Title

How to construct multicast cryptosystems provably secure against adaptive chosen ciphertext attack

Permalink

https://escholarship.org/uc/item/8bv5t6f6

Journal

Topics in Cryptology - CT-RSA 2006, Proceedings, 3860

ISSN

0302-9743

Authors

Duan, Yitao
Canny, J

Publication Date

2006

Peer reviewed

How to Construct Multicast Cryptosystems Provably Secure against Adaptive Chosen Ciphertext Attack [⋆]

Yitao Duan and John Canny

Computer Science Division
University of California, Berkeley
Berkeley, CA 94720, USA
{duan, jfc}@cs.berkeley.edu

Abstract. In this paper we present a general framework for constructing efficient multicast cryptosystems with provable security and show that a line of previous work on multicast encryption are all special cases of this general approach. We provide new methods for building such cryptosystems with various levels of security (e.g., IND-CPA, IND-CCA2). The results we obtained enable the construction of a whole class of new multicast schemes with guaranteed security using a broader range of common primitives such as OAEP. Moreover, we show that multicast cryptosystems with high level of security (e.g. IND-CCA2) can be based upon public key cryptosystems with weaker (e.g. CPA) security as long as the decryption can be securely and efficiently "shared". Our constructions feature truly constant-size decryption keys whereas the lengths of both the encryption key and ciphertext are independent of group size.

1 Introduction

Multicast offers an efficient way to deliver the same message to a group of receivers and has become the basis of many applications. The Internet today supports a basic form of multicast service. On the Internet, a multicast group is identified by a Class D IP address and any receivers can join or leave a multicast group by sending IGMP (Internet Group Management Protocol) [1] messages to their local router. Any sender can send message to a multicast group by addressing the message to the group address.

The current IP Multicast service does not provide mechanisms to restrict message delivery to a specified set of receivers therefore other means have to be used to secure the communication. A multicast encryption system provides *confidentiality* for multicast data – ensuring that any parties other than the intended recipients should not be able to access the message. To this end, most of the existing work use one of two approaches. The first is represented by the

[⋆] This work was supported by National Science Foundation award #EIA-0122599 (Title: "ITR/SI: Societal Scale Information Systems: Technologies, Design, and Applications").

work in network research that is concerned with *multicast security*. In this approach symmetric key encryption is used and the data is encrypted with a traffic encryption key (TEK) that is known only to the multicast group members. The difficulty here is key management: The TEK may have to be changed when members join or leave the group. This is known as re-keying. Early schemes (e.g., Group Key Management Protocol (GKMP) [2]) let the group controller or the sender share a pairwise key with each group member and distribute keys to them on a one-to-one basis. For obvious reasons this cannot scale to large groups.

Some work has been done to improve the scalability of such schemes. Among the efficient solutions, the Logical Key Hierarchy (LKH) (or Key Graph) was independently discovered in [3] and [4] and has been an inspiration for many subsequent works [5–10]. In these schemes, individual and auxiliary keys are organized into a hierarchy and each group member is assigned to a leaf and holds all the keys from its leaf to the root. The root key is shared by all group members and used as the TEK. New TEK is distributed by encrypting it with keys that deleted members do not have. So far $O(\log n)$ seems to be the best storage (for both center and members) and communication complexity the LKH-based schemes achieved, where n is the size of the multicast group.

The problem with this approach is that revoking a single user involves changing the keys for all others and the receivers must be stateful and always online in order to receive the latest TEK.

The second approach uses asymmetric key cryptosystem and allows the receivers to be stateless. This includes the work in cryptography such as *traitor tracing*, a concept introduced by Chor, Fiat and Naor [11], and *broadcast encryption*, initiated by Fiat and Naor [12]. Both are based on encryption schemes where a ciphertext can be decrypted by multiple parties with different keys. The scheme in [12] requires $O(t \log t \log n)$ keys per user and the transmission of $O(t^2 \log^2 t \log n)$ messages where t is the number of revoked users. Subsequent work proposed a number of other schemes including [13–17], and [18, 19] which achieved $O(t)$ message complexity and $O(\log^{1+\epsilon} n)$ keys per user. Boneh and Franklin's scheme proposed in [13] is based on Reed-Solomon codes and the representation problem for discrete logs. They also presented a modification, using techniques by Cramer and Shoup [20], that was provably secure against adaptive chosen ciphertext attack.

Recently Boneh et al. presented a broadcast encryption scheme based on bilinear map with constant-size ciphertexts and private keys (and $O(n)$-size public key) [21]. However, in this system, the decryption requires the public key and the knowledge of the set of legitimate recipients. Therefore the "effective" decryption key and/or ciphertext in a real application actually become linear in the total number of receivers.

There is a line of work in the second approach that we classify as Asymmetric Threshold Decryption-based (ATD-based) multicast encryption. This includes [14, 17, 22–24], although none of them explicitly formalized their schemes this way. In these schemes a private key is shared using a $(t + 1, n + t)$-threshold scheme and the shares are distributed *asymmetrically.* Namely the center is given t shares and each user is given 1 share. The center broadcasts a ciphertext together with t partial decryptions. Any member with a valid share of the private key can produce another decryption share and recover the message. With such schemes, user only needs to store a key of constant length. And both the message complexity and sender storage are $O(t)$, independent of the group size.

1.1 Our Results

We focus on the ATD-based multicast encryption cryptosystems and introduce a general framework for constructing such systems with guaranteed security. As we will show later, all existing ones are special cases of our constructions. In particular, they are all based on specific ElGamal encryption that relies on specific assumptions (e.g. DDH). The results we obtained in this paper, on the other hand, are more general. The main contributions are: (1) We show that *any* threshold encryption scheme can be used to construct a multicast cryptosystem that retains the same level of security (e.g. IND-CPA, IND-CCA2) as the underlying threshold encryption. (2) We obtain new results that improve over existing ATD-based schemes in both security and efficiency. Specifically, the resulting scheme from our construction can be made CCA-secure even if the underlying threshold scheme is not. (3) Furthermore, we show that an IND-CCA2 secure multicast scheme can be constructed from a public key cryptosystem that does *not* have a secure threshold implementation (such as OAEP) or has only weaker security (e.g. only IND-CPA), provided the decryption can be securely and efficiently shared (to be elaborated in Sect. 4.4). All of our security proofs are in the same (standard or random oracle) model as the underlying threshold scheme or public key cryptosystem.

These general security results can be used to analyze existing systems in a more unified framework and provide guidelines for constructing future schemes with guaranteed security. This frees the system designer from the burden of security consideration and allows them to focus on other aspects of their schemes.

2 Preliminaries

We consider the scenario where a single party, called *the center*, sends messages, over insecure channels, to a group U of n parties who are denoted *members* of the group. In such a setting, the center often has a special role. Since it is often distributing information of its own choice, it is assumed to have control over the group membership, i.e., the center is allowed to make decisions about who can join the group and whose membership should be revoked. This is in line with almost all multicast schemes such as [25, 4, 26, 27, 22–24].

We assume a computationally bounded adversary who is allowed to attack the system from both outside and *inside* the group. The insider's attack is modelled by allowing the adversary to corrupt and gain total control of up to t group members where t is a predefined threshold. We only consider non-adaptive adversary who chooses what members to corrupt before the key generation.

