# Infeasible Group Inversion and Broadcast Encryption

Jim Irrer irrer@eecs.umich.edu
Satyanarayana Lokam satyalv@eecs.umich.edu
Lukasz Opyrchal lukasz@eecs.umich.edu
Atul Prakash atul@eecs.umich.edu

**Abstract**

Broadcast encryption allows a sender to broadcast messages in such a way that only the sender-specified subset of users can decrypt them. Over the years a number of schemes have appeared that solve different variants of the problem. The number of applications that could take advantage from an efficient broadcast encryption scheme has also grown and includes such examples as content-based publish subscribe systems. Such systems are characterized by messages going to arbitrary and rapidly changing subsets of users. Each message can potentially go any of the $2^n - 1$ possible subsets. Collusion resistance and small resource requirements are necessary.

In this paper we present a simple and efficient broadcast encryption scheme based on *one-way group operations*. A group operation $\otimes$ is one-way if $x \otimes y$ is easy to perform for any two group elements $x$ and $y$, but computing the inverse $x^{-1}$ (w.r.t. $\otimes$) is infeasible. Groups with infeasible inversion have recently received attention as a cryptographic primitive and their existence is still uncertain. Under the assumption that such groups exist, we prove the security of our broadcast encryption scheme and explore a few areas where such constructions might come from. Finally, we discuss some functions used in existing cryptosystems and compare them with one-way group operations.

## 1 Introduction

Broadcast encryption schemes provide a convenient way to distribute digital content from a sender to dynamically changing sets of users in such a way that only the *privileged* users can decrypt it. Such schemes are useful in pay-TV systems, distribution of copyrighted material, streaming audio/video, information distribution, etc. As discussed in [OP01], broadcast encryption can also be useful for securing content-based publish subscribe systems where the set of receivers varies as a function of the the message content being distributed [SAB$^+$00, BCM$^+$99, Car98].

Since its introduction by Fiat and Naor [FN93], significant amount of research has been done on different broadcast encryption schemes. The basic problem has many variants. For example, the *privileged* set can be fixed, slowly changing, or rapidly changing. A scheme can support a single, bounded, or unbounded number of broadcasts. The keys stored at each user can be fixed or changing over time. The number of revoked users can be bounded or unbounded. The scheme can be resistant to random or arbitrary adversarial coalitions of various sizes. etc.

We study one of the most interesting (and one of the hardest) variants of broadcast encryption when receivers are *stateless*. Broadcast encryption with *stateless receivers*, as defined in [HS02], has the following requirements:

- Each user is initially given a collection of keys.

- The keys can be used to decrypt any number of broadcasts.
- The privileged set can be defined as any subset of receivers.
- The keys do not change based on user's viewing history.
- The keys do not change as new users join/leave the system (in our case, every new user's public key must be made available to others).
- Consecutive broadcasts can be destined to unrelated privileged sets.
- Each privileged user can decrypt the broadcast by herself.
- Even a coalition of all the non-privileged users cannot decrypt the broadcast.

A number of approaches solving different variants of the broadcast encryption problem are described in section 2.

We introduce and define *one-way group operations* and show how such operations can be used to construct a secure broadcast encryption scheme which satisfies the above requirements. One-way group operations are operations on special types of groups: *groups with infeasible inversion.* If $\otimes$ denotes the operation on such a group $G$, then for any $x, y \in G$, $x \otimes y$ can be computed in polynomial time (in the length of a suitable representation of the elements) whereas no probabilistic polynomial time algorithm can compute $x^{-1}$ with a non-negligible probability. This new cryptographic primitive has also been independently studied in [Hoh03].

Our broadcast encryption scheme is contributory in nature and allows any user to send and receive broadcasts. The scheme supports rapidly changing and arbitrary subsets of receivers. It is also fully collusion resistant. We prove that one-way group operations are sufficient for the security of our scheme. The existence of groups with infeasible inversion is still in question and we examine functions used in existing cryptosystems and show where they fall short.

The remainder of this paper is organized as follows. Section 2 describes in more detail different existing broadcast encryption schemes and their limitations. It also describes alternatives to broadcast encryption. Section 3 defines groups, groups with infeasible inversion and the properties of one-way group operations. Section 4 describes our broadcast encryption scheme and presents proof of the scheme's security. Section 5 discusses the properties of our broadcast encryption scheme and explores possible sources of one-way group operations. Finally, Section 6 presents the current challenge to construct groups with infeasible inversion. It also presents concluding remarks.

