manner described above), requires at most $2\lceil \log_2(p_i) \rceil$ modular multiplications in $Z_{p_i}^*$ for each bit of C received (subsequent to the reduction of the $\log_2 N$ bit C with the modulus p_i). Since l bits are broadcast, this operation is repeated l times.

The security of this system is based on the assumption that an opponent can not determine the quadratic residuosity of an integer $q \bmod N$ without knowledge of N's prime factors. Given an integer $q \in Z_N^*$ and N, it is shown in [7] that if determining quadratic residuosity was easy to solve for some q, then it could be solved easily for all q. The fact that N is kept secret, gives an even stronger result.

5.2 True Broadcasting Using Inverses

Once again, T shares a distinct, independent prime number p_i with each of n users. The broadcast key from (7) is the same as (12) above, i.e. $K_P = N$. T now randomly selects K in the range $0 < K < min\{p_{j_1}, p_{j_2}, \ldots, p_{j_t}\}$. The broadcast message C from (8) is the integer $C = K^{-1} \bmod N$. Upon receipt of C, each $U_i \in P$ computes

$$K = C^{-1} \pmod{p_i} \tag{13}$$

$U_i \in P$ can recover K because of the following. Consider that if $KK^{-1} \equiv 1 \pmod{N}$, then by the Chinese Remainder Theorem, $KK^{-1} \equiv 1 \bmod p_i$ for each prime factor p_i of N. Therefore, for each p_i that $K < p_i$ holds true, computation of K from K^{-1} with only p_i is realized. This is done by reducing K^{-1} modulo p_i, and computing its inverse in $Z^*_{p_i}$. User $U_j \notin P$ has little chance in recovering K since he does not possess a prime factor of N.

If the each of the p_i are encoded with $\log_2 p$ bits, for $p = max(p_i)$, then the size of the broadcast message C is $t \log_2 p$ bits. This satisfies the lower bound on the size of the broadcast message of section 3. The amount of work performed by the center to produce C is at most $2t\lceil \log_2(p) \rceil$ modular multiplications in Z^*_N. (This can be done using Euclid's Extended Algorithm). On the other hand, since each user i requires only their prime p_i to recover K (in the manner described above), at most $2\lceil \log_2(p_i) \rceil$ modular multiplications in $Z^*_{p_i}$ are required once C has been reduced modulo p_i.

In this scheme, it may be possible, over time, for privileged users to collect information allowing the deduction of a multiple of another privileged user's secret prime p_i. In order to prevent this from possibly allowing the compromise of another user's secret p_i itself in practice, careful consideration would have to be given to the bit length of the user primes and the modulus N. Indeed, other aspects of the security of this scheme require further study.

6 Conclusion and Open Problems

In this paper, we have provided a foundation upon which true broadcasting can be achieved. In this way, a trusted center can initiate a secure point-to-multipoint communication with a set of privileged users. This allows the center to broadcast only one message, requiring no addressing for the message to reach the intended recipients.

Moreover, each user need only share one key with the center. An important feature is that user privacy (anonymity) is preserved, i.e. the identities

of other users in the privileged set are neither required nor revealed in the broadcast message. This feature is absent in many other schemes, yet is considered a crucial aspect in many practical applications, such as "video-on-demand".

Our results in section 3 are for a model where the shares of the users are chosen independently of one another. It may be possible to achieve a tighter bound if the user's shares are dependent upon one another.

For the model in section 4, we presented two possible implementations. By no means is this an exhaustive list. It is worth further study to find other implementations satisfying the model, possibly characterizing a general set of functions to encompass all possible implementations.

Also consider that this model has other useful applications. As mentioned previously, the model in section 4 creates a master key K_P associated with the privileged set P. The message that is "locked" by this master key can be opened by that same key, and equally by each key of the members of P. Note how this differs from secret sharing where a collection of user's keys are required to open the lock. The model presented here can be applied to many applications requiring such a setup. One such application might be for access control. Consider the locking of information with a master key generated from the keys of members in a privileged set. The information can be unlocked or revealed, only with a key of a member in the privileged set.

References

[1] Berkovits, S., "How to Broadcast a Secret", *Advances in Cryptology: Proceedings of EUROCRYPT '91*, Springer-Verlag, 1992, pp.536-541.

[2] Blakley, G., "One-time pads are Key Safeguarding Schemes, not Cryptosystems: Fast Key Safeguarding Schemes (Threshold Schemes) Exist", *Proceedings of IEEE Symposium on Security and Privacy*, IEEE Computer Society Press, 1980, pp.108-113.

[3] Capocelli, R., De Santis, A., Gargano, L., Vaccaro, U., "On the Size of Shares for Secret Sharing Schemes", *Advances in Cryptology: Proceedings of CRYPTO '91*, Springer-Verlag, 1992, pp.101-113.

[4] Chang, C.C., Hwang, S.J., "A Secure Broadcasting Scheme Based on Discrete Logarithms", *Control and Computers*, Vol.20, No.2, 1992, pp.49-53.

[5] Chiou, G.H., Chen, W.T., "Secure Broadcasting Using the Secure Lock", *IEEE Transactions on Software Engineering*, Vol.15, No.8, August 1989, pp.929-934.

[6] Fiat, A., Naor, M., "Broadcast Encryption", to appear in *Advances in Cryptology: Proceedings of CRYPTO '93*, Springer-Verlag.

[7] Goldwasser, S., Micali, S., "Probabilistic Encryption", *Journal of Computer and System Sciences*, Vol.28, 1984, pp.270-299.

[8] Gopal, J., Jaffe, M., "Point-to-multipoint Communication over Broadcast Links", *IEEE Transactions on Communications*, COM-32(9), 1982, pp.1034-1044.

[9] Jumarie, G., *Relative Information: Theories and Applications*, Springer-Verlag, Berlin, 1990.

[10] Krawczyk, H., "Secret Sharing Made Short", to appear in *Advances in Cryptology: Proceedings of CRYPTO '93*, Springer-Verlag.

[11] Laih, C., Lee, J., Harn, L., "A New Threshold Scheme and its Applications in Designing the Conference Key Distribution Cryptosystem", *Information Processing Letters*, Vol.32, 1989, pp.95-99.

[12] Lin, C.H., Chang, C.C., Lee, R.C., "A Conference Key Broadcasting System Using Sealed Locks", *Information Systems*, Vol.17, No.4, 1992, pp.323-328.

[13] Shamir, A., "How to Share a Secret", *Communications of the ACM*, Vol.22, No.11, November 1979, pp.612-613.

[14] Shannon, C., "Communication Theory of Secrecy Systems", *Bell System Technical Journal*, Vol.28, 1949, pp.656-715.

[15] Simmons, G., "An Introduction to Shared Secret and/or Shared Control Schemes and their Application", *Contemporary Cryptology*, IEEE Press, 1991, pp.441-497.