Efficient broadcast from trapdoor functions

Yi Mu

University of Wollongong, ymu@uow.edu.au

Willy Susilo

University of Wollongong, wsusilo@uow.edu.au

Xinyi Huang

Nanjing Normal University, xh068@uow.edu.au

Publication Details


Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au
Efficient broadcast from trapdoor functions

Abstract
We present a novel scheme of broadcast encryption that is suitable for broadcast servers such as pay TV services. The important feature of our scheme is that the length of a broadcast string in our scheme is independent of the number of receivers in the system; hence it is suitable for large groups. Our scheme is based on a trapdoor encryption technique under the RSA assumption. We also describe a variant of our scheme which provides stronger security.

Disciplines
Physical Sciences and Mathematics

Publication Details

This conference paper is available at Research Online: http://ro.uow.edu.au/infopapers/2834
Efficient Broadcast from Trapdoor Functions

Yi MU a,1, Willy SUSILO a and Xinyi HUANG b

a University of Wollongong
b Nanjing Normal University

Abstract. We present a novel scheme of broadcast encryption that is suitable for broadcast servers such as pay TV services. The important feature of our scheme is that the length of a broadcast string in our scheme is independent of the number of receivers in the system; hence it is suitable for large groups. Our scheme is based on a trapdoor encryption technique under the RSA assumption. We also describe a variant of our scheme which provides stronger security.

Keywords. Broadcast encryption

1. Introduction

Pay TV broadcasting schemes can be related to broadcast encryption, which allows a sender to deliver information to a group of users; each holds a different decryption key. The broadcast encryption was introduced by Fiat and Naor [7]. Since then, there have been a number of schemes in the literature (e.g., [12,8,9,11]). Those schemes vary from bounded to unbounded number of broadcasts. They may be composed of either fixed user groups or variable (or dynamic) user groups. Most broadcast encryption schemes allow a server to deliver information to a set of users that can be dynamically formed. Namely, the broadcaster can determine which users will receive the information with the pre-defined user information. Each authorized user can recover the information by using the corresponding secret key.

A typical Pay TV system consists of a broadcaster and a number of subscribers. The broadcaster broadcasts TV programs to its subscribers. When a Pay TV program is transmitted through an optical fibre or a microwave network, the protection of the program must be enforced against non-subscribers and also the subscribers who want to forge new decryption keys. A pay TV scheme can be achieved with a symmetric-key scheme, where all receivers share the same decryption key. Although it has the advantage of computational efficiency, the key management is often problematic. Public-key schemes in pay TV allow each receiver to hold a different decryption key. Therefore, revocation can be easily done by the broadcaster.

There is a tradeoff between the following two scenarios in broadcast encryption: perfect user revocation and ideal computational efficiency.

1Correspondence to: Yi Mu, School of IT and Computer Science, University of Wollongong, Wollongong NSW 2522, Australia. Tel.: +61 2 4221 5228; Fax: +61 2 4221 4170; E-mail: ymu@uow.edu.au.
To realize Scenario 1, we must assume that the encryption key is dependent on the number of receivers and the size of encryption data is proportional to the number of users. Although the broadcaster can easily add or remove a user, the drawback is obvious: the encryption key and the associated data must be changed whenever a user is added or removed. There are various realizations of Scenario 1 (eg. [4,7,9]). The recent invention of identity-based cryptography has also made identity-based broadcast (broadcast based on identities of users) becomes feasible [3].

Scenario 2 gives us a much more efficient way in handling broadcast encryption, since the encryption key and associated parameters can be kept the same and independent of the number of receivers. This feature even stands when a new user is added to the system. The drawback is due to revocation. It is hard to remove a user from the system. The only realization of the scenario is due to Narayanan et al.[13]. They also tried to sort the revocation problem out by introducing a new parameter for each program; namely, the user receives such a parameter provided he subscribes the specific program. It does not satisfy the ultimate goal of user revocation; that is, any user should be able to be removed from the system by the broadcaster whenever he wants to.

Therefore, there is a tradeoff between Scenario 1 and Scenario 2. If we want to achieve Scenario 1, we have to make a compromise due to computational overhead. On contrast, if we want to achieve Scenario 2, then we will not have a revocation advantage. In this paper, we are not going to find a solution to this tradeoff in which we believe there exists no any desirable solution. Instead, we will propose a new realization of Scenario 2 with a more efficient encryption algorithm and provide a simple and effective revocation scheme that accommodates the basic needs in pay TV broadcast. We also give a variant of our scheme which is secure against IND-CCA2 attacks.