## 2 Related Work

Broadcast encryption involves $n$ receivers and a broadcaster. The broadcaster wishes to broadcast messages to a specific subset of $S \subseteq \{1, ..., n\}$ of receivers (for example, receivers that previously paid for the broadcasts). In the idealized scheme, which is the topic of this paper, any receiver in $S$ should be able to decrypt the broadcast. However, *even if all receivers outside of* $S$ *collude* they should not be able to decrypt the broadcast.

Broadcast encryption was first introduced by Fiat and Naor [FN93] in the context of pay-TV. The authors presented methods for securely broadcasting information such that only a selected subset of users can decrypt the information while coalitions of up to $k$ unprivileged users learn nothing. Unfortunately, schemes presented in [FN93] as well as in extensions found in [BC94, BMS98, SvT98] require a large numbers of keys to be stored at the receivers or large broadcast messages. Another problem in the context of secure content-based systems is that some coalitions of more than $k$ unprivileged users can decrypt the information.

Luby and Staddon [LS98] studied the trade-off between the number of keys stored in the receivers and the transmission length in large and small target receiver sets. They prove a lower bound that shows that either the transmission must be very long or a prohibitive number of keys must be stored in the receivers.

An extension proposed in [ASW00] decreases the number of keys required and the length of transmissions by relaxing the target set. It allows a small fraction of users outside the target set to be able to decrypt the information. The authors introduce "f-redundant broadcast encryption schemes" where the total number of broadcast recipients is no more than $f$ times the size of the target set. Such a scheme may work well for pay-TV and similar applications but is unacceptable when confidentiality of broadcast information must be preserved.

More recently, Naor-Naor-Lotspiech [NNL01] gave a construction where each receiver has to store only $O((\log n)^2)$ encryption keys and the header of a broadcast message must contain $O(n-|S|)$ encryptions of a message key ($|S|$ is the size of the receiver set).

Halevi and Shamir [HS02] showed that the number of keys stored at the receiver can be reduced to $O(k * \log n)$. This property holds only for systems designed to broadcast to large sets $S$ (when size of $S$ is close to $n$, so that $n - |S|$ is small. $k$ is the size of an encryption key.

Whether one can build a secure broadcast encryption scheme where both the size of the header and the size of each receiver's key store depend at most logarithmically on $n$, remains an open problem. Fiat and Naor [FN93] and Chick and Tavares [CT89] proposed solutions but they are either not completely collusion resistant [FN93] or they can only handle a small number of receiver sets $S$ [CT89].

In a recent paper, Boneh and Silverberg [BS03] present an efficient solution using $n$-multilinear maps. The authors construct a secure broadcast scheme that requires $O((\log t)^2)$ keys stored at receivers and a constant size header. The multilinear maps are introduced as generalizations of Weil and Tate pairings on supersingular elliptic curves [RS02]. Boneh and Silverberg use broadcast encryption as a motivating application hoping that an efficient way to construct multilinear maps will be discovered. Despite the fact that $n$-multilinear maps provide an efficient and clean solution to the secure broadcast encryption problem, the authors admit that finding examples of $n$-multilinear maps with $n > 2$ may be difficult.

During the same time that we were formalizing our broadcast encryption scheme, Susan Hohenberger was writing her thesis "The Cryptographic Impact of Groups with Infeasible Inversion" [Hoh03]. In the process of searching for a *directed transitive signature* scheme, the author describes *groups with infeasible inversion (GII).* The author formalizes properties of such groups and proves a number of *black-box* reductions between cryptographic primitives (including GII). Independently, using reduction from GII, the author arrives at a construction for two-party key agreement that is similar to our broadcast encryption scheme (for 2-member groups). But there is no generalization in [Hoh03] to key agreement in groups with size larger than two. Our scheme, as presented in this paper, provides construction for $n$-party key agreement as well as a proof of the scheme's security.

An alternative to broadcast encryption was earlier presented in [OP01]. The authors present a number of key caching schemes to enable sending messages to an arbitrary subset $S \subseteq \{1, \ldots, n\}$. The basic idea is to generate and cache keys for most frequent subsets $S_i$. The most promising is the *clustered cache approach*, which divides the set of receivers $n$ into $k$ clusters. Separate keys $K_1, \ldots, K_k$ are maintained for subsets from each cluster and each message must be encrypted up to $k$ times (once for a subset in each cluster). This approach works with relatively small size key stores at receivers and requires a small number of encryptions per message. It is also collusion-proof.

