Privacy in Encrypted Content Distribution Using Private Broadcast Encryption

Adam Barth[1], Dan Boneh*[1], and Brent Waters[2]

[1] Stanford University, Stanford, CA 94305
{abarth, dabo}@cs.stanford.edu
[2] SRI International, Menlo Park, CA 94025
bwaters@csl.sri.com

Abstract. In many content distribution systems it is important both to restrict access to content to authorized users and to protect the identities of these users. We discover that current systems for encrypting content to sets of users are subject to attacks on user privacy. We propose a new mechanism, private broadcast encryption, to protect the privacy of users of encrypted file systems and content delivery systems. We construct a private broadcast scheme, with a strong privacy guarantee against an active attacker, that achieves ciphertext length, encryption time, and decryption time comparable with the non-private schemes currently used in encrypted file systems.

1 Introduction

In both large and small scale content distribution systems it is often important to make certain data available to only a select set of users. In commercial content distribution, for example, a company may wish for its digital media to be available only to paying customers. On a smaller scale, suppose a department's faculty need to access the academic transcripts of graduate applicants. If electronic copies of the transcripts were stored on the department's file server, they should only be accessible by the faculty and students.

It is often equally important to protect the identities of the users who are able to access protected content. Students receiving an email from an instructor notifying all students failing a class would likely wish to keep their identities private. Commercial sites often not want to disclose identities of customers because competitors might use this information for targeted advertising. If an employee is up for promotion, a company might wish to hide who is on his promotion committee and therefore who is able to read his performance evaluation.

Employing a trusted server is the most commonly used method for protecting both electronic content and the privacy of users who can access it. Whenever a user wishes to access content stored on a trusted server, the user contacts the server, authenticates himself or herself, and is sent the content over a secure channel. As long as the server behaves correctly, only authorized users will be

* Supported by NSF.

able to access the content and which users are authorized to access which content will not be divulged, even to other authorized users. While this simple method of data protection is adequate for some applications, it has some significant drawbacks. First, both data content and user privacy are subject to attack if the server is compromised. Additionally, content providers often not distribute their data directly, but for economic reasons outsource distribution to third parties or use peer-to-peer networks. In this case, the content owners will no longer be directly in control of data distribution.

For these reasons, we examine the problem of efficiently distributing encrypted content in such a way that (1) only authorized users can read the content and (2) the identities of authorized users are hidden. We study this problem for the case of encrypted file systems. However, our results can be generalized to larger content distribution systems, including encrypted email.

Encrypted File Systems. Encrypted file systems implement read access control by encrypting the contents of files such that only users with read permission are able to perform decryption. Typical encrypted file systems, such as Windows EFS, encrypt each file under its own symmetric key, K_F, and then encrypt the symmetric key separately under the public keys of those users authorized to access the file, labeling these encryptions with user identities to speed decryption (Fig. 1(a)).

(a)
$$\begin{array}{l} A : \{K_F\}_{\mathsf{pk}_A} \\ B : \{K_F\}_{\mathsf{pk}_B} \\ C : \{K_F\}_{\mathsf{pk}_C} \\ \quad \{F\}_{K_F} \end{array}$$

(b)
$$\begin{array}{l} \{K_F\}_{\mathsf{pk}_B} \\ \{K_F\}_{\mathsf{pk}_C} \\ \{K_F\}_{\mathsf{pk}_A} \\ \{F\}_{K_F} \end{array}$$

Fig. 1. Simple constructions of broadcast encryption systems. File F is encrypted under the key K_F, which in turn is encrypted under the public keys of users A, B, and C. (a) The scheme typically used by encrypted file systems reveals the set of users authorized to access F. (b) Modifying this scheme by removing the labels, using a key-private cryptosystem, and randomly reordering the users yields a private broadcast scheme resistant to passive attacks on recipient privacy, but decryption time is increased because recipients must attempt to decrypt each unlabeled component. These simple schemes are both vulnerable to active attacks.

While these systems protect file contents from unauthorized users, they do little to protect the identities of users allowed to access the file. Who can access a file, however, is often more sensitive than the contents of the file itself. Suppose, for example, a university provides a document on its file server to students with low average grades. To maintain the privacy of the students, the set of authorized users should be kept private, not only from outsiders, but from the students in the group as well.