The multicast communication we are considering in this paper is assumed to be "closed", i.e., we only provide *the center* with the ability to encrypt messages (and of course only the intended recipients can decrypt them). This is different from the public key systems such as [17, 22–24] where the information to encrypt a message is public. The openness is unnecessary for some applications and unacceptable for some others (e.g. military communication). By "closing" the communication, we can provide more flexible constructions that can make use of a broader range of primitives. The price for this flexibility is the loss of the public key feature, which should not be a problem for many applications. However we observe that in many instantiations of our constructions, it is easy to "publicize" the encryption key, without affecting the security of the scheme, as demonstrated by works such as [22–24]. This effectively turns the scheme into a public key system and all the openness features are reinstalled.

3 Multicast Cryptosystem

Definition 1. *An n-way multicast encryption scheme* $\mathcal{ME} = (\mathsf{KeyGen}, \mathsf{Reg}, \mathsf{E}, \mathsf{D})$ *consists of the following set of algorithms:*

1. *Key Generation* KeyGen*: a probabilistic polynomial-time (in k) algorithm which takes as inputs a security parameter 1^k, a threshold t, the number of (initial) group members n, and generates global information I, the encryption key Σ and the master secret key Γ.*
2. *Registration algorithm* Reg*: a probabilistic algorithm to compute the secret initialization data for a new user subscribing to the system.* Reg *receives as input the master key Γ and a new index i associated with the user; it returns the user's secret key Γ_i.*
3. *Encryption* E*: a probabilistic polynomial-time algorithm that, on inputs Σ, the encryption key, and a string $m \in \{0,1\}^k$, and a set R of revoked users (with $|R| \leq t$) and their keys, produces as output $\psi \in \{0,1\}^*$ called the ciphertext* [1]*.*
4. *Decryption* D*: a deterministic polynomial-time algorithm such that $\forall m \in \{0,1\}^k$, $\forall i \in U \setminus R$, $\mathsf{D}(\Gamma_i, \mathsf{E}(\Sigma, \{(j, \Gamma_j) | j \in R\}, m)) = m$. On all other inputs it outputs a special symbol $\perp$.*

KeyGen and Reg should be run by the center and the two can also be executed together with an initial set of n members as input. Admitting new members is relatively trivial, at least for all the construction we will be presenting, so in the following we simply omit Reg and use $(I, \Sigma, \Gamma, \boldsymbol{\Gamma}) \leftarrow \mathsf{KeyGen}(1^k, t, n)$ to denote this process, where $\boldsymbol{\Gamma} = (\Gamma_1, \ldots, \Gamma_n)$ is a vector of secret keys for the n members.

[1] Note that member revocation is implicitly embedded in the encryption algorithm.

3.1 Notion of Security

The communication paradigm we are considering shares similarities with both symmetric key and public key cryptosystems. On one hand the communication is "closed" in that we only allow the center to send messages to the group. On the other hand the keys are "asymmetric" since now there are multiple recipients and our definition includes member revocation which means the encryption key and the decryption keys must be different.

Dodis and Fazio [23] first precisely formalized the notion of adaptive security for public key multicast encryption schemes, which allow anyone having access to the public key to send messages to the group, at both CPA and CCA2 levels. Since our setting is different from the "public key" paradigm, we adopt a slightly modified definition. The major difference is that, we do not *explicitly* allow the adversary to see the sender's keys since ours is not a public key cryptosystem. Instead the adversary can obtain encryptions of arbitrary messages by querying an *encryption oracle* who also encrypts the target message later. This is similar to the security definition based on indistinguishability for symmetric key cryptosystems. The ability to handle member revocation is modelled by allowing the adversary to corrupt members and obtain their secret keys. This formalization is general and captures the security notions of many multicast schemes such as those LKH schemes [3, 4] which are based on symmetric key cryptography. However we note that in all the construction we introduce later, the secret keys of the revoked members constitute the actual *encryption key.* In essence in our constructions the exposure of encryption key can be modelled as corrupting members. This effectively turns our scheme into a "public key" paradigm from the adversary's point of view and the security definitions from [23] are appropriate.

Formal Model Given a multicast encryption scheme $\mathcal{ME} = (\mathsf{KeyGen}, \mathsf{E}, \mathsf{D})$, a polynomial time adversary $\mathcal{A}$'s attack is modelled by the following game:

Game ME:

M1 The adversary $\mathcal{A}$ chooses to corrupt a fixed set R of t members.

M2 $(I, \Sigma, \Gamma, \boldsymbol{\Gamma}) \leftarrow \mathsf{KeyGen}(1^k, t, n)$ is run and $\mathcal{A}$ is given the public information I and the secret keys of corrupted members. User i receives Γ_i. The center is given R and their keys.

M3 The adversary interacts with the center, who acts as the encryption oracle, in an arbitrary fashion. On any query m from $\mathcal{A}$, the center returns its encryption.

M4 $\mathcal{A}$ chooses two plaintexts m_0 and m_1 of the same length and gives them to the center who chooses $b \in \{0, 1\}$ at random, and gives the "target" ciphertext $\psi' = \mathsf{E}(\Sigma, \{(j, \Gamma_j) | j \in R\}, m_b)$ to $\mathcal{A}$.

M5 $\mathcal{A}$ continues to interact with the center.

M6 At the end of the game, $\mathcal{A}$ outputs $b' \in \{0, 1\}$.

The advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{CPA}_{\mathcal{ME},\mathcal{A}}(k) = |Pr(b' = b) - 1/2|$$

In addition, in the case of a adaptive chosen ciphertext attack (CCA2) [2], in both stages M3 and M5, $\mathcal{A}$ is also allowed to interact in an arbitrary manner with the group members who act as the decryption oracles. On a query ψ from $\mathcal{A}$, member i returns $\mathsf{D}(\Gamma_i, \psi)$. The only restriction on the interaction is that the target ciphertext ψ' cannot be one of the queries made to any of the decryption oracles. As before, $\mathcal{A}$'s advantage in the CCA2 case is defined as

$$\mathsf{Adv}^{CCA2}_{\mathcal{ME},\mathcal{A}}(k) = |Pr(b' = b) - 1/2|$$

Definition 2 (t-Resilient Multicast Encryption Scheme). *Let $\mu \in \{CPA, CCA2\}$. A multicast encryption scheme $\mathcal{ME}$ is t-resilient against a μ-type attack if the advantage, $\mathsf{Adv}^{\mu}_{\mathcal{ME},\mathcal{A}}(k)$, of any probabilistic polynomial time adversary $\mathcal{A}$ is a negligible function of k.*

4 ATD-Based Multicast Encryption

In this section we define two constructions and show that a line of previous work on multicast or broadcast encryption can actually be characterized as special cases of these constructions.