Unfortunately, the approach may be insufficient, when the key store size is very small and only a small (preferably one) number of encryptions per message is allowed. It may also prove inefficient when $n$ is very large.

A related and active area of research is secure group communication [MPH99]. Specifically, group key management services are closely related to the problem described above. Secure group communication systems are usually meant to provide secure channel for the exchange of data between group members. Secure groups are often identified by a session key, known to all group members, which is used to encrypt all data sent to the group. Key management services are used to facilitate member joins and leaves (including expulsions) as well as periodic re-keying to ensure validity of the session key.

The problem of secure delivery of messages from a sender to an arbitrary subset $S \subseteq \{1, \ldots, n\}$ can be cast as group communication with very dynamic membership (the subset $S$ can change from one broadcast to another). If a group communication system were used to broadcast messages from a sender to receivers, a group would have to be reconstructed (possibly entirely) every time the subset $S$ changes. In the worst case the group would have to be reconstructed for every message. The existing group key management approaches were not designed to support dynamic changes that occur in content-based systems do not satisfy our requirements for broadcast encryption algorithms.

A number of group key management schemes attempt to reduce the complexity of reconstructing a group when joins/leaves occur. Most of those systems were designed to with single joins/leaves in mind and with the assumption that membership changes are infrequent in comparison to message sends. Mittra's Iolus system [Mit97] attempts to overcome the problems in scalability of key distribution by introducing locally maintained subgroups. Approaches based on logical key hierarchies (LKH) [WHA99, WGL98, YL00] use a key tree to provide $O(\log N)$ re-keying in response to a single join or leave but scale at least linearly in the presence of random group changes. The VersaKey system [WCS$^{+}$99] extends the LKH algorithm by converting the key hierarchy into a table of keys but is vulnerable to collusion of ejected members.

A number of key agreement protocols (as opposed to key distribution approaches outlined above) have been suggested in which group keys are created from contributions of inputs (or key shares) of desired members [ITW82, SSDW88, BD94, BW98, STW00, KPT00]. The general approach taken in these protocols is to extend the 2-party Diffie-Helman exchange protocol to $N$ parties. Due to the contributory nature and perfect key independence, most of these protocols require exponentiations linear in the number of members [STW00] for most group updates. Kim *et al* unify the notion from key hierarchies and Diffie-Helman key exchange to achieve a cost of $O(\log(N))$ exponentiations for individual joins; however, the number of exponentiations for $k$ joins or leaves will still be usually at least linear in $k$ (unless the nodes that are leaving happen to be all clustered in the same parts of the key tree). Since exponentiation is expensive, these protocols are primarily suitable for establishing small groups at present and not suitable when group membership can change drastically from event to event.

## 3 Groups with Infeasible Inversion

This section defines groups with infeasible inversion and formalizes assumptions for the necessary group operation.

## 3.1 Black Box Groups

Let $\mathbb{G} = \{G_k\}_{k \in \mathbb{N}}$ be an infinite sequence of *commutative* groups. We assume that the elements of $G_k$ are represented by strings in $\{0,1\}^k$. The integer $k$ will be our *security parameter.* We often omit the subscript $k$ in $G_k$. We denote the group operation in $G$ by $\otimes$, and the identity by $\mathbf{1}$ – thus $\otimes$ is a commutative and associative binary operator over $G$ and for every $x \in G$, there is a unique inverse denoted $x^{-1}$ such that $x \otimes x^{-1} = x^{-1} \otimes x = \mathbf{1}$. We assume that $\mathbb{G}$ is presented as a *black box.* This means we assume the following algorithms.

- **Samplability:** There is a polynomial time *sampling algorithm* $S_G(1^k)$ that produces random elements of $G_k$ according to (nearly) uniform distribution.
- **Equality Testing:** Given $x, y \in \{0,1\}^k$, there exists a polynomial time algorithm $\mathrm{EQ}(x,y)$ that returns **true** if and only if $x$ and $y$ represent the same group element in $G_k$. We assume that a special string, say $0^k$, represents the identity element of $G_k$. Thus, we also have an algorithm for *testing if an element is the identity.*

  Note that equality testing is not necessary if we assume a *canonical* representation of every element of $G_k$ as a string in $\{0,1\}^k$. However, we are not assuming so. For example, different matrices might represent the same element in a group of linear transformations.
- **Membership Oracle:** It is sometimes convenient to have an algorithm $\mathrm{Member}(x)$ that, given $x \in \{0,1\}^k$, returns **true** if and only if $x$ represents an element of $G_k$.