Current implementations expose the identities of authorized users in two ways. First, the individual public key encryptions of the symmetric key, K_F, are

labeled with the identity of the user, as shown in Fig. 1(a). These labels direct authorized users to their encryptions of K_F, speeding decryption. Second, even without these labels, an adversary examining the actual ciphertexts can learn information about the user's identity. For example, suppose an attacker wants to determine whether Alice or Bob has access to a particular file. Further suppose Alice has a 1024-bit key and Bob has a 2048-bit key. An attacker can easily determine which of the two has access to the file by examining the encryption of K_F, specifically the ciphertext length. Thus, the encryptions of K_F leak some information about who has access to the file.

Private Broadcast Encryption. Our goal is to provide recipient privacy: an encrypted file should hide who can access its contents. We approach the problem of recipient privacy by introducing a notion we call *private broadcast encryption*. A private broadcast encryption scheme encrypts a message to several recipients while hiding the identities of the recipients, even from each other.

The most straightforward construction of a private broadcast encryption scheme is to modify the scheme currently used in encrypted file systems by removing the identifying labels and using a public key system that does not reveal the public key associated with a ciphertext, such as ElGamal or Cramer-Shoup [1] (Fig. 1(b)). While this scheme is secure against passive attacks on recipient privacy, it has two disadvantages. First, decryption time is increased as recipient must perform, on average, $n/2$ trial decryptions to obtain K_F, where n is the number of message recipients. Second, an active attacker can mount a chosen-ciphertext attack and learn whether a user can decrypt a message.

Returning to our example, consider an active attacker who is authorized to decrypt the document for students with low average grades, where the list of authorized users should be private. Now, suppose that the attacker wishes to determine whether Alice can read the document. Because the attacker is a legitimate recipient, he or she knows K_F and can maliciously prepare a different encrypted file by replacing the encrypted contents of the original file with content of the attacker's choice, encrypted under K_F. Alice is able to read this maliciously created file if, and only if, she can read the original file. For example, a malicious legitimate recipient of the document could copy the document header, but replace the document body with the message "please visit the following URL for free music," as illustrated in Fig. 2. Students with low average grades will expose themselves when they visit the given URL because they are the only ones who can read the message.

While one could avoid this attack by giving separate encryptions for each user of the bulk data, this would greatly increase the overall storage demands, as the contents of each file would need to be replicated for each authorized user. We solve this problem by building efficient *private broadcast encryption* systems that are secure under chosen-ciphertext attacks. Our construction achieves storage space, encryption time, and decryption time comparable to schemes currently employed in encrypted file systems.

The remainder of the paper is organized as follows. We define private broadcast encryption in Sect. 2, giving a game definition of recipient privacy under a

(a) $c_1; c_2; c_3; \{F\}_{K_F}$ (b) $c_1; c_2; c_3; \{F'\}_{K_F}$

Fig. 2. Active attack on recipient privacy. (a) The sensitive document, F, is encrypted for three recipients. If the attacker is a recipient, he or she learns K_F. (b) The malicious document created by an attacker contains F' of the attacker's choice and can be decrypted by the same users as the original document. Recipients of the original document can be discovered by tricking them into decrypting the malicious document.

chosen-ciphertext attack. In Sect. 3, we examine the PGP encryption system and demonstrate attacks against recipient privacy. We present our private broadcast encryption constructions in Sect. 4. Finally, we conclude in Sect. 5.

1.1 Related Work

The notion of key privacy in the public key setting was first formalized by Bellare et. al. [1]. A public key encryption system is key-private if ciphertexts do not leak information about the public keys for which they were encrypted. Specifically, an adversary viewing a chosen message encrypted under one of two public keys is unable to guess (with non-negligible advantage) which public key was used to produce the ciphertext. The authors formalize these definitions for key privacy under chosen-plaintext attack (IK-CPA) and chosen-ciphertext attacks (IK-CCA). They show that ElGamal and Cramer-Shoup are secure under these definitions, respectively, when public keys share a common prime modulus.

Our constructions use a key-private public key system as a component in building private broadcast encryption systems. One interesting observation is that the straightforward construction of a private broadcast encryption scheme using an IK-CCA secure encryption scheme does not result in a private broadcast encryption system resistant to chosen-ciphertext attacks.