Bellare et al.[2] pointed out the relation between many-to-one trapdoor functions and public-key cryptosystems. They found that many-to-one trapdoor functions can be constructed from public-key cryptosystems. Our interest is different. We are interested in how to construct a public-key scheme from a many-to-one trapdoor function. Our new scheme is based on the trapdoor algorithm that has been widely studied in the literature [5,6,10]. We take advantage of the algorithm to construct the cryptographic keys such that one encryption key maps multiple decryption keys. The security of our scheme is based on the RSA assumption.

The rest of this paper is arranged as follows. In section 2, we will give a set of definitions associated with our schemes and security consideration. Section 3 describes our new scheme based on the trapdoor technique and the security proofs to our scheme. Section 4 presents is a variant of our scheme, which is secure against IND-CCA2 attacks. The final section is our conclusion.

2. Definitions

In this section, we describe the formal definitions of our scheme and give the security definition.

**Definition 1** Our broadcast scheme consists of the following four phases:

- **Setup:** A probabilistic algorithm that on input a security parameter \( l \), outputs definitions of the set of users \( U \), the broadcaster \( B \), the message space \( M \), and the
ciphertext space $C$. Each user in $U$ obtains the associated private key $k_{ij}$ corresponding to the encryption keys $y_j$. For the system with $n$ users and $m$ programs, we have $1 \leq i \leq n$ and $1 \leq j \leq m$. All other parameters are denoted by $\pi$. For revocation purposes, we assume that each user also obtains a symmetric key $\kappa_i$ shared with the broadcaster.

- **Subscribe**: The user $i$ subscribes program $j$ from the broadcaster and obtains a decryption key $k_{ij}$, encrypted with $\kappa_j$.
- **Encrypt**: a probabilistic algorithm that on input $(M, y_j)$, where $M \in M$ and $y_j$ is the encryption key, outputs a ciphertext tuple $(C_0, \rho)$, where $C_0 \in C$ and $\rho$ denotes the remaining parameters.
- **Decrypt**: A deterministic algorithm that on input $(C_0, \rho)$ and a valid decryption key, outputs the message $M$.

Based on the definition above, we will give two concrete schemes. Our schemes are mixed with ElGamal encryption and the RSA assumption, but not incorporating the standard setting for ElGamal encryption. Tsiounis and Yung have given a general study of ElGamal encryption security [15]. They showed that ElGamal encryption scheme is as secure as the Decisional Diffie-Hellman problem. Our scheme differs from this, its security is based on the RSA assumption. We define security in terms of the sense of indistinguishability. Intuitively, if it is infeasible for an adversarial algorithm to distinguish between the encryption of any two messages, even if these messages are given, then the encryption is secure. Our first scheme is not secure against IND-CCA2 [?], but our second scheme is. We allows the attacker to access the decryption oracle even after he has received the challenge (the ciphertext).

**Definition 2** (Security of the first scheme) Let $(\text{Setup}, \text{Encrypt}, \text{Decrypt})$ be an encryption scheme. If we say it is secure in the sense of indistinguishability and intractability of the RSA assumption, then there exists no polynomial-time adversarial oracle $A$ that, on input a ciphertext, outputs the original message.

**Definition 3** (Security of the second scheme) Let $(\text{Setup}, \text{Encrypt}, \text{Decrypt})$ be an encryption scheme. If we say it is secure against IND-CCA2, then there exists no polynomial-time adversarial oracle $A$ that can solve the RSA problem in polynomial time and on input a ciphertext, outputs the original message.

**Definition 4** (Collusion Resistant) Given $\eta$ decryption keys $\{k_i\}$ for $1 \leq i \leq n'$, there exist no polynomial forgers who can collaboratively find a valid decryption key $\chi$ such that $\chi \not\in \{k_i\}_{i=1, \ldots, n'}$.

We consider the scenario that the forged key is not necessarily a key generated by the broadcaster previously. It can be any form as soon as it can be used to decrypt a ciphertext generated by the broadcaster.