In all cases above, we may often allow the algorithms to be randomized.

## 3.2 One-Way Group Operations

The group operation $\otimes$ as defined above is called *one-way* if the following assumptions hold.

- **Assumption 1 (Easiness of $\otimes$):** There is a polynomial time (randomized) algorithm $T$ such that for all $x, y \in G$, $T(x,y) = x \otimes y$.
- **Assumption 2 (Hardness of inversion):** No randomized polynomial time algorithm (PPTM = Probabilistic Poly-time Turing Machine) can compute $x^{-1}$ for a randomly chosen $x \in G$ with non-negligible probability. Formally,

$$\forall \ \text{PPTM} \ A, \ \ \Pr[A(x) = x^{-1} : x \overset{R}{\leftarrow} S_G(1^k)] \leq \nu(k), \tag{1}$$

  where $x \in \{0,1\}^k$ and $\nu(k)$ is a *negligible* function of $k$, i.e., $\nu(k) < 1/k^c$ for any constant $c > 0$. Here the probability is taken over uniformly distribution on $x \in G$ and the internal randomness of $A$.

Sometimes a slight generalization of Assumption 2 is useful.

- **Assumption 2′:** Given randomly chosen $x, y_1, \ldots y_t$, it is infeasible to compute $y_1 \otimes \cdots \otimes y_t \otimes x^{-1}$: $\forall$ PPTM $A'$

$$\Pr[A'(x, y_1, \ldots, y_t) = y_1 \otimes \cdots \otimes y_t \otimes x^{-1} : x \overset{R}{\leftarrow} S_G(1^k), \ y_i \overset{R}{\leftarrow} S_G(1^k)] \leq \nu(k), \tag{2}$$

  where $t = \mathrm{poly}(k)$, and $\nu$ is a negligible function. Again, the probability is taken over random choices $x$ and $y_i$ and the internal randomness of $A'$.

It is easy to show that

**Proposition 1** *Assumption 2 and Assumption 2' are equivalent.*

**Proof:** Suppose Assumption 2 is false. Hence there is a poly-time algorithm $A$ to compute $x^{-1}$ for most $x$. Let $x, y_1, \ldots, y_t$ be inputs in Assumption 2'. Construct $A'$ from $A$ by first computing $x^{-1}$ and then computing $y_1 \otimes \cdots \otimes y_t \otimes x^{-1}$ using the poly-time multiplier $T$ for computing $\otimes$. Time complexity of $A'$ is at most time complexity of $A$ + $t\times$ time complexity of $T$ which is polynomial since $t$ is poly. Thus Assumption 2' is false.

Next, suppose Assumption 2' is false. Note that falsity of Assumption 2 is not immediately obvious by simply letting $y_i = \mathbf{1}$ for $1 \leq i \leq t$. This is because $A'$ could be *wrong* when all $y_i$ are $\mathbf{1}$ and yet have a non-negligible success probability by being correct on most other inputs. Let $x$ be a random element of $G$ (input in Assumption 2). Algorithm $A$ does the following. Generate random elements $y_1, \ldots, y_t \in G$ (using Sampler $S_G(1^k)$). Compute $z := x \otimes y_1 \otimes \cdots \otimes y_t$. Note that $z$ is a uniformly distributed random element of $G$. Feed $z$ and $y_1, \ldots y_t$ as inputs to poly-time algorithm $A'$ that falsifies Assumption 2'. Now, $A'$ outputs $y_1 \otimes \cdots \otimes y_t \otimes z^{-1}$. But $z^{-1} = (x \otimes y_1 \otimes \cdots \otimes y_t)^{-1} = {y_t}^{-1} \otimes \cdots \otimes {y_1}^{-1} \otimes x^{-1}$. Hence,

$$\begin{aligned} A'(z, y_1, \ldots, y_t) &= y_1 \otimes \cdots \otimes y_t \otimes z^{-1} \\ &= y_1 \otimes \cdots \otimes y_t \otimes {y_t}^{-1} \otimes \cdots {y_1}^{-1} \otimes x^{-1} \\ &= x^{-1}. \end{aligned}$$