4.1 Threshold Decryption Scheme

A $(t+1, n)$-threshold cryptosystem $\mathcal{TD} = (\mathsf{KeyGen_{TD}}, \mathsf{D_{TD}}, \mathsf{V_{TD}}, \eta, \mathsf{E_{TD}})$ consists of the following algorithms:

- Key generation algorithm $(PK, VK, \boldsymbol{SK}) \leftarrow \mathsf{KeyGen_T}(1^k, t, n)$: a probabilistic algorithm that, given a security parameter 1^k, a threshold t, and the number of players n, generates a public key, PK, a verification key VK, and n private keys $\boldsymbol{SK} = (SK_1, \ldots, SK_n)$. PK and VK are made public while SK_i is known only to player i, $i = 1, 2, \ldots, n$.
- Share computation $\mathsf{D_{TD}}$: a probabilistic algorithm that, given a private key SK_i and the ciphertext c, $\mathsf{D_{TD}}$ computes $\rho = \mathsf{D_{TD}}(SK_i, c)$, called a *decryption share.*
- Share verification V: a deterministic algorithm that takes as input the public verification key VK, the ciphertext c, and a share ρ, and outputs $\mathsf{V}(VK, c, \rho) \in \{0, 1\}$.

[2] We do not explicitly consider non-adaptive chosen ciphertext attack (CCA1). It should be easy to see that all the discussions and proofs still hold in the case of CCA1, by simply restricting the adversary from interacting with the decryption oracles after the target ciphertext is generated in both Game ME and TD (Sect. 4.1).

- Share combination algorithm η: given the verification key VK, the ciphertext c, and a set Λ of $t+1$ shares, η either outputs the corresponding result $r = \eta(VK, c, \Lambda)$ or a special symbol $\perp$ that is different from all possible correct results.
- Encryption algorithm E_{TD}: the "opposite" of D_{TD}. This function is carried out in the normal manner by a single party and should follow the same definition as the encryption algorithm in a standard public key cryptosystem.

The operation of a threshold decryption scheme can be modelled as follows. There is a trusted dealer (e.g. the center) and a set of n decryption servers indexed $1, \ldots, n$. In an initialization phase, the dealer runs the key generation algorithm and creates PK, VK and $\boldsymbol{SK}$. SK_i is given to server i. To decrypt a ciphertext ψ, a client gives ψ to the servers, requesting a decryption share from each of them. It can verify the validity of the shares using the given verification key. Once the client collects valid shares from at least $t+1$ servers, she can apply η to obtain the decryption.

Threshold cryptosystems are part of a general approach known as threshold cryptography, introduced by Boyd [28], Desmedt [29], and Desmedt and Frankel [30]. There are schemes based on both Diffie-Hellman problem [30] and RSA [31]. All these schemes can be shown to be secure against chosen plaintext attack, but they are not known to withstand chosen ciphertext attack. After Cramer and Shoup discovered the first truly practical public key cryptosystem that is provably secure against chosen ciphertext attack *without* using random oracles [20], several of its threshold implementations have been proposed and proved CCA2 secure (also without using the random oracle model) [32–34]. Shoup and Gennaro presented a more efficient threshold scheme in [35] that is proven CCA2 secure in the random oracle model.

We adopt Shoup and Gennaro's definition of security for threshold decryption schemes from [35], which is a natural extension of security for a public key cryptosystem, and define the security of a $(t+1, n)$-threshold decryption scheme $\mathcal{TD} = (\mathsf{KeyGen}_{\mathsf{TD}}, \mathsf{D}_{\mathsf{TD}}, \mathsf{V}_{\mathsf{TD}}, \eta, \mathsf{E}_{\mathsf{TD}})$ with respect to the following game:

Game TD:

TD1 The adversary $\mathcal{A}$ chooses to corrupt a fixed set of t servers.
TD2 The key generation algorithm is run. The public key, verification key and the private keys of the corrupted servers are given to $\mathcal{A}$. Other private keys are given to the uncorrupted servers.
TD3 $\mathcal{A}$ chooses two plaintexts m_0 and m_1 of the same length and gives them to an "encryption oracle" that chooses $b \in \{0, 1\}$ at random, and gives the "target" ciphertext $\psi' = \mathsf{E}_{\mathsf{TD}}(PK, m_b)$ to $\mathcal{A}$.
TD4 At the end of the game, the adversary outputs $b' \in \{0, 1\}$.

This game defines the attack scenario for CPA security. The adversary's advantage is defined to be the absolute difference between 1/2 and the probability that $b' = b$:

$$\mathsf{Adv}^{CPA}_{\mathcal{TD},\mathcal{A}}(k) = |Pr(b' = b) - 1/2|$$

For CCA2 attacks, $\mathcal{A}$ is allowed to interact with uncorrupted decryption servers, who act as the decryption oracles, in an arbitrary fashion, feeding them ciphertexts $\psi \neq \psi'$, and obtaining decryption shares. The calls to the decryption oracles can happen at any point during the execution of the game, both before and after stage TD3, and be arbitrarily interleaved with other oracle calls. $\mathcal{A}$'s advantage is defined as

$$\mathsf{Adv}^{CCA2}_{\mathcal{TD},\mathcal{A}}(k) = |Pr(b' = b) - 1/2|$$

Definition 3 (t-Resilient Threshold Decryption Scheme). *Let $\mu \in \{CPA, CCA2\}$. A threshold decryption scheme $\mathcal{TD}$ is t-resilient against μ-type attacks if the advantage, $\mathsf{Adv}^{\mu}_{\mathcal{TD},\mathcal{A}}(k)$, of any probabilistic polynomial time adversary $\mathcal{A}$ is a negligible function of k.*

4.2 Basic Construction

Construction 1 (ME1) *Given a threshold decryption $\mathcal{TD} = (\mathsf{KeyGen}_{\mathsf{TD}}, \mathsf{D}_{\mathsf{TD}}, \mathsf{V}_{\mathsf{TD}}, \eta, \mathsf{E}_{\mathsf{TD}})$, a security parameter 1^k, a threshold t and the number of (initial) members n, a multicast encryption scheme $\mathcal{ME}^{\mathcal{TD}}_{C1} = (\mathsf{KeyGen}, \mathsf{E}, \mathsf{D})$ can be constructed as follows:*

1. *Key Generation* KeyGen*: Run $(PK, VK, \boldsymbol{SK}) \leftarrow \mathsf{KeyGen}_{\mathsf{TD}}(1^k, t, n+t)$. Set $I = (PK, VK)$ and the encryption key $\Sigma = \{(j, SK_j) : j = n+1, \ldots, n+t\}$. Σ is given to the center. Member i receives secret key $\Gamma_i = (i, SK_i)$. The master secret key is $\boldsymbol{\Gamma} = (\Gamma_1, \ldots, \Gamma_{n+t})$.*
2. *Encryption* E*: Given a set R of revoked members, and their secret keys, with $|R| \leq t$, a message m, the encryption proceeds as follows. Let $T = \{n+1, \ldots, n+t\}$. The encryptor randomly selects a subset of T with $t - |R|$ elements, denoted T', and computes the ciphertext $\psi = (c, \{(j, c_j) : j \in T' \cup R\})$ where $c = \mathsf{E}_{\mathsf{TD}}(PK, m)$ and $c_j = \mathsf{D}_{\mathsf{TD}}(SK_j, c)$.*
3. *Decryption* D*: Given a secret key Γ_i and a ciphertext ψ, the ciphertext is first parsed into $\psi = (c, \Lambda')$ where $\Lambda' = \{(j, c_j) : j \in T' \cup R\}$ with $c_j = \mathsf{D}_{\mathsf{TD}}(SK_j, c)$. For all $j \in T' \cup R$, the decryption first test $v_j = \mathsf{V}_{\mathsf{TD}}(VK, c, c_j)$. If any $v_j = 0$,* D *returns $\perp$. Otherwise it returns*

$$m = \eta(VK, c, \Lambda' \cup \{(i, \mathsf{D}_{\mathsf{TD}}(SK_i, c))\}) \tag{1}$$

With this construction, the multicast ciphertext essentially consists of the ciphertext of the underlying threshold scheme, together with t partial decryptions produced using the keys of revoked members. To decrypt, a legitimate member combines the partial decryptions embedded in the ciphertext with another one computed using her own share of the private key. As we will show, this construction preserves the security of the underlying threshold scheme.