### 2.1. Trapdoor Based on the RSA Assumption

In this section, we describe the trapdoor construction, which has been studied in the literature [5,6,10]. We will utilize this trapdoor construction to our new broadcast scheme in the next section.
We consider the RSA setting. Let $N$ be the composition of two safe primes $p$ and $q$. Set $\phi(N) = (p - 1)(q - 1)$. Select an integer $e \in \mathbb{Z}_{\phi(N)}^*$ relatively prime to $\phi(N)$. The trapdoor is defined as follows.

**Definition 5 (Trapdoor)** Given $g \in \mathbb{Z}_{N}^*$ of order $\phi(N)$, there exist a set of integers $(a_i, b_i)$, for $i = 1, \cdots, n$, such that $y = g^{a_i}b_i^e \mod N$ is a constant, where $a_i$ are selected from $\mathbb{Z}_{\phi(N)}$ and $b_i$ are selected from $\mathbb{Z}_{N}^*$.

The number of trapdoors that can be found are dependent on the value of $\phi(N)$. It is trivial to find suitable $(a, b)$ for $a \in \mathbb{Z}_{\phi(N)}$ and $b \in \mathbb{Z}_{N}^*$ that for given $g, y \in \mathbb{Z}_{N}^*$ and $e$ be a prime chosen from $\mathbb{Z}_{\phi(N)}$, $y = g^a b^e \mod N$ forms a trapdoor. We find that given a value $y = g^a b^e \mod N$ along with $g$ and $1/e$, for each value $a' \neq a$, it is trivial to find a unique value $b'$ such that $y = g^{a'}b'^e \mod N$. Note that $b' = (yg^{-a'})^{1/e} \mod N$.

For convenience in the presentation, we will omit modulus if it is clear.

**Theorem 1** If $(c, N, g, e)$ defined above are given, the trapdoor value is $x = g^{1/e}$. [10]

**Proof (Sketch):** We prove that given $x$ along with $g, e$, a trapdoor can be constructed. Given values of $(x, a, b)$, compute $d = x^ab$. The trapdoor can be formed by raising the $e$-roots on both sides. We find $y = d^e = g^a b^e$. We can find another pair $(a', b')$ by randomly selecting a value $a'$ and computing $b' = dx^{-a'}$. Obviously, $y = g^{a'}b'^e$. □

If $e$ is a fixed public value, then the security of the trapdoor is based on the RSA assumption:

**Definition 6 (RSA assumption)** Let $N$ is the composition of two safe primes $p$ and $q$. Set $\phi(N) = (p - 1)(q - 1)$. Let $e$ be an integer relatively prime to $\phi(N)$. Given a random element $s$ selected from $\mathbb{Z}_{N}^*$ and a fixed $e \in \mathbb{Z}_{\phi(N)}$, it is hard to find $x$ such that $x^e = s \mod N$.

We will see that this assumption is sufficient to our scheme, since we do not require $e$ to vary. Most previous applications of this kind of trapdoors are based on the strong RSA assumption where $e$ can be chosen by the attacker.

## 3. The Basic Scheme (BS)

In this section, we describe our new scheme based on the trapdoor discussed above. The basic idea for our construction is to achieve one to many maps by taking advantage of trapdoor.

Before going to the scheme in detail, we briefly describe how the scheme works. There is a broadcast server Bob who broadcasts several programs to the valid subscribers. Any user who wants to get the service must register with Bob first to get a permanent subscription key shared with Bob. Bob possesses a set of broadcast encryption keys, one for each program. The subscription keys are used to deliver the program keys to the users who have subscribed the program, respectively. These keys can be used for revocation, namely, Bob can refuse to send the user further program keys if the user has not paid. For a program, each user holds a different key, all map to the program encryption key.
The scheme is given as follows in terms of Setup, Encrypt, Subscribe, and Decrypt phases as defined previously in this paper.

Setup: The broadcaster sets the system up by selecting two large primes $p, q$, setting $N = pq$ and $\phi(N) = (p-1)(q-1)$. He also finds a number $e, \theta \in \mathbb{Z}_\phi(N)$ relatively prime to $\phi(N)$, a public generator, $g \in \mathbb{Z}_N^*$, and sets $M = C = XN$. Each user obtains a pair of secret key from the broadcaster. For User $i$, the secret key pair denoted by $fei = (a_i, b_i)$, where $a_i \in \mathbb{Z}_\phi(N)$ and $b_i = b_0^e$. All $\{a_i, b_i\}$ maps to a single value of $y_j$ for program $j$. In other words, for a fixed $y_j$, we have $y_j = g^{a_i}b_i^e \mod N, i = 1, \ldots, n$. which can be constructed as follows: select $a_i \in \mathbb{Z}_\phi(N)$ and then compute $b_i = (y_jg^{-a_i})^{1/e}$. The public parameter is $N$ only. We have assumed that $y_j$ is associated with a single problem in the Pay TV system. For a multi-program system, a suitable $y_j$ is selected for each program.