Hence $A$ outputs whatever $A'$ does. Thus we have a poly-time inverter $A$ from $A'$: time complexity of $A$ is at most time complexity of $A'$ + time to generate the $y_i$ + time to compute $z$. Finally, note that if $A'$ has non-negligible probability of success, so does $A$: $\Pr[A(x) \text{ succeeds}] = \sum_{y \in_R G^t} \Pr[A'(x, y) \text{ succeeds}]$. ■

## 4 Broadcast Key Agreement

Let $U$ be a group (a collection of processors) of users $1, \ldots, n$. Given a *privileged set* $S \subseteq U$ of users, we want a method for each member of $S$ to compute the group key $K_S$ with *no communication* (assuming a global initialization). The key $K_S$ should be infeasible to compute by users outside $S$. This key is then used to broadcast[1] messages meant to be received only by members of $S$.

In our protocol, user $i$ has a secret key $s_i$ and a public key $p_i$. There is a *common string* $c$, $c \neq \mathbf{1}$, known to every user in $U$. All the keys and the common string are elements of the group $G$. The public key $p_i$ is defined by $p_i = s_i \otimes c$. The secret keys $s_i$ and the common string $c$ are picked *uniformly at random* from $G$ using the poly-time sampler. In a global initialization every user is given the public keys of all users in the system (so, we assume each user has sufficient memory to store $n$ public keys).

The group key $K_S$ for $S \subseteq U$ is defined as

$$K_S = \bigotimes_{i \in S} p_i \otimes c^{-1}. \tag{3}$$

By the definition of $p_i$ and commutativity and associativity of $\otimes$, note that for each $i \in S$,

$$K_S = \bigotimes_{j \in S,\, j \neq i} p_j \otimes s_i. \tag{4}$$

[1]In a typical application, there may be a designated *server* or broadcaster. In that case we assume that the server is always a member of any privileged set.

Since the $p_j$'s are known to everyone and $s_i$ is known (only) to user $i$, every user in the set $S$ can compute $K_S$ from (4). No user outside of $S$ can compute $K_S$ *directly* from (4). In fact, we prove that *no coalition of non-members* can efficiently compute the group key:

**Theorem 2** *If Assumption 2 holds, then for every $S \subseteq U$ no coalition of users $T$ such that $T \cap S = \emptyset$ can compute the group key $K_S$ in randomized polynomial time with non-negligible probability when the secret keys $s_i$ and the common value $c$ are chosen uniformly at random from $G$.*

**Proof:** Suppose theorem 2 does not hold. W.l.o.g. let $S = \{1, \dots, m\}$ and $T = \overline{S} = \{m+1, \dots, n\}$. An attack by the coalition $T$ can be viewed as a randomized algorithm that knows the common value $c$, public keys $p_i$ for all users $i \in S$, and (at best) the secret keys $s_j$ of all users $j \in T$ (from these and $c$, $p_j$ for $j \in T$ can be computed). Hence, by our assumption on the attack, there exists a PPTM $B$ such that

$$\Pr[B(c, p_1, \dots, p_m, s_{m+1}, \dots, s_n) = K_S] \geq 1/k^d,$$

for some constant $d > 0$. Using $B$ we construct an algorithm $A'$ that violates Assumption 2'. By Proposition 1, the theorem follows.

Let $x, y_1, \dots, y_n$ be random elements of $G$ input to algorithm $A'$. Algorithm $A'$ constructs an input instance of $B$ as follows.

Let $c := x$, $p_i = y_i$ for $i \in S$, and let $s_j = y_j$ for $j \in T$.

Implicitly, we have $s_i = p_i \otimes c^{-1}$ for $i \in S$. Since $B$ does not know $s_i$ for $i \in S$, this is OK. Note also that the input to $B$ has the right probability distribution. Let $B$ output $z$, supposedly $K_S$. Then $A'$ outputs $z \otimes s_{m+1} \otimes \cdots \otimes s_n$. We have

$$\begin{aligned} A'(x, y_1, \dots, y_n) &= z \otimes s_{m+1} \otimes \cdots \otimes s_n \\ &= K_S \otimes s_{m+1} \otimes \cdots \otimes s_n \\ &= p_1 \otimes \cdots p_m \otimes c^{-1} \otimes s_{m+1} \otimes \cdots \otimes s_n \\ &= p_1 \otimes \cdots p_m \otimes s_{m+1} \otimes \cdots \otimes s_n \otimes c^{-1} \quad \text{by commutativity} \\ &= y_1 \otimes \cdots \otimes y_n \otimes x^{-1}. \end{aligned}$$

Hence $A'$ is correct whenever $B$ recovers the key. It follows that $A'$ has a non-negligible probability of success. Furthermore, $A'$ runs in polynomial time if $B$ does; its additional cost is essentially for multiplying $B$'s output by the $s_j$'s. This violates Assumption 2'. ■

## 5 Discussion

### 5.1 Features and Implications

Given the above outline, an implementation of this key distribution scheme possesses a number of properties:

- The set of members to be included in a group is entirely arbitrary. This supports applications such as content-based publish subscribe systems, where group membership changes frequently.