Previous work on broadcast encryption has focused on increasing collusion resistance and reducing the length of the ciphertext [2–4]. We differ from these works in that we focus on maintaining the privacy of users, but do not attempt to achieve ciphertext overhead that is sub-linear in the number of users. Whether private broadcast encryption systems can be realized with sub-linear ciphertext overhead is currently an open problem.

2 Private Broadcast Encryption

In this section, we define private broadcast encryption in terms of its correctness and security properties. A private broadcast encryption system consists of four algorithms.

- $I \leftarrow \mathsf{Setup}(\lambda)$. Setup is a randomized algorithm that generates global parameters I for the system from a security parameter λ.

– $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(I)$. Keygen is a randomized algorithm that generates public-private key pairs from the global parameters I.
– $C \leftarrow \mathsf{Encrypt}(S, M)$. Encrypt is a randomized algorithm that generates a ciphertext C for a message M using a set of public keys $S = \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$ generated by $\mathsf{Keygen}(I)$.
– $M \leftarrow \mathsf{Decrypt}(\mathsf{sk}, C)$. Decrypt extracts M from a ciphertext C using a private key sk if the corresponding public key $\mathsf{pk} \in S$, where S is the set used to generate C. Decrypt can also return $\perp$ if $\mathsf{pk} \notin S$ or if C is malformed.

For ElGamal-like systems, the global parameters I simply contain the prime p and generator $g \in \mathbb{Z}_p$. This definition enables each user to generate his or her own public-private key pair individually.

The definition above departs from the standard definition of broadcast encryption in that the standard definition explicitly provides S, the set of recipients, to the Decrypt algorithm. Here we omit this parameter in order to capture systems that hide S. There is no loss of generality, however, as S can be included in the ciphertext, C, directly.

2.1 Recipient Privacy

We define a notion of recipient privacy under a chosen-ciphertext attack for private broadcast encryption systems using a game between a challenger and an adversary. This game ensures that the adversary cannot distinguish a ciphertext intended for one recipient set from a ciphertext intended for another recipient set. To model a chosen-ciphertext attack we allow the adversary to issue decryption queries. More precisely, the game defining recipient privacy of a private broadcast encryption system with n users is as follows:

Init: The challenger runs $I \leftarrow \mathsf{Setup}(\lambda)$ and publishes the global parameters I. The adversary outputs $S_0, S_1 \subseteq \{1, \ldots, n\}$ such that $|S_0| = |S_1|$.

Setup: The challenger generates keys for each potential recipient, $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Keygen}(I)$, and sends to the adversary each pk_i for $i \in S_0 \cup S_1$ as well as each sk_i for $i \in S_0 \cap S_1$.

Phase 1: The adversary makes decryption queries of the form (u, C), where $u \in S_0 \cup S_1$, and the challenger returns the decryption $\mathsf{Decrypt}(\mathsf{sk}_u, C)$. The adversary may repeat this step as desired.

Challenge: The adversary gives the challenger a message M. The challenger picks a random $b \in \{0, 1\}$, runs $C^* \leftarrow \mathsf{Encrypt}(\{\mathsf{pk}_i \mid i \in S_b\}, M)$, and sends ciphertext C^* to the adversary.

Phase 2: The adversary makes more decryption queries, with the restriction that the query ciphertext $C \neq C^*$. The adversary may repeat this step as desired.

Guess: The adversary outputs its guess $b' \in \{0, 1\}$.

We say that the adversary wins the game if $b' = b$.

Definition 1. *A private broadcast encryption system is* (t, q, n, ϵ) *CCA recipient private if, for all* t*-time adversaries* A*, the probability* A *wins the above game using recipient sets of size at most* n *and making at most* q *decryption queries is at most* $1/2 + \epsilon$*.*

Definition 2. *A private broadcast encryption system is* (t, n, ϵ) *CPA recipient private if it is* $(t, 0, n, \epsilon)$ *CCA recipient private.*

In addition to recipient privacy, a secure broadcast encryption scheme must be semantically secure against CCA attacks under the standard definition of semantic security for broadcast encryption systems (see for example [5]).

A standard hybrid argument [6] shows that our definition also implies unlinkability among sets of ciphertexts. We also observe our definition of recipient privacy allows C to leak the number of recipients, just as semantic security allows a ciphertext to leak the length of the plaintext. The number of recipients can be hidden by padding the recipient set to a given size using dummy recipients.