Theorem 1 (Security Inheritance). *Let $\mu \in \{CPA, CCA2\}$. Given a threshold decryption scheme $\mathcal{TD} = (\mathsf{KeyGen}_{\mathsf{TD}}, \mathsf{D}_{\mathsf{TD}}, \mathsf{V}_{\mathsf{TD}}, \eta, \mathsf{E}_{\mathsf{TD}})$ that is t-resilient against μ-type attacks, the multicast encryption scheme $\mathcal{ME}^{\mathcal{TD}}_{C1}$ constructed using Construction 1 with threshold t and (initial) group size n is t-resilient against μ-type attacks.*

The proof of this theorem is similar to that of Theorem 2, which is more interesting and is presented later, and is omitted from this paper.

Many existing multicast schemes can be shown to be special cases of our Construction 1 and their security can be readily predicted by Theorem 1. The Revocation method 1 in [17] and the group key distribution scheme in [14] are just Construction 1 instantiated with a special use of threshold ElGamal [3]. The basic scheme in [22], the "public key (multicast) encryption" from [17] and the CPA secure scheme from [23] can all be shown to be Construction 1 with a standard threshold ElGamal cryptosystem. These schemes are shown to be secure against chosen plaintext attacks in their individual papers. The same conclusion can be reached immediately through Theorem 1.

Theorem 1 also provides guidelines for constructing *new* multicast encryption schemes with guaranteed security. For example, some threshold schemes are known to be CCA2 secure (e.g. [35, 32–34] and the IND-CCA2 threshold ElGamal in [36]) and a multicast encryption constructed via Construction 1 using one of these schemes is therefore guaranteed to be CCA2 secure too. In addition, all existing ATD-based multicast encryption schemes [17, 14, 22–24] are based on discrete logarithm. Theorem 1 provides security guarantee for constructing multicast encryption using any other assumptions. For example, [31] provides a threshold RSA scheme with CPA security. Such scheme can be used to construct a RSA-based CPA secure multicast cryptosystem. Another example of factorization-based scheme is the threshold version of Paillier cryptosystem [37] presented in [36]. [36] provides techniques to make this scheme IND-CCA2. A multicast cryptosystem with the same level of security based on Paillier cryptosystem can thus be constructed using Construction 1. All the above examples have never been proposed before. They are the natural products of Construction 1 and their security is guaranteed by Theorem 1.

4.3 Extension to Construction 1

Construction 1 provides a simple way to utilize a threshold scheme to construct multicast encryption and we have shown that the resulting scheme is as secure as the underlying threshold scheme. It is basically an "encrypt-then-decrypt-t-times" scheme. It can be improved both in efficiency and security with simple extension.

In Construction 1, the encryptor has access to what are equivalent to t decryption shares in $\mathcal{TD}$ which are not available to an encryptor in the underlying threshold scheme. This gives her a chance to "protect" these shares and, as a result, the resulting multicast encryption can be made more secure than $\mathcal{TD}$. This can be seen as an extension of Construction 1:

[3] Their scheme uses this construction not to encrypt any useful messages. Instead, it is basically a distributed Diffie-Hellman key exchange which is equivalent to producing an ElGamal encryption of an arbitrary message (which is ignored) and allowing any member with proper keys to derive from the ciphertext, and partial decryptions, a secret key that can be used to encrypt actual data.

Construction 1e (ME1e) *Same as Construction 1 except for the following:*

- *The encryption* E *produces ciphertext as* $\psi = (c, \{(j, \mathsf{D}_{\mathsf{TD}}(SK_j, c)) : j \in T' \cup R\}, v)$ *where* $c = \mathsf{E}_{\mathsf{TD}}(PK, m)$ *and* $v = \mathsf{Tag}(c, \Sigma, I)$ *is a "tag" for the ciphertext.*
- *The decryption* D *first computes* $\mathsf{Valid}(\Gamma_i, \psi, I)$ *where* Valid *is a checking function outputting* 0*, or* 1*. If* Valid *outputs 0,* D *returns* $\perp$*. Otherwise it proceeds the same as Construction 1.*

This construction can be used to build a multicast scheme with higher security than the underlying threshold scheme. This is essentially what was done in [23] and [24]. The protection mechanism (i.e. Tag and Valid) depends on the threshold scheme and the security goal. In [23], the standard techniques of [20] (which attaches tags to the ciphertext so that the recipients with proper keys can verify its validity) was applied to protect the decryption shares and the security achieved is what [23] called gCCA2 (Generalized CCA) which is a variant, and weaker version, of CCA2. To achieve real CCA2 security, [23] used secure message authentication code (MAC) to make the verification tags non-malleable. And [24] essentially used a threshold version of M-CS [32].

4.4 Sharable Trapdoor Permutation-Based Construction

A whole class of public key cryptosystems are based on trapdoor permutations. Let $f_{PK} : \{0,1\}^k \rightarrow \{0,1\}^k$ be a k-bit to k-bit trapdoor (one-way) permutation with inverse f_{SK}^{-1}, defined by the public-private key pair (PK, SK). A public key cryptosystem $\mathcal{E}^{f,g,h}$ encrypts a message m as $\mathsf{E}(m) = h(f_{PK}(g(m)))$ where g and h are probabilistic, invertible functions that specify pre- and post-encoding operations, respectively. Given a ciphertext c, the decryption algorithm D computes $u = h^{-1}(c), v = f_{SK}^{-1}(u)$ and $m = g^{-1}(c, u, v)$ [4]. Depending on the security, the decryption may involve computing $\mathsf{Valid}(c, u, v) \in \{0,1\}$ which is the verification of the encoding. The decryption returns $\perp$ if $\mathsf{Valid}(c, u, v) = 0$. We denote such cryptosystem as $\mathcal{E}^{f,g,h} = (\mathsf{KeyGen}, \mathsf{E}, \mathsf{D}, \mathsf{Valid})$ where KeyGen generates (PK, SK) on given security parameter 1^k. In the following, the keys will be dropped from the notations when there is no need to make them explicit.