- The system is immune to failures on the part the recipients, who may be off line or otherwise unavailable. A member wishing to establish a group key may do so independently, and does not require input from recipients other than their initial public key. This is a weakness in systems relying on approaches such as *Group Diffie-Hellman* [STW98, KPT00] that require coordinated input from group members for key generation. In such systems the number of potential points of failure increase as the group size increases, magnifying the effects of the vulnerability.
- The system is immune to collusion. No set of outside or former members may calculate a key to a group to which it does not belong.
- The contributory nature of this scheme has multiple effects. Each member knows that only members that are authorized to join are able to join. Members also know that their private key has been incorporated into the group key, which prevents other members from fabricating keys that are not available to the entire group.
- Keys are not obsoleted. Simple key distribution protocols obsolete keys as new members join, requiring new keys to be generated frequently. This also means that if a member does not have all the public keys needed to decrypt a message, then they can potentially store the encrypted messages until the keys are obtained.
- As a different key is used for each different set of members, perfect forward and backward secrecy are provided.

As stated, each group member publishes its public key, which essentially means that it is broadcast. There is some question as to whether this should be considered as part of the overhead of the system or not. Other key distribution systems consider authentication outside their scope for purposes of evaluation and comparison. Here, authenticating a member requires distribution of its public key, which also builds the required infrastructure. For the purpose of evaluation and comparison, we also choose to consider the publication of public keys outside the scope of key distribution.

The storage requirements for a group of size $n$ is $n \times keysize$. On laptop and larger systems this is easily accommodated. On very small systems such as smart cards this may be a problem, however the problem is somewhat mitigated by the fact that public keys are static, and so can be stored in cheaper read-only memory.

Group keys may be more efficiently computed by storing intermediate results from previous computations. Extremely computationally limited devices may in fact have a set of pre-calculated values downloaded from more computationally able members. Some systems may take advantage of known patterns in membership changes, use algorithms such as “most frequently used”, “most recently used”, or organize keys in a binary tree structure as used in logical key hierarchies [WHA99] which requires $\log(n)$ time for single membership changes. Again, members are independent, so each may choose the speed verses space algorithm which best suits their individual needs.

The scheme predicates that public keys have been distributed. For large centrally managed systems such as publish subscribe, it may be preferable to calculate the expected maximum number of public keys required beforehand, and provide all of them to members as they join the system. A new member is assigned an unused public key and given its corresponding private key. This avoids the incremental method of revisiting existing members to supply them with the public keys of newly joined members.

## 5.2 Examination of Existing Functions

Our examination shows that commonly known cryptographic techniques do not satisfy the assumptions for one-way group operations. We are not aware of any constructions that satisfy the restrictions of *groups with infeasible inversions.* However, similarly to the author of [Hoh03], we postulate the existence of such groups and encourage additional research to find examples of this cryptographic primitive.

This section presents discussion of why some of the more popular cryptographic functions do not satisfy the requirements for the operator in a group with infeasible inversion.

### 5.2.1 Diffie-Hellman and RSA

We considered groups where the Diffie-Hellman problem was easy but the inverse in the exponent was hard. However, discrete logarithms (which support Diffie-Hellman and RSA) lack the associativity property:

$$(a^b)^c \neq a^{(b^c)}$$

where $a$, $b$, and $c$ are positive integers. The lack of associativity violates one of the group criteria as presented in section 3.1. Additionally, Sadeghi and Steiner show that using computational Diffie-Hellman problem to create a group with infeasible inversion would only be possible if the order of the group remains secret [SS01].

### 5.2.2 Elliptic Curve Cryptography

Our original idea is based on the elliptic scalar multiplication operation in elliptic curve cryptography [LD00]. This operation is similar to exponentiation in multiplicative groups in a way that inverses are hard to compute. For an integer $d$, a point on an elliptic curve $P$, and an assignment $F = d \times P$, it is difficult to calculate $d$ given $F$ and $P$.