Subscribe: A user wishes to subscribe a program or programs, he or she should register with the broadcaster and obtains a permanent subscription key $\kappa_i$, which is a symmetric key such as an AES key. This key is used to encrypt the corresponding program key which is then sent the subscriber. For example, if user $i$ has subscribed program $j$, then encrypted $(a_i, b_i)$ along with the related parameters are sent to user $i$.

Encrypt: To broadcast a message $M$ of program $j$, the broadcaster selects a random $r \in \mathbb{Z}_N$, computes the broadcast triplet $(My_j, g^r, r/\theta)$, where $M \in \mathbb{M}, g \in \mathbb{Z}_N^*$, and $e \in \mathbb{Z}_\phi(n)$. Then, the triplet $(A, B, C)$ is broadcasted to all users.

Decrypt: Upon receiving the broadcast triplet $(A, B, C)$, any user who has previously subscribed program $j$ can retrieve $M_j$ by computing $A(Ba_kCk)^{-1} = M_j$ for $(a_k, b_k) \in \{a_i, b_i\}_{i=1,\ldots,n}$.

The correctness of the scheme is obvious: all users who hold a valid program decryption key can retrieve the program. However, it has to be sound, i.e., only legitimate subscribers can retrieve the program. We discuss this issue in the next section.

3.1. Security of BS

We consider security in our scheme as two aspects:

- Attacks from outsiders who have not got a valid decryption key and attempts to find a valid decryption key that can be used to decrypt any broadcasted ciphertext in the corresponding program.

- Attacks from insiders, each has got a valid decryption key and attempts to find another valid decryption key by collusion.

For outsider attacks, we consider the security in our scheme in the sense of indistinguishability [15]. We can refer our encryption scheme (omit the Subscribe phase) to as a variation of ElGamal encryption. Following the definition of indistinguishability [15], we have the following definition.

**Definition 7** (Indistinguishability) An encryption scheme (Setup, Encrypt, Decrypt) is said to be secure in the sense of indistinguishability, if, for every polynomial time
algorithm \( F \) for every probabilistic polynomial time algorithm \( A \), for every constant \( \tau > 0 \) and for every sufficiently large \( \ell \),

\[
\Pr \left[ F(1^\ell) = (\alpha, \beta, \gamma) \text{ s.t. } \Omega(\alpha, \beta, \gamma) > \frac{1}{\ell^\tau} \right] < \frac{1}{\ell^\tau}.
\]

\[
\Omega(\alpha, \beta, \gamma) = \left| \Pr[A(\gamma, \text{Encrypt}_{\text{setup}}(1^\ell)(\alpha) = 1] - \Pr[A(\gamma, \text{Encrypt}_{\text{setup}}(1^\ell)(\beta) = 1] \right|.
\]

Here, \( \alpha, \beta \in M \) and \( \gamma \) is a polynomial random variable.

### 3.1.1. Security against Outsiders.

Although our scheme is a variation of the ElGamal encryption scheme, the security of our scheme is not based on the decision Diffie-Hellman problem but the RSA problem.

Let us take a look at why it is not based on the DDH problem. Given a valid ciphertext triplet \((My^r, g^r, r/e)\), the associated DDH triplet should be \((g^r, y = g^\phi, g^\gamma)\). That is, given \(g^r, g^\phi\), decide if \(x = \phi r\). However, in our scheme, \(y\) is not public.

**Theorem 2** If BS is not secure in the sense of indistinguishability, then there exists a probabilistic polynomial time adversary that can solve the RSA problem with overwhelming probability.

**Proof:** If our scheme is not secure in the sense of indistinguishability it suffices to show that we can find with non-negligible probability a pair of plaintext messages such that their encryption can be distinguished with non-negligible probability of success.