Such cryptosystems are prevalent in practice. One example is the RSA Public Key Cryptography Standard # 1 [38], where $g(m)$ is essentially m padded with a string of random non-zero bytes in the high-order bit positions and post-encoding is simply omitted. Other schemes make use of hash functions. Let $G : \{0,1\}^* \rightarrow \{0,1\}^\infty$ be a random number generator and $H : \{0,1\}^* \rightarrow \{0,1\}^{k_0}$ be a hash function where $l = k - k_0$ is the length of the message. In [39] Bellare and Rogaway proposed the scheme $\mathcal{E}_{BR}^{G}$ where $\mathsf{E}(m) = f(r) \parallel G(r) \oplus m$ with

[4] Note that g and h are easily invertible and do not require trapdoors. Also note that both g and h are probabilistic and $g(m)$ maybe independent of m. In this case simply inverting v does not reveal m. However, these decryptions all have the following property: once the pre-image of the trapdoor permutation is recovered, it is easy to compute m. We simply use $g^{-1}(\cdot)$ to denote this process.

$r \leftarrow_R \{0,1\}^k$. [39] showed that it is semantically secure in the random oracle model. [39] also presented another scheme, denoted $\mathcal{E}_{BR}^{G,H}$, that is shown to be CCA2 secure, also in the random oracle model. In $\mathcal{E}_{BR}^{G,H}$, message m is encrypted as $\mathsf{E}(m) = (f(r), m \oplus G(r), H(r,m))$ where $r \leftarrow_R \{0,1\}^k$. Given a ciphertext (s, c, v), the decryption algorithm computes $r = f^{-1}(s), m = G(r) \oplus c$, and $v' = H(r,m)$. If $v' = v$, it outputs m, and $\perp$ otherwise.

Another popular scheme is the OAEP scheme introduced in [40]. In this scheme, to encrypt a message m of length l bits, one selects a random value $r \leftarrow_R \{0,1\}^{k_0}$ and computes $s = (m \parallel 0^{k_1}) \oplus G(r)$ and $t = r \oplus H(s)$ where $k_1 = k - l - k_0$. The ciphertext is $c = f(s,t)$. To decrypt a ciphertext c, the decryptor extracts (s,t) using the private key $(s,t) = f^{-1}(c)$ and computes $r = t \oplus H(s)$ and $M = s \oplus G(r)$. If $[M]_{k_1} = 0^{k_1}$, it returns $[M]^l$. Otherwise it returns $\perp$. In the above, $[M]_l$ (resp. $[M]^l$) denotes the l least (resp. most) significant bits of M.

In [40], Bellare and Rogaway proved that OAEP construction together with any trapdoor one-way permutation is IND-CCA1. OAEP was widely believed to achieve stronger security (i.e. IND-CCA2). But Shoup showed in [41] that it is unlikely such security proof exists, for any trapdoor permutation. However, he proved that, when instantiated with low-exponent RSA, OAEP was IND-CCA2. This result was extended to arbitrary exponent RSA in [42].

All these schemes provide practical public key cryptosystems with various security and efficiency. (The OAEP scheme provides optimal bit complexity in that the ciphertext size is only slightly greater than that of plaintext.) However, they do not have threshold implementations that retain the same security, especially at CCA2 level. As Shoup and Gennaro noted in [35], the difficulty in transforming a non-threshold CCA secure public key encryption scheme, $\mathcal{E}$, into a CCA secure threshold scheme is that $\mathcal{E}$'s security proof can rely in a critical way on the fact that the decryption algorithm makes the "validity test" before generating an output. In a distributed setting, this means the test can only be performed *after* the individual decryption shares are combined. A single decryption server is unable to carry out such test. Both $\mathcal{E}_{BR}^{G,H}$ [39] and OAEP can be easily shown to have this difficulty.

One way to address this difficulty is to introduce a validity test that is publicly checkable so that a decryptor can perform the check before carrying out the decryption. This was suggested in [43] and followed by systems such as [35] which used non-interactive zero-knowledge proofs of membership to construct such check which is costly.

An ATD-based multicast encryption scheme, on the other hand, does not suffer from this difficulty at all. This is because in such a scheme, the decryptor is presented with what are equivalent to t decryption shares in the underlying sharing scheme. She can proceed to combine these shares with the one produced using her private key and perform the simple validity test as in the original public key cryptosystem (not the expensive publicly checkable threshold version) *before* emitting any output. As we show in Theorem 2, this construction preserves the CCA security of the public key cryptosystem even though its threshold implementation does not.

Our new construction is based on sharable trapdoor functions.

Definition 4 ($(t+1, n)$-Secure Sharing Scheme). *Let f be a trapdoor function with inverse f^{-1} defined by the public-private key pair (PK, SK). A sharing scheme $\mathcal{SS}^f = (\mathsf{S}, \eta)$ for f consists of two polynomial time algorithms:*

- S*: Given (PK, SK), a threshold t and an integer $n > t$, S generates $SK_1, \ldots, SK_n$ (in the same space as SK), called* shares *of SK.*
- η*: Given the public key PK, a set Λ of $t+1$ evaluations $f^{-1}_{SK_i}(u)$, for any u in the domain of f_{PK}, η computes $f^{-1}_{SK}(u)$.*

And $\mathcal{SS}^f$ is $(t+1, n)$-secure if for all $\{i_1, \ldots, i_j\} \subset U$ where $0 \leq j \leq t < n$, for all probabilistic polynomial time algorithm A, for all polynomial $poly(\cdot)$, for all k large enough

$$\begin{aligned} &Pr[f_{PK}(u) = w : (SK_1, \ldots, SK_n) \leftarrow \mathsf{S}(PK, SK, t, n); \\ &w \in_R \{0,1\}^k; u \leftarrow A(1^k, w, H, SK_{i_1}, \ldots SK_{i_j})] < 1/poly(k) \end{aligned}$$

where H is the history tape of length polynomial in k containing all the partial evaluations the players generated so far.

And f is $(t+1, n)$-sharable if it has one $(t+1, n)$-secure sharing scheme.

This is essentially the same definition as $(t+1, n)$-secure function sharing primitive in [31]. [31] also showed how to implement such sharing with trapdoor permutations such as RSA. We show that using this primitive we can construct efficient multicast encryption schemes with high security.

Construction 2 (ME2) *Let $\mathcal{E}^{f,g,h} = (\mathsf{KeyGen_E}, \mathsf{E_E}, \mathsf{D_E}, \mathsf{Valid})$ be a public key cryptosystem based on $(t+1, n)$-sharable trapdoor permutation f with sharing scheme $\mathcal{SS}^f = (\mathsf{S}, \eta)$. Given a security parameter 1^k, a threshold t and the number of (initial) members n, a multicast encryption scheme $\mathcal{ME}^{\mathcal{E}^f}_{C2} = (\mathsf{KeyGen}, \mathsf{E}, \mathsf{D})$ can be constructed as follows:*

1. *Key Generation KeyGen: The center runs $\mathsf{KeyGen_E}$ with parameter 1^k, and obtains $(PK, SK) \leftarrow \mathsf{KeyGen_E}(1^k)$. It sets $I = PK$ and shares SK using the sharing algorithm S with parameter $(t+1, n+t)$ to obtain $\boldsymbol{SK} = \mathsf{S}(PK, SK, t, n+t)$. The encryption key is $\Sigma = \{(j, SK_j) : j \in T\}$ where $T = \{n+1, \ldots, n+t\}$. Σ is given to the center. Member i receives secret key $\Gamma_i = (i, SK_i)$. The master secret key is $\boldsymbol{\Gamma} = (\Gamma_1, \ldots, \Gamma_{n+t})$.*
2. *Encryption E: Given a set R of revoked members, and their secret keys, with $|R| \leq t$, a message m, the encryptor randomly selects a subset of T with $t - |R|$ elements, denoted T', and computes the ciphertext*

$$\psi = (c, \{(j, f^{-1}_{SK_j}(u)) : j \in T' \cup R\}) \tag{2}$$

where $c = \mathsf{E_E}(PK, m)$ and $u = h^{-1}(c)$.