Just as public key encryption is a special case of broadcast encryption, key privacy is a special case of recipient privacy. In key privacy [1], the adversary is restricted to $n = 1$, that is to recipient sets S_0 and S_1 of size 1, mirroring the restriction on the public key Encrypt algorithm to encrypt only for a single recipient. Therefore, the IK-CCA definition is equivalent to our recipient privacy definition with $n = 1$.

3 Broadcast Encryption in Practice

In this section, we make concrete our discussion of privacy problems in broadcast encryption systems by examining broadcast encryption systems used in practice. We study the widely used OpenPGP [7] encryption standard and the GNU Privacy Guard (GPG) [8] implementation as well as discuss the systems used by Windows EFS and by Microsoft Outlook.

3.1 The PGP Encryption System

While OpenPGP is commonly associated with encrypted email, it can be used as a general encryption system. When encrypting a message to multiple recipients, OpenPGP functions as a broadcast encryption system: it encrypts each message under a symmetric key K and then encrypts K to each user using his or her public key. Either ElGamal or RSA encryption can be used for the public key encryption.

Key IDs and Recipient Privacy. In standard operation, GPG completely exposes recipient identities (including blind-carbon-copy recipients). Figure 3 contains a transcript of an attempted GPG decryption of a ciphertext created with a PGP implementation. The ciphertext reveals the key IDs of two recipients. A key's ID is essentially its hash. PGP uses key IDs for two purposes. First, public keys in the Web of Trust are indexed by key ID. For example, the MIT PGP Public Key

Server [9], when queried for a specific name, returns the key IDs, names, and email addresses of the principals with the specified name. A principal's public key can then be retrieved by querying the server by key ID. Second, key IDs are used in ciphertexts to label encryptions of the message key (Fig. 1(a)). These labels speed decryption because the decryptor knows his or her key ID and can locate the encryption of the message key he or she is able to decrypt. Unfortunately, attackers also know key IDs. Moreover, after examining a ciphertext, an attacker need only query a public key server to learn the full name and email address of the owner of the associated public key.

```
C:\gpg>gpg --verbose -d message.txt
gpg: armor header: Version: GnuPG v1.2.2 (MingW32)
gpg: public key is 3CF61C7B
gpg: public key is 028EAE1C
```

Fig. 3. Transcript of an attempted GPG decryption of a file encrypted for two users. The identities of the users are completely exposed by their key IDs. These key IDs can be translated to real identities by a reverse look up on a public key directory.

Throwing Away Key IDs. The OpenPGP standard allows implementations to omit key IDs from ciphertexts by replacing them with zeros (ostensibly to foil traffic analysis [10]). This option is available in GPG using the `--throw-keyids` command line option, but is disabled by default and thus will not be used if the command is not given. Omitting key IDs increases the amount of work required to decrypt a message. A message without key IDs, encrypted to n recipients, contains n unidentified ciphertexts. To decrypt the message, every recipient must attempt to decrypt each ciphertext, performing on average $n/2$ decryption operations.

Even when omitting key IDs, GPG does not achieve recipient privacy. When GPG generates an ElGamal public key, it does so in the group of integers modulo a random prime. Thus, different principals are very likely to have public keys in different groups, making GPG encryptions vulnerable to passive key privacy attacks. These attacks can be directly translated into attacks on CPA recipient privacy. GPG could defend against these attacks by using the same prime for every public key, for example one standardized by NIST [11].

Active Attack. While omitting key IDs and standardizing the group used for public keys achieves CPA recipient privacy, it does not achieve CCA recipient privacy. An active attacker could determine the recipients as follows. Suppose Charlie, the attacker, received the encrypted message $\{K\}_{\mathsf{pk}_A} || \{K\}_{\mathsf{pk}_C} || \{M\}_K$ and wishes to determine whether Alice or Bob was the other recipient. As Charlie possesses his private key sk_C, he can recover K, the message key. He can then encrypt a new message M' for the same recipient as the original message,

$\{K\}_{\mathsf{pk}_A} || \{M'\}_K$, by copying the first portion of the header and encrypting M' under K. When Alice decrypts this message, she obtains M', whereas when Bob decrypts this message, he does not obtain M'.

This active attack is potentially more dangerous than the passive attack in practice. If an attacker wishes to determine a recipient from a large pool of recipients, the passive attack will likely only eliminate some fraction of the possible recipients. An active attack, however, could probe each of the potential recipients individually and learn exactly which ones were recipients of the original message.