Briefly, key agreement for groups of size two using elliptic curve cryptography is done in the following way: an equation describing an elliptic curve and point $F$ on that curve is agreed upon by group members and published. Each group member chooses a random number to be used as private key $s$. Their public key $P = s \times F$. Two members can agree on a common key by calculating:

$$K_{12} = F \times s_1 \times s_2$$

$M_1$ calculates:

$$\begin{aligned} K_{12} &= F \times s_1 \times s_2 \\ &= (F \times s_2) \times s_1 \\ &= (P_2) \times s_1 \end{aligned}$$

$M_2$ calculates:

$$\begin{aligned} K_{12} &= F \times s_1 \times s_2 \\ &= (F \times s_1) \times s_2 \\ &= (P_1) \times s_2 \end{aligned}$$

This technique is not applicable to larger groups because the elliptic scalar multiplication operation $\times$ is only valid between an integer value and a point, not between multiple points.

### 5.2.3 Symmetric Key and Hashes

Finally, classic symmetric key and cryptographic hashes also lack the associative property. Using AES [DR99] as a function where $A \otimes B$ indicates that value $B$ should be encrypted with key $A$, the following generally holds true:

$$(A \otimes B) \otimes C \neq A \otimes (B \otimes C)$$

This clearly violates the associativity criteria as presented in section 3.1

## 6 Conclusions and Future Work

The proposed key distribution scheme provides a number of highly desirable features, including most notably arbitrary subgroup addressing and immunity to failure. This scheme is not being presented as the only possible solution with the desired properties, but is attractive because of its simplicity and elegance. Our research to date has not yielded a group with the required properties. Discrete logarithm, elliptic curve cryptography, symmetric encryption, and hashing functions have been shown to be cryptographically hard, but do not satisfy other requirements for *groups with infeasible inversion*. Future efforts will focus on discovering a mathematical group with the given properties, which is pivotal to a successful implementation.

Broadcast encryption is not the only potential application of *groups with infeasible inversion*. Such groups are necessary and sufficient for the implementation of a *directed transitive signature* scheme and are related to several other cryptographic primitives such as Strongly Associative One-Way Functions [Hoh03]. We wish to join the author of [Hoh03] in challenging the cryptographic community to help answer the question regarding the existence of such groups.

## References

[ASW00] Michel Abdalla, Yuval Shavitt, and Avishai Wool. Key management for restricted multicast using broadcast encryption. *IEEE/ACM Transactions on Networking*, 8(4):443–454, 2000.

[BC94] Carlo Blundo and Antonella Cresti. Space requirements for broadcast encryption. In *Advances in Cryptology - EUROCRYPTO'93, LNCS 950*, pages 287–298, 1994.

[BCM+99] Guruduth Banavar, Tushar Chandra, Bodhi Mukherjee, Jay Nagarajarao, Robert E. Strom, and Daniel C. Sturman. An Efficient Multicast Protocol for Content-Based Publish-Subscribe Systems. In *International Conference on Distributed Computing Systems (ICDCS '99)*, June 1999.

[BD94] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In *Advances in Cryptology - EUROCRYPT'94*, 1994.

[BMS98] Carlo Blundo, Luiz A. Frota Mattos, and Douglas R. Stinson. Generalized beimel-chor schemes for broadcast encryption and interactive key distribution. *Theoretical Computer Science*, 200(1-2):313–334, 1998.

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *To appear in Contemporary Mathematics, American Mathematical Society*, 2003.

[BW98] Klaus Becker and Uta Wille. Communication complexity of group key distribution. In *ACM Conference on Computer and Communications Security*, pages 1–6, November 1998.

[Car98] Antonio Carzaniga. *Architectures for an Event Notification Service Scalable to Wide-area Networks.* PhD thesis, Politecnico di Milano, December 1998. Available from http://www.cs.colorado.edu/~carzanig/papers/.

[CT89] G. Chick and S. Tavares. Flexible access control with master keys. In *Advances in Cryptology - CRYPTO'89, LNCS 435*, pages 316–322, 1989.

[DR99] Joan Daemen and Vincent Rijmen. The Rijndael block cypher, AES proposal: Rijndael. www.esat.kuleuven.ac.be/ rijmen/rijndael/, September 1999.