3. *Decryption* D*: Given a secret key Γ_i and a ciphertext ψ, the ciphertext is first parsed into $\psi = (c, \Lambda')$ where $\Lambda' = \{(j, f^{-1}_{SK_j}(u)) : j \in T' \cup R\}$. The decryptor computes $u = h^{-1}(c)$ and $v = \eta(u, \Lambda' \cup \{(i, f^{-1}_{SK_i}(u))\})$. If all these steps are successful, it computes $w = \mathsf{Valid}(c, u, v)$. If $w = 0$, it returns $\perp$. Otherwise it returns $m = g^{-1}(c, u, v)$.*

Theorem 2. *Let $\mu \in \{CPA, CCA2\}$. If a public key cryptosystem $\mathcal{E}^{f,g,h} = (\mathsf{KeyGen}_\mathsf{E}, \mathsf{E}_\mathsf{E}, \mathsf{D}_\mathsf{E}, \mathsf{Valid})$ based on $(t+1, n)$-sharable trapdoor permutation f with sharing scheme $\mathcal{SS}^f = (\mathsf{S}, \eta)$ is secure against μ type attacks, then a multicast encryption scheme $\mathcal{ME}^{\mathcal{E}}_{C2} = (\mathsf{KeyGen}, \mathsf{E}, \mathsf{D})$ constructed using Construction 2 with threshold t and (initial) group size n is t-resilient against μ-type attacks.*

Proof. First note that it is trivial to verify that the scheme is correct – i.e., the decryption produces the correct plaintext given a valid ciphertext. We prove its security by showing that if $\mathcal{ME}^{\mathcal{E}}_{C2}$ is not t-resilient against μ-type attacks, neither is $\mathcal{E}^{f,g,h}$. Let $\mathcal{A}^{\mathcal{ME}}$ be a polynomial time adversary that wins the game ME with non-negligible advantage. We can construct another polynomial time adversary $\mathcal{A}^{\mathcal{E}}$ that breaks $\mathcal{E}^{f,g,h}$ with at least the same advantage. $\mathcal{A}^{\mathcal{E}}$ achieves this by simulating a game ME and running $\mathcal{A}^{\mathcal{ME}}$ to win.

$(PK, SK) \leftarrow \mathsf{KeyGen}_\mathsf{E}(1^k)$ is run and PK is given to $\mathcal{A}^{\mathcal{E}}$ while SK is kept secret from it. $\mathcal{A}^{\mathcal{E}}$ selects randomly t numbers $SK_1, \ldots, SK_t$ from the space of SK. $\mathcal{A}^{\mathcal{E}}$ starts Game ME and lets $\mathcal{A}^{\mathcal{ME}}$ select t members to corrupt. Without loss of generality, let $T = \{1, 2, \ldots, t\}$ be the indexes of the members $\mathcal{A}^{\mathcal{ME}}$ chooses to corrupt. $\mathcal{A}^{\mathcal{E}}$ simulates the key generation process in game ME and gives $\Sigma = ((1, SK_1), \ldots, (t, SK_t))$ as the corrupted keys and $I = PK$ as the public information to $\mathcal{A}^{\mathcal{ME}}$.

$\mathcal{A}^{\mathcal{E}}$ lets $\mathcal{A}^{\mathcal{ME}}$ run and simulates the rest of game ME as follows:

- Whenever $\mathcal{A}^{\mathcal{ME}}$ queries the encryption oracle with message m, $\mathcal{A}^{\mathcal{E}}$ returns ψ computed using Equation 2 with $T' \cup R$ replaced by T.
- $\mathcal{A}^{\mathcal{E}}$ chooses whatever $\mathcal{A}^{\mathcal{ME}}$ choose as the two test plaintexts m_0 and m_1. Whenever $\mathcal{A}^{\mathcal{ME}}$ makes a query to the encryption oracle with m_0 and m_1, $\mathcal{A}^{\mathcal{TD}}$ passes them to its own encryption oracle in its game attacking $\mathcal{E}^{f,g,h}$ (denoted game E). Let c' be the result returned by the encryption oracle in game E. $\mathcal{A}^{\mathcal{E}}$ computes and returns the following to $\mathcal{A}^{\mathcal{ME}}$:
$$\psi' = (c', \{(j, f^{-1}_{SK_j}(u)) : j \in T\}) \tag{3}$$
where $c = \mathsf{E}_\mathsf{E}(PK, m)$ and $u = h^{-1}(c')$. This corresponds to the target ciphertext in game ME.
- In the case of $\mu =$ CCA2, whenever $\mathcal{A}^{\mathcal{ME}}$ makes a query to one of the decryption oracles with ciphertext ψ, $\mathcal{A}^{\mathcal{E}}$ first parses ψ into a form as specified by Equation 2. Let $\{(j, u_j) : j \in T\}$ be the shares embedded in ψ. $\mathcal{A}^{\mathcal{E}}$ then verifies these shares by checking whether $u_j = f^{-1}_{SK_j}(u)$, where $u = h^{-1}(c)$, holds. If any of the tests fails it returns $\perp$ to $\mathcal{A}^{\mathcal{ME}}$. Otherwise it forwards c to its own decryption oracle and passes whatever the decryption oracle returns to $\mathcal{A}^{\mathcal{ME}}$.

$\mathcal{A}^{\mathcal{E}}$ stops when $\mathcal{A}^{\mathcal{ME}}$ stops and outputs whatever the latter does.

We need to show that $\mathcal{A}^{\mathcal{ME}}$ simulated by $\mathcal{A}^{\mathcal{E}}$ has all the information it would have in a real game ME and that its interaction with the simulated oracles is indistinguishable from that in a real game. First note that here, although the encryption key for $\mathcal{A}^{\mathcal{ME}}$, $SK_1, \ldots, SK_t$, are not actually generated by running S ($\mathcal{A}^{\mathcal{E}}$ does not have access to SK), they are just as good: the encryption key given to $\mathcal{A}^{\mathcal{ME}}$ is not distinguishable from that in a real game ME and does not affect its ability to win the game. This follows Lemma 1 from [31].

Second, $\mathcal{A}^{\mathcal{ME}}$ will receive $\perp$ on ciphertext ψ in the simulated game ME in one of the following two cases: (1) $\mathcal{A}^{\mathcal{E}}$'s decryption oracle returns $\perp$ on c; and (2) one of the tests on $u_j = f^{-1}_{SK_j}(u)$ fails. In the first case, $\mathcal{A}^{\mathcal{ME}}$ will receive $\perp$ in a real game ME, as specified by the decryption in Construction 2. In the second case $f^{-1}_{SK_j}(u)$, together with any partial evaluation of one of the decryption oracles in a real game ME, will combine to a u' that is not consistent with c and will fail Valid (otherwise it can be shown that either f is not $(t+1, n)$-sharable or $\mathcal{E}^{f,g,h}$ is not IND-CCA2). Again $\mathcal{A}^{\mathcal{ME}}$ will receive $\perp$ in a real game ME.