3.2 Other Broadcast Systems

Windows EFS. An encrypted file system can be viewed as a broadcast encryption system: the file system itself is the broadcast channel, the files are the messages, and the users who can access a file are the broadcast recipients. The underlying broadcast encryption scheme used in the Windows Encrypted File System (EFS) is vulnerable to privacy attacks. A file in EFS is encrypted under a symmetric key, which in turn is encrypted under the public keys of the users authorized to read the file. These encryptions of the symmetric key are stored in the file header and are usually accessible only to the operating system kernel. An attacker who has physical access to the storage media, for example by duplicating a file server's hard drive or stealing a backup copy of the file system, can learn the list of users authorized to read a file by directly examining the ciphertext component labels.

Outlook. Microsoft Outlook, like many S/MIME clients, is vulnerable to attacks on recipient privacy. When Outlook sends an encrypted email message to multiple recipients, it prepares a single encrypted message and sends copies of that ciphertext to each recipient. Components of the ciphertext are labeled with the issuer and serial number of each recipient's public key certificate. Many certificate authorities, including VeriSign [12], provide a free directory service that translates certificate serial numbers into the certificates themselves. This reveals the identities of all recipients, compromising the privacy of blind-carbon-copy (BCC) recipients. Worse, if a BCC recipient uses a self-signed certificate, Outlook includes his or her full name and email address *in the clear* in the message ciphertext sent to all recipients.

In addition to these passive attacks, active attacks on recipient privacy are particularly easy to mount against encrypted email systems. Each legitimate message recipient can mount an active attack simply by sending a carefully constructed email message to a suspected recipient. Some S/MIME clients avoid these attacks by separately encrypting messages for each recipient. This prevents legitimate recipients both from learning the identities of other message recipients and from learning the number of other recipients. However, sending separate encryptions decreases mail server efficiency and uses more bandwidth.

4 Constructions

In this section, we present two constructions for private broadcast encryption that achieve CCA recipient privacy. The first is a generic construction from any public key encryption scheme that has key indistinguishability under chosen-ciphertext attacks (IK-CCA) [1]. The disadvantage of this first scheme is that decryption time is linear in the number of recipients because the decryption algorithm must try decrypting each ciphertext component until it decrypts successfully. The second construction is a specialized system in which the decryption algorithm performs one asymmetric key operation to locate the appropriate ciphertext component (if one exists). This construction is more efficient for decryptors than the first because no trial decryptions are required. We describe our two schemes and give intuition for their security.

Both constructions require the underlying public key scheme to be *strongly correct*. Essentially, a public key scheme is strongly correct if decrypting a ciphertext encrypted for one key with another key results in $\perp$, the reject symbol, with high probability. While this property is not ensured by the standard public key definitions, most CCA-secure cryptosystems, such as Cramer-Shoup, are strongly correct. Before giving a formal definition of strong correctness, we define a function that generates a random encryption of a given message and then returns the decryption of that ciphertext with a different random key.

$$\begin{aligned}\mathsf{Test}(M): \quad & I \leftarrow \mathsf{Init}(\lambda);\ (\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{Gen}(I);\ C \leftarrow \mathsf{Enc}_{\mathsf{pk}_0}(M);\\ & (\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{Gen}(I);\ \text{Return } \mathsf{Dec}_{\mathsf{sk}_1}(C).\end{aligned}$$

Definition 3. *A public key scheme* $(\mathsf{Init}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* ϵ *strongly correct if, for all* M*, the probability* $\mathsf{Test}(M) \neq \perp$ *is at most* ϵ*.*

4.1 Generic CCA Recipient Private Construction

We realize our first construction by modifying the simple CPA recipient private construction (Fig. 1(b)). Encrypt first generates a random signature and verification key for a one-time, strongly[3] unforgeable signature scheme [13, 14] such as RSA full-domain hash. Then, the encryption algorithm encrypts a ciphertext component containing the generated verification key using a public key encryption scheme that has key-indistinguishability under CCA attacks (IK-CCA). Finally, the algorithm signs the entire ciphertext with the signing key.