We first show that given a valid encryption triplet \((A, B, C)\) and the public information \(N\), the security of our scheme can be reduced to a RSA problem. Observe \(A = MB^a b^C\). The adversary chooses random \(a'\) and computes \(A(B^{a'})^{-1}\) which should give the equality \(A(B^{a'})^{-1} = b^C\) if the RSA problem can be solved and \(b'\) can be found in polynomial time.

Assume there exists a RSA oracle. The game for the RSA assumption is as follow:

**Game 1:** Given \((A, B, C)\) and \(N\),

G1-1: Select random \(a' > 1\).
G1-2: Compute \(s = A(B^{a'})^{-1}\).
G1-3: Select random \(b' \in Z_N^*\).
G1-4: Test \(b'^C \overset{?}{=} s\). If yes, output 1, otherwise 0.

The adversary plays Game 1 and asks \(q_1\) queries to the RSA oracle. The probability of success is \(q_1/2^{2\ell}\).

We then show that if the RSA oracle outputs 1, the adversary can distinguish the ciphertext for messages \(m_0, m_1\). Our adversarial algorithm selects random \(m_0, m_1 \in Z_N^*\). Then given the encryption of these messages:

\[
(m_i y^{r_0}, g^{r_0}, r_0/\theta) \leftarrow \text{Encrypt}(m_i),
\]

\[
(m_{1-i} y^{r_1}, g^{r_1}, r_1/\theta) \leftarrow \text{Encrypt}(m_{1-i}), i \in_R \{0, 1\},
\]
where \( r_0, r_1 \in \mathbb{Z}_\phi(N) \), we only need to show given that the RSA oracle outputs 1, the adversary can distinguish non-negligibly better than random guessing which ciphertext encrypts which message (find \( i \)). The success probability of random guess is \( \frac{1}{2} + \epsilon \) for a small number \( \epsilon \).

If the adversary can solve the RSA problem, then he can output: \( (a', b'), y^{r_0} = B^{a'}b'^C \), and

\[
\begin{align*}
    m_i y^{r_0} / m_0 &= \begin{cases} y^{r_0} & (i = 0) \\ m_i y^{r_0} / m_0 & (i = 1) \end{cases}
\end{align*}
\]

In this instance, the adversary is sure that the first ciphertext encrypts the first message with probability non-negligibly better than random guessing. Of course, if the RSA problem is intractable or the RSA oracle outputs 0, then the adversary cannot determine \( y^{r_0} \). He can only randomly pick \( i \). The advantage of the adversary is \( \Pr[\text{win}] = \frac{1}{2} - \epsilon \).

### 3.1.2. Security against Insiders.

We consider the scenario that several valid users collude to find a valid decryption key which is not one of keys they currently hold. We will show that a collusion will not give the adversaries only advantage in gaining a new pair of decryption key.

**Definition 8 (Insider Attacks) Given a set of decryption keys \( \{k_i\}_{i \in \mathcal{F}} \subseteq \{k_j\}_{j = 1, \ldots, n} \), where \( \mathcal{F} \) is the set of indices for the set of forgers in \( \mathcal{U}_s \) which denotes a set of legal subscribers, find a new decryption key \( k_f \) such that \( f \notin \mathcal{F} \), where \( k_f \) is a valid key that decrypts all messages belonging to the same program.

**Remark:** Since \( r \) is unique to the ciphertext in the program, a successfully forged decryption key must be independent to \( r \). For example, we consider the following case to be a unsuccessful forgery. Given a valid key \( k = (a, b) \) and a valid ciphertext \( (A, B, C) \), we select \( a' \) at random and try to find the corresponding \( b'^C \) from \( b'^C = B^{a' - a}b'^C \). Although \( b'^C \) along with \( a' \) can also be used to decrypt \( (A, B, C) \), it is dependent on \( r \) and cannot be used to decrypt other ciphertexts in the program.

**Theorem 3** Our scheme is secure against collusion attacks from insiders if the RSA problem is intractable.

**Proof (Sketch):** Observe that a user does not learn \( (g, b_i, e) \), which prevents him from finding the trapdoor value \( x = g^{1/e} \). It is trivial to see that if the these values are known to two users, \( x \) can then be found: Assuming there are four forgers; each possesses a valid decryption key, \( (a_i, b_i) \), for \( i = 1, \ldots, 4 \). Given the first and second ones, we find they have the relation: \( b_2/b_1 = (g^{a_1 - a_2})^{1/e} \). Similarly, for the third and four ones, we have \( b_4/b_3 = (g^{a_3 - a_4})^{1/e} \). The forgers can then try to find two suitable integers \( \alpha \) and \( \beta \) such that \( \alpha (a_1 - a_2) + \beta (a_3 - a_4) = 1 \). With the resulting \( \alpha \) and \( \beta \), they can compute \( (b_2/b_1)^\alpha (b_4/b_3)^\beta = g^{1/e} = x \). Once \( x \) is found, they can compute other decryption keys.