[FN93] Amos Fiat and Moni Naor. Broadcast Encryption. In *Advances in Cryptology - CRYPTO'93*, August 1993.

[Hoh03] Susan Rae Hohenberger. The cryptographic impact of groups with infeasible inversion. Master's thesis, Massachusetts Institute of Technology, 2003.

[HS02] D. Halevi and A. Shamir. The LSD broadcast encryption scheme. In *Advances in Cryptology - CRYPTO'02, LNCS 2442*, pages 47–60, 2002.

[ITW82] I. Ingemarsson, D. Tang, and C. Wong. A conference key distribution system. *IEEE Transactions on Information Theory*, 28(5):714–720, September 1982.

[KPT00] Yongdae Kim, Adrian Perrig, and Gene Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *ACM Conference on Computer and Communications Security*, pages 235–244, 2000.

[LD00] Julio Lopez and Ricardo Dahab. An overview of elliptic curve cryptography, May 2000. http://citeseer.nj.nec.com/333066.html.

[LS98] Michael Luby and Jessica Staddon. Combinatorial bounds for broadcast encryption. In *Advances in Cryptology - EUROCRYPT'98, LNCS 1403*, pages 512–526, 1998.

[Mit97] Suvo Mittra. Iolus: A framework for scalable secure multicasting. In *ACM SIGCOMM*, pages 277–288, September 1997.

[MPH99] Patrick McDaniel, Atul Prakash, and Peter Honeyman. Antigone: A flexible framework for secure group communication. In *Proceedings of the 8th USENIX Security Symposium*, pages 99–114, Washington, August 1999.

[NNL01] Dalit Naor, Moni Naor, and Jeff Lotspiech. Revocation and tracing schemes for stateless receivers. In *Advances in Cryptology - CRYPTO'01, LNCS 2139*, pages 41–62, 2001.

[OP01] L. Opyrchal and A. Prakash. Secure distribution of events in content-based publish subscribe systems. In *Proceedings of the 10th USENIX Security Symposium*, pages 281–295, August 2001.

[RS02] K. Rubin and Alice Silverberg. Supersingular abelian varieties in cryptology. In *Advances in Cryptology - CRYPTO'02, LNCS 2442*, pages 336–353, 2002.

[SAB$^+$00] B. Segall, D. Arnold, J. Boot, M. Henderson, and T. Phelps. Content Based Routing with Elvin4. In *Proceedings of AUUG2K*, Canberra, Australia, June 2000.

[SS01] Ahmad-Reza Sadeghi and Michael Steiner. Assumptions related to discrete logarithms: Why subtleties make a real difference. In *Eurocrypt 2001, Lecture Notes in Computer Science*, volume 2045, pages 244–262, 2001.

[SSDW88] D. Steer, L. Strawczynski, W. Diffie, and M. Wiener. A secure audio teleconference system. In *Advances in Cryptology - CRYPTO'88*, August 1988.

[STW98] Michael Steiner, Gene Tsudik, and Michael Waidner. CLIQUES: A new approach to group key agreement. In *Proceedings of the 18th International Conference on Distributed Computing Systems (ICDCS'98)*, pages 380–387, Amsterdam, May 1998. IEEE Computer Society Press.

[STW00] M. Steiner, G. Tsudik, and M. Waidner. Key agreement in dynamic peer groups. *IEEE Transactions on Parallel and Distributed Systems*, 11(8):769–780, 2000.

[SvT98] D. R. Stinson and T. van Trung. Some new results on key distribution patterns and broadcast encryption. *Designs, Codes and Cryptography*, 3(14):261–279, 1998.

[WCS$^+$99] Marcel Waldvogel, Germano Caronni, Dan Sun, Nathalie Weiler, and Bernhard Plattner. The versakey framework: Versatile group key management. *IEEE Journal on Selected Areas in Communications*, 17(9):1614–1631, September 1999.

[WGL98] C. K. Wong, M. Gouda, and S. S. Lam. Secure group communication using key graphs. In *In Proceedings of ACM SIGCOMM '98*, pages 68–79, September 1998.

[WHA99] Debby Wallner, Eric Harder, and Ryan Agee. Key Management for Multicast: Issues and Architectures. Technical report, IETF, June 1999. RFC 2627.

[YL00] Y. Yang and S. Lam. A secure key management protocol communication lower bound. Technical Report TR2000-24, The University of Texas at Austin, Austin, TX, September 2000.