The decryption algorithm attempts to decrypt each ciphertext component. If the public key decryption is successful (i.e. returns non-$\perp$), Decrypt will decrypt the message only if the signature verifies under the extracted verification key. Intuitively, an adversary cannot reuse a ciphertext component from the challenge ciphertext in another ciphertext because he or she will be unable to sign the new ciphertext under the same verification key. We now give a formal description of our scheme.

[3] In a strongly unforgeable signature scheme, an adversary cannot output a new signature, even on a previously signed message.

Given a strongly correct, IK-CCA public key scheme $(\mathsf{Init}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, a strongly existentially unforgeable signature scheme $(\mathsf{Sig\text{-}Gen}, \mathsf{Sig}, \mathsf{Ver})$, and semantically secure symmetric key encryption and decryption algorithms (E, D), we construct a private broadcast encryption system as follows.

$\mathsf{Setup}(\lambda)$**:** Return $\mathsf{Init}(\lambda)$.
$\mathsf{Keygen}(I)$**:** Run $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(I)$ and return $(\mathsf{pk}, \mathsf{sk})$.
$\mathsf{Encrypt}(S, M)$**:**

1. $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Sig\text{-}Gen}(\lambda)$.
2. Choose a random symmetric key K.
3. For each $\mathsf{pk} \in S$, $c_{\mathsf{pk}} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathsf{vk}||K)$.
4. Let C_1 be the concatenation of the c_{pk}, in random order.
5. $C_2 \leftarrow E_K(M)$.
6. $\sigma \leftarrow \mathsf{Sig}_{\mathsf{sk}}(C_1||C_2)$.
7. Return the ciphertext $C = \sigma||C_1||C_2$.

$\mathsf{Decrypt}(\mathsf{sk}, C)$**:** Parse C as $\sigma||C_1||C_2$ and $C_1 = c_1||\cdots||c_m$. For each $i \in \{1, \ldots, m\}$:

1. $p \leftarrow \mathsf{Dec}(\mathsf{sk}, c_i)$.
2. If p is $\perp$, then continue to the next i.
3. Otherwise, parse p as $\mathsf{vk}||K$.
4. If $\mathsf{Ver}_{\mathsf{vk}}(C_1||C_2, \sigma)$, return $M = D_K(C_2)$.

If none of the c_i decrypts and verifies, return $\perp$.

Notice the time taken by $\mathsf{Decrypt}$ to execute could leak information. Recipient privacy relies on the attacker being unable to determine whether a decryption fails because $p = \perp$ or because the signature did not verify. Implementations must take care to prevent such timing attacks. We state our main theorem as follows. Due to space constraints, we give the proof in the full version of the paper [15].

Theorem 1. *If* $(\mathsf{Init}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is both* ϵ_1 *strongly correct and* (t, q, ϵ_2) *CCA key private and* $(\mathsf{Sig\text{-}Gen}, \mathsf{Sig}, \mathsf{Ver})$ *is* $(t, 1, \epsilon_3)$ *strongly existentially unforgeable, the above construction is* $(t, q, n, n(\epsilon_1 + \epsilon_2 + \epsilon_3))$ *CCA recipient private.*

The semantic security of our scheme follows in a straightforward manner. Because our scheme achieves broadcast encryption by concatenating public key encryptions, each user can generate his or her own public key and have an authority issue a certificate binding it to his or her identity.

4.2 CCA Recipient Privacy with Efficient Decryption

To decrypt a ciphertext in the CCA recipient private scheme above, a recipient must attempt to decrypt $n/2$ components of the ciphertext, on average, where n is the number of recipients. Non-private schemes improve performance by labeling ciphertext components with recipient identities, directing the attention of decryptors to appropriate ciphertext components. However, these labels reveal the identities of the recipients.

In this section, we construct a private broadcast encryption system that requires only a constant number of cryptographic operations in order to decrypt, regardless of the number of recipients. Our scheme is similar to the previous one with small modifications. Each private key is extended with a random exponent a and each public key is extended with the corresponding value g^a. The encryption algorithm first chooses a random exponent r and labels the ciphertext component for the public key (pk, g^a) with $H(g^{ra})$, where the hash function H is modeled as a random oracle. When decrypting, each user first calculates $H(g^{ra})$ and then uses the result to locate the ciphertext component encrypted for him or her. Users need only perform one public key decryption to recover the message.