And in all other cases $\mathcal{A}^{\mathcal{ME}}$ will receive the correct decryption in both real and the simulated game ME. So if $\mathcal{A}^{\mathcal{ME}}$ can win a real game, it can win the simulated one.

It is easy to verify that if $\mathcal{A}^{\mathcal{ME}}$ wins the simulated game ME, $\mathcal{A}^{\mathcal{E}}$ distinguishes the two target ciphertexts with at least the same advantage. This is because, by definition of Construction 2, if ψ' in Equation 3 is the encryption of $m_{b'}$ in $\mathcal{ME}$, c' must be the encryption of $m_{b'}$ in $\mathcal{E}$.

Finally $\mathcal{A}^{\mathcal{E}}$'s running time is polynomial in that of $\mathcal{A}^{\mathcal{ME}}$ which itself is a polynomial in k. So $\mathcal{A}^{\mathcal{E}}$'s running time is also polynomial in k.

This is very powerful result because securing threshold scheme is hard so it is not always possible to use Construction 1 to construct multicast cryptosystems with high security. Construction 2 and Theorem 2 offer a simple method to construct multicast schemes with guaranteed security using a whole class of existing primitives. For instance, both RSA-OAEP [42] and $\mathcal{E}^{G,H}_{BR}$ [39], which have been shown to be difficult to obtain threshold implementations with the same level of security, can be used to build multicast scheme with CCA2 security. This has never been achieved before.

Besides security, Construction 2 also enjoys higher efficiency than Construction 1, which directly uses a threshold scheme. Note that in a sharing scheme used by Construction 2, there is neither decryption share verification nor publicly checkable validity test on ciphertext, both of which are essential for a threshold scheme or a real function sharing application to achieve robustness (as in e.g. [44]) and CCA security. With Construction 2, both can be omitted and the encoding verification that is part of the public key cryptosystem used can achieve both goals.

4.5 From IND-CPA to IND-CCA: Generic Conversion

In Construction 2, the security of $\mathcal{ME}_{C2}^{\mathcal{E}}$ relies on that of $\mathcal{E}^{f,g,h}$. Combined with results from previous work, we show that $\mathcal{ME}_{C2}^{\mathcal{E}}$ can be IND-CCA even if $\mathcal{E}^{f,g,h}$ is only IND-CPA.

In [45] Naor and Yung presented a generic conversion from an IND-CPA public key cryptosystem to one secure against "lunch-time" attack (a.k.a. non-adaptive chosen ciphertext attack, CCA1). The conversion used a twin-encryption paradigm and non-interactive zero-knowledge proof (NIZKP) of language membership in the common random string setting to show the consistency of the ciphertext. Rackoff and Simon later [46] improved this construction to be secure against adaptive chosen ciphertext attack (CCA2). Their solution involves replacing one of the twin encryption keys with the *sender*'s public key and providing a NIZKP of knowledge of the plaintext. [36] also provided similar conversion, in the random oracle model, that also works directly with *threshold* cryptosystems. The NIZKPs used in [45, 46, 36] are all publicly verifiable thus can be readily used in a threshold setting.

Putting all these together, we have the following whose proof immediately follows the results of [36, 46, 45] and ours.

Corollary 1. *If a public key cryptosystem $\mathcal{E}^{f,g,h}$ based on $(t+1,n)$-sharable trapdoor permutation f with sharing scheme $\mathcal{SS}^f$ is secure against chosen plaintext attacks, then there exists a multicast encryption scheme $\mathcal{ME}$ by Construction 2 with threshold t and (initial) group size n that is t-resilient against chosen ciphertext attacks.*

SUMMARY. Figure 1 summarizes the possible conversions covered in this paper between various primitives, including public key cryptosystem (PKC), threshold decryption scheme (TD) and multicast encryption (ME), at different security levels such as IND-CPA, IND-CCA (1 and 2). A solid arrow from A to B indicates "generic conversion", meaning that, under some reasonable assumptions, *any* A can be transformed into B. A dashed arrow, on the other hand, denotes "existential conversion", meaning that *some* A can be transformed into B. The conditions under which such conversions can succeed were stated in the literature. Some of the relevant ones covered in this paper are labelled on the arrows.

5 Conclusion

In this paper we have presented a general framework for constructing efficient multicast cryptosystems with provable security. Our constructions are based on asymmetric use of threshold schemes and we showed that a line of previous work on multicast encryption are all special cases of this general approach. We provided new methods for constructing multicast cryptosystems that achieve various levels of security (e.g., IND-CPA, IND-CCA2) from primitives with even weaker security. Using our scheme, each member only needs to store a key of

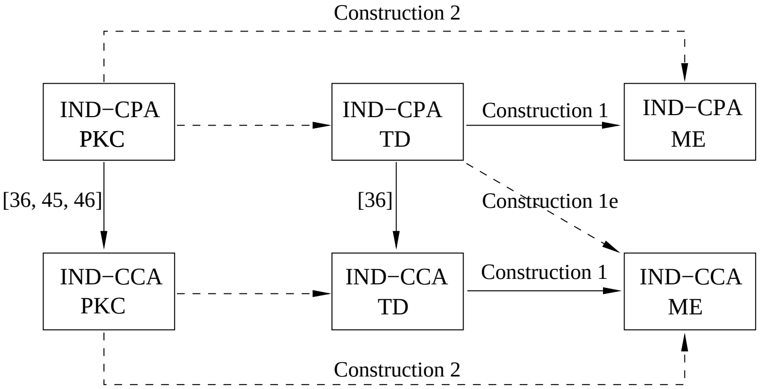

Fig. 1. Conversions.

constant length while both the encryption key size and the ciphertext length are $O(t)$ which is independent of the group size.

Acknowledgements The first author would like to thank David Wagner for his encouragement and suggestions during the inception of this work. The authors thank the anonymous reviewers for their valuable comments.

References

1. Fenner, W.: Internet group management protocol, version 2. RFC-2236 (1997)
2. Harney, H., Muckenhirn, C.: Group key management protocol (gkmp) architecture. IETF Request for Comments, RFC 2094 (1997)
3. Wallner, D., Harder, E., Agee, R.: Key management for multicast: Issues and architectures. IETF Request For Comments, RFC 2627 (1999)
4. Wong, C.K., Gouda, M., Lam, S.S.: Secure group communications using key graphs. IEEE/ACM Trans. Netw. **8** (2000) 16–30
5. Canetti, R., Garay, J., Itkis, G., Micciancio, D., Naor, M., Pinkas, B.: Multicast security: A taxonomy and some efficient constructions. In: INFOCOMM'99. (1999)
6. Chang, I., Engel, R., Kandlur, D., Pendarakis, D., Saha, D.: Key management for secure internet multicast using boolean function minimization techniques. In: Proceedings IEEE Infocomm'99. Volume 2. (1999) 689–698
7. Wong, C.K., Lam, S.S.: Keystone: A group key management service. In: International Conference on Telecommunications, ICT 2000. (2000)
8. Li, X.S., Yang, Y.R., Gouda, M.G., Lam, S.S.: Batch rekeying for secure group communications. In: Proceedings of the tenth international World Wide Web conference on World Wide Web, Orlando, FL USA (2001) 525–534
9. Setia, S., Koussih, S., Jajodia, S., Harder, E.: Kronos: A scalable group re-keying approach for secure multicast. In: IEEE Symposium on Security and Privacy. (2000) 215–228
10. Yang, Y.R., Li, X.S., Zhang, X.B., Lam, S.S.: Reliable group rekeying: a performance analysis. In: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, ACM Press (2001) 27–38

11. Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: CRYPTO 1994. Volume 839 of Lecture Notes in Computer Science., Springer-Verlag (1994) 257–270
12. Fiat, A., Naor, M.: Broadcast encryption. In: CRYPTO 1993. Volume 773 of Lecture Notes in Computer Science., Springer-Verlag (1994) 480–491
13. Boneh, D., Franklin, M.: An efficient public key traitor tracing scheme. In: CRYPTO 1999. Volume 1666 of Lecture Notes in Computer Science., Springer-Verlag (1999) 338–353
14. Anzai, J., Matsuzaki, N., Matsumoto, T.: A quick group key distribution scheme with "entity revocation". In: ASIACRYPT 1999. Volume 1716 of Lecture Notes in Computer Science., Singapore, Springer (1999) 333–347
15. Luby, M., Staddon, J.: Combinatorial bounds for broadcast encryption. In: EUROCRYPT 1998. Volume 1403 of Lecture Notes in Computer Science., Springer-Verlag (1998) 512–526
16. Garay, J.A., Staddon, J., Wool, A.: Long-lived broadcast encryption. In: CRYPTO 2000. Volume 1880 of Lecture Notes in Computer Science., Springer-Verlag (2000) 333–352
17. Naor, M., Pinkas, B.: Efficient trace and revoke schemes. In: Proceedings of Financial Crypto 2000. (2000)
18. Halevy, D., Shamir, A.: The LSD broadcast encryption scheme. In: CRYPTO 2002. Volume 2442 of Lecture Notes in Computer Science., Springer-Verlag (2002) 47–60
19. Naor, D., Naor, M., Lotspiech, J.B.: Revocation and tracing schemes for stateless receivers. In: CRYPTO 2001. Volume 2139 of Lecture Notes in Computer Science., Springer-Verlag (2001) 41–62
20. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: CRYPTO 1998. Volume 1462 of Lecture Notes in Computer Science., Springer-Verlag (1998) 13–25
21. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: CRYPTO 2005. Volume 3621 of Lecture Notes in Computer Science., Springer-Verlag (2005) 258–275
22. Tzeng, W.G., Tzeng, Z.J.: A public-key traitor tracing scheme with revocation using dynamic shares. In: Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography, Springer-Verlag (2001) 207–224
23. Dodis, Y., Fazio, N.: Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In: Workshop on Public Key Cryptography – PKC '03. Volume 2567 of Lecture Notes in Computer Science. (2003) 100–115
24. Kim, C.H., Hwang, Y.H., Lee, P.J.: An efficient public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In: ASIACRYPT 2003. Volume 2894 of Lecture Notes in Computer Science., Springer-Verlag (2003) 359–373
25. Liu, D., Ning, P., Sun, K.: Efficient self-healing group key distribution with revocation capability. In: Proceedings of the 10th ACM conference on Computer and communication security, ACM Press (2003) 231–240
26. Staddon, J., Miner, S., Franklin, M., Balfanz, D., Malkin, M., Dean, D.: Self-healing key distribution with revocation. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy, IEEE Computer Society (2002) 241
27. Wang, H.: Resilient lkh: Secure multicast key distribution schemes. In: Proceedings of the 2003 International Workshop on Advanced Developments in Software and Systems Security (WADIS). (2003)
28. Boyd, C.: Digital multisignatures. Cryptography and Coding (1986) 241–246

29. Desmedt, Y.: Society and group oriented cryptography: A new concept. In: CRYPTO 1987. Volume 293 of Lecture Notes in Computer Science., Springer-Verlag (1987) 120–127
30. Desmedt, Y.G., Frankel, Y.: Threshold cryptosystems. In: CRYPTO 1989. Volume 435 of Lecture Notes in Computer Science., Springer-Verlag (1989) 307–315
31. De Santis, A., Desmedt, Y., Frankel, Y., Yung, M.: How to share a function securely. In: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing, ACM Press (1994) 522–533
32. Canetti, R., Goldwasser, S.: An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In: EUROCRYPT 1999. Volume 1592 of Lecture Notes in Computer Science., Springer-Verlag (1999) 90–106
33. Abe, M.: Robust distributed multiplication without interaction. In: CRYPTO 1999. Volume 1666 of Lecture Notes in Computer Science., Springer-Verlag (1999) 130–147
34. Jarecki, S., Lysyanskaya, A.: Adaptively secure threshold cryptography: Introducing concurrency, removing erasures (extended abstract). In: Proceedings of Eurocrypt 2000. Volume 1807 of Lecture Notes in Computer Science., Springer-Verlag (2000) 221–242
35. Shoup, V., Gennaro, R.: Securing threshold cryptosystems against chosen ciphertext attack. J. Cryptology **15** (2002) 75–96
36. Fouque, P.A., Pointcheval, D.: Threshold cryptosystems secure against chosen-ciphertext attacks. In: ASIACRYPT 2001. Volume 2248 of Lecture Notes in Computer Science., Springer-Verlag (2001) 351–368
37. Paillier, P.: Public-key cryptosystems based on discrete logarithms residues. In: EUROCRYPT 1999. Volume 1592 of Lecture Notes in Computer Science., Springer-Verlag (1999) 223–238
38. RSA Labs: PKCS#1 v2.1: RSA cryptography standard (2002)
39. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM conference on Computer and communications security, ACM Press (1993) 62–73
40. Bellare, M., Rogaway, P.: Optimal asymmetric encryption – how to encrypt with RSA. In: EUROCRYPT 1994. Volume 950 of Lecture Notes in Computer Science., Springer-Verlag (1994) 92–111
41. Shoup, V.: OAEP reconsidered. In: CRYPTO 2001. Volume 2139 of Lecture Notes in Computer Science., Springer-Verlag (2001) 239–259
42. Fujisaki, E., Okamoto, T., Pointcheval, D., Stern, J.: RSA-OAEP is secure under the rsa assumption. In: CRYPTO 2001. Volume 2139 of Lecture Notes in Computer Science., Springer-Verlag (2001) 260–274
43. Lim, C.H., Lee, P.J.: Another method for attaining security against adaptively chosen ciphertext attacks. In: CRYPTO 1993. Volume 773 of Lecture Notes in Computer Science., Springer-Verlag (1993) 420–434
44. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Robust and efficient sharing of RSA functions. In: CRYPTO 1996. Volume 1109 of Lecture Notes in Computer Science., Springer-Verlag (1996) 157–172
45. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the twenty-second annual ACM symposium on Theory of computing, ACM Press (1990) 427–437
46. Rackoff, C., Simon, D.R.: Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: CRYPTO 1991. Volume 576 of Lecture Notes in Computer Science., Springer-Verlag (1992) 433–444