Therefore, in our scheme \( (g, b_i, e) \) are not given to the users. With a similar attack to the above, at best the forgers can compute \( (b_2/b_1)^C \) and \( (b_4/b_3)^C \) which are equal to \( g^{r(a_1 - a_2)} \) and \( g^{r(a_3 - a_4)} \), respectively. Using the same approach, they obtain a pair of \( (\alpha, \beta) \) and thus \( g^r \). Since \( g^r \) is public, they gain nothing from it.
4. The Scheme with IND-CCA2 Security

Soldera et al. [14] proposed an encryption scheme that is claimed having IND-CCA2 security under the DDH assumption. His scheme is a variant of Zheng-Seberry scheme [16], which is believed insecure under IND-CCA2 [14]. The reader is referred to [1] for more information. We have shown that our scheme is not based on the DDH but the RSA assumption. However, their algorithm can be converted into our scheme, although the security assumption is different.

The Setup and Subscribe of the scheme are the same. Here, we give the Encrypt and Decrypt phases:

Encrypt: Select random \( r, \theta \in \mathbb{Z}_p(N) \), compute \( u = y^r \), \( v = H(M\|u) \), \( w = M\|v \), \( A = uw \), \( B = g^r \), \( C = r\theta \). The resulting ciphertext is \((A, B, C)\).

Decrypt: Compute \( A(B^{\alpha}C^{-1} - 1) = M\|v' = w' \), and then verify the decryption by checking \( H(M\|\alpha) = v' \).

Theorem 4 Our scheme is secure against the CCA2 attacks in the random oracle model assuming that the RSA assumption is intractable.

Proof (Sketch): There exists a simulator that simulates the encryption oracle as follows. The simulator is different from the "standard" one which can randomly pick an encryption key. We assume that the encryption key is fixed to suit our scheme better. There exits a decryption oracle. The adversary can send any ciphertext, which might not be necessarily from the simulator, to the decryption oracle and obtain a pair \((y_i, M_i)\).

On input two random messages \( M_0, M_1 \in \mathbb{Z}_N^* \), the simulator outputs the encryption of these messages:

\[
(u_0y_0r_0, g_0r_0, r_0/\theta, u_0 = y_0r_0, v_0 = H(M_0\|u_0), w_0 = M_0\|v_0) \leftarrow \text{Encrypt}(m_i),
\]

\[
(u_1y_1r_1, g_1r_1, r_1/\theta, u_1 = y_1r_1, v_1 = H(M_1\|u_1), w_1 = M_1\|v_1) \leftarrow \text{Encrypt}(M_{1-i}),
\]

\( i \in_R \{0, 1\} \),

where \( r_0, r_1 \in_R \mathbb{Z}_p(N) \). If the RSA assumption is intractable, then the adversary picks \( i \) randomly. The advantage of the adversary is \( \text{Pr}(\text{win}) - \frac{1}{2} \).

The ciphertext strings are sent to the decryption oracle. If the RSA assumption is trackable to the adversary, the adversarial algorithm outputs a forgery: \((a', b'), y_0r_0 = Ba'c'\), and if the adversary chooses \( i = 0 \), then

\[
u_iy_0r_0 = \begin{cases} y_0r_0 & (i = 0) \\ u_1y_0r_0 & (i = 1) \end{cases}
\]

The result is distinguishable to the adversary who is sure that the first ciphertext encrypts the first message with probability non-negligibly better than random guessing, because \( y_0r_0 \) is associated with the forged key \((a', b')\). Of course, if the RSA problem is intractable, then the adversary cannot determine \( y_0r_0 \). □

5. Conclusion

We propose an efficient broadcast encryption scheme based on the many trapdoor function by the assumption that the RSA problem is not solvable in polynomial time. Our
scheme achieves the same goals given in the original Pay TV paper but is more efficient.
We also described a variant of our scheme, which provides IND-CCA2 security.

References