The recipient privacy of this scheme relies on a group G in which the computational Diffie-Hellman problem is believed to be hard, but there exists an efficient algorithm for testing Diffie-Hellman tuples (i.e. CDH is hard, but DDH is easy). For example, groups with efficiently computable bilinear maps are widely believed to have this property [16]. The algorithm for deciding DDH is used in the simulation proof, but not in the construction itself. The requirement that DDH be easy can be relaxed, at the cost of a looser reduction.

Let G be a group, with generator g, where CDH is hard and DDH is easy and let $H : G \to \{0,1\}^\lambda$ be a hash function that is modeled as a random oracle (for some security parameter λ). Given a strongly correct, IK-CCA public key scheme $(\mathsf{Init}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, a strongly existentially unforgeable signature scheme $(\mathsf{Sig\text{-}Gen}, \mathsf{Sig}, \mathsf{Ver})$, and semantically secure symmetric key encryption and decryption algorithms (E, D), we construct a private broadcast encryption system as follows.

$\mathsf{Setup}(\lambda)$**:** Return $\mathsf{Init}(\lambda)$.

$\mathsf{Keygen}(I)$**:** Run $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(I)$ and choose a random exponent a. Let $\mathsf{pk}' = (\mathsf{pk}, g^a)$ and $\mathsf{sk}' = (\mathsf{sk}, a)$. Return $(\mathsf{pk}', \mathsf{sk}')$.

$\mathsf{Encrypt}(S, M)$**:**

1. $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Sig\text{-}Gen}(\lambda)$.
2. Choose a random symmetric key K.
3. Choose a random exponent r and set $T = g^r$.
4. For each $(\mathsf{pk}, g^a) \in S$, $c_{\mathsf{pk}} \leftarrow H(g^{ar}) || \mathsf{Enc}_{\mathsf{pk}}(\mathsf{vk} || g^{ar} || K)$.
5. Let C_1 be the concatenation of the c_{pk}, ordered by their values of $H(g^{ar})$.
6. $C_2 \leftarrow E_K(M)$
7. $\sigma \leftarrow \mathsf{Sig}_{\mathsf{sk}}(T || C_1 || C_2)$.
8. Return the ciphertext $C = \sigma || T || C_1 || C_2$.

$\mathsf{Decrypt}((\mathsf{sk}, a), C)$**:** Parse C as $\sigma || T || C_1 || C_2$ and $C_1 = c_1 || \cdots || c_m$.

1. Calculate $l = H(T^a) = H(g^{ar})$.
2. Find c_j such that $c_j = l || c$ for some c, if it exists, else return $\perp$ and stop.
3. Calculate $p \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$.
4. If p is $\perp$, return $\perp$ and stop.
5. Otherwise, parse p as $\mathsf{vk} || x || K$.
6. If $x \neq T^a$, return $\perp$ and stop.
7. If $\mathsf{Ver}_{\mathsf{vk}}(T || C_1 || C_2, \sigma)$, return $M = D_K(C_2)$; otherwise, return $\perp$.

Notice decryption time is independent of the number of recipients. Also, the algorithm for deciding DDH tuples is not used in the construction. The DDH decisions algorithm is used by the simulator in the proof of the following theorem to recognize when the adversary has successfully computed a ciphertext component label (and thus can be used to defeat CDH). Due to space constraints, we give the proof in the full version of the paper [15].

Theorem 2. *If* $(\mathsf{Init}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* ϵ_1 *strongly correct,* (t, q, ϵ_2) *CCA semantically secure and* (t, q, ϵ_3) *CCA key private,* $(\mathsf{Sig\text{-}Gen}, \mathsf{Sig}, \mathsf{Ver})$ *is* $(t, 1, \epsilon_4)$ *strongly existentially unforgeable, CDH is* (t, ϵ_5) *hard in* G*, and DDH is efficiently computable in* G*, then the above construction is* $(t, q, n, n(\epsilon_1 + 2\epsilon_2 + \epsilon_3 + \epsilon_4 + 2\epsilon_5))$ *CCA recipient private.*

4.3 Identity-Based Encryption Extensions

We can extend our constructions to use Identity-Based Encryption (IBE) [17, 18] by using an anonymous IBE [19, 20] scheme. An IBE scheme is called anonymous if a ciphertext does not reveal the identity under which it was encrypted. The Boneh-Franklin scheme [18] has this property, for example. Identity-based encryption is advantageous in our setting where the identities of the recipients should be kept private because an identity-based encryption algorithm need not retrieve a recipient's public key certificate in order to encrypt to him or her. Typical public key systems require encryptors to obtain the public keys of recipients over a network. An eavesdropper could potentially ascertain information about message recipients by monitoring the public keys requested by the encryptor. An IBE encryptor, however, need not make such network requests and avoids this potential vulnerability.

5 Conclusions

In many content distribution applications it is important to protect both the content being distributed and the identities of users allowed to access the content. Currently, encrypted file systems and encrypted email systems fail to protect the privacy of their users. User privacy is compromised because the underlying encryption schemes disclose the identities of a ciphertext's recipients. Many such systems simply give away the identities of the users in the form of labels attached to the ciphertext. Additionally, those systems that attempt to avoid disclosing the recipient's identity, such as GnuPG, are vulnerable to having their user's privacy compromised by a new chosen-ciphertext attack that we introduced.

Our proposed mechanism, private broadcast encryption, enables the efficient encryption of messages to multiple recipients without revealing the identities of message recipients, even to other recipients. We presented two constructions of private broadcast encryption systems. Both of these satisfy a strong definition of recipient privacy under active attacks. The second additionally achieves decryption in a constant number of cryptographic operations, performing comparably to current systems that do not provide user privacy.

References

1. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-privacy in public-key encryption. In: ASIACRYPT '01: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security. Volume 2238 of LNCS., Springer–Verlag (2001) 566–582
2. Fiat, A., Naor, M.: Broadcast encryption. In: CRYPTO '93: Proceedings of Advances in Cryptology. Volume 773 of LNCS., Springer–Verlag (1994) 480–491
3. Naor, M., Pinkas, B.: Efficient trace and revoke schemes. In: Financial cryptography 2000. Volume 1962 of LNCS., Springer–Verlag (2000) 1–20
4. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: CRYPTO '01: Proceedings of Advances in Cryptology. Volume 2139 of LNCS., Springer–Verlag (2001) 41–62
5. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: CRYPTO '05: Proceedings of Advances in Cryptology. Volume 3621 of LNCS., Springer–Verlag (2005) 258–275
6. Bellare, M., Boldyreva, A., Micali, S.: Public-key encryption in a multi-user setting: Security proofs and improvements. In: EUROCRYPT '00: Proceedings of International Conference on the Theory and Application of Cryptographic Techniques. Volume 1807 of LNCS., Springer–Verlag (2000) 259–274
7. OpenPGP: The OpenPGP alliance home page (2005) http://www.openpgp.org/.
8. Werner Koch: The GNU privacy guard (2005) http://www.gnupg.org/.
9. MIT: MIT PGP public key server (2005) http://pgpkeys.mit.edu/.
10. Callas, J., Donnerhacke, L., Finney, H., Thayer, R.: RFC 2440: OpenPGP message format (1998) http://www.ietf.org/rfc/rfc2440.txt.
11. National Institute of Standards and Technology: Digital signature standard (DSS) (2000) http://www.csrc.nist.gov/publications/fips/.
12. VeriSign: Search for digital IDs (2005) https://digitalid.verisign.com/services/client/.
13. Lamport, L.: Constructing digital signatures from a one way function. Technical report, SRI International (1979)
14. Rompel, J.: One-way functions are necessary and sufficient for secure signatures. In: STOC '90: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, ACM Press (1990) 387–394
15. Barth, A., Boneh, D., Waters, B.: Privacy in encrypted content distribution using private broadcast encryption (2006) http://www.adambarth.org/papers/barth-boneh-waters-2006-full.pdf.
16. Joux, A., Nguyen, K.: Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups. Technical Report eprint.iacr.org/2001/003 (2001)
17. Shamir, A.: Identity-based cryptosystems and signature schemes. In: CRYPTO '84: Advances in Cryptology. Volume 196 of LNCS., Springer–Verlag (1985) 47–53
18. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: CRYPTO '01: Proceedings of Advances in Cryptology. Volume 2139 of LNCS., Springer–Verlag (2001) 213–229
19. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: EUROCRYPT '04: Proceedings of International Conference on the Theory and Application of Cryptographic Techniques. Volume 3027 of LNCS., Springer–Verlag (2004) 506–522
20. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. Technical Report eprint.iacr.org/2005/254 (2005)