A Secure and Efficient Key Escrow Protocol for Mobile Communications

Byung-Rae Lee, Kyung-Ah Chang, and Tai-Yun Kim

Department of Computer Science and Engineering, Korea University,
1, 5-ga, Anam-dong, Sungbuk-ku, Seoul, 136-701, Korea
{brlee, gypsy93, tykim}@netlab.korea.ac.kr

Abstract. In this paper we present secure and efficient key escrow protocols that guarantees escrow secrecy, public verifiability, and robustness for mobile telecommunications systems. We present a new construction for key escrow scheme, which compared to previous solutions by Chen, Gollman, Mitchell and later by Martin, achieves improvements in efficiency and security. We proposed a new key escrow protocol, designed for the case where the pair of communicating users are in different domains, in which the pair of users and all the third parties jointly generate a session key for end-to-end encryption.

1 Introduction

In modern secure telecommunications systems there are likely to be two contradictory requirements. On the one hand users want to communicate securely with other users, and on the other hand governments have requirements to intercept user traffic in order to combat crime and protect national security. A key escrow system is designed to meet the needs of both users and governments, where a cryptographic key for user communications is escrowed with an escrow authority (or a set of authorities) and later delivered government agencies when lawfully authorized.

When users communicate internationally, there is a potential requirement to provide the law enforcement agencies of all the relevant countries, e.g. the originating and destination countries for the communication, with warranted access to the user traffic. For example, mobile telecommunications systems might provide an end-to-end confidentiality service to two users in two different countries, and law enforcement agencies in both these countries might independently wish to intercept these communications.

In this paper we suppose that, in some environments where international key escrow is required, Trusted Third Parties (TTPs, acting as escrow authorties) may not be trusted individually to provide proper contributions to an escrowed key and to

V.N. Alexandrov et al. (Eds.): ICCS 2001, LNCS 2073, pp. 433-443, 2001.

reveal the key legally, and users also may not be trusted to provide proper contributions to an escrowed key. We refer to domains instead of countries throughout.

Requirements for key escrow in an international (i.e. a multi-domain) context has been described by Chen, Gollman and Mitchell [1]:

1. No domain can individually control the generation of an escrowed key, and hence the escrowed key cannot be chosen by entities in only one domain and then transferred to the other domain.
2. The interception authorities in any domain can gain access to an escrowed key without communicating with any other domain, i.e. the key has to be capable of being escrowed in all relevant domains independently.
3. The entities in any domain can ensure the correctness and freshness of the escrowed key.

In this paper we present secure and efficient key escrow protocols for mobile telecommunications systems. Major achievements of our protocols are public verifiability on the correctness in recoverability of the session key and robustness. In our scheme the faulty behavior of any reasonably sized coalition of TTPs can be tolerated. The user just sends a particular ElGamal encryption [2,3] of the private key plus a proof that it indeed is a valid encryption of a discrete logarithm of the public key. In our scheme user needs to communicate only $O(k)$ bits and to perform the work $O(k)$. Moreover, the work of a TTP is also $O(k)$.

Our new key escrow protocols work in the model set forth by Chen et al. [1], where two entities A and B, located in mutually mistrusting domains, establish a shared session key based on Diffie-Hellman key exchange.

In our model, each user escrows his private key by posting an ElGamal encryption to the bulletin board. The encryption does not reveal any information on the private key itself but it is ensured by a proof of knowledge that the encryption indeed contains a valid private key. Due to the verifiable encryption technique and the bulletin board communication model, the recoverability of the user's private key can be verified by observer. This ensures public verifiability.

To prevent deviant users from obtaining a 'shadow-key', users and third parties jointly generate an escrowed key.

1.1 Properties for Our Key Escrow Protocols

Let us state and discuss the properties of the key escrow considered in this paper. Escrow secrecy and robustness come from the robust threshold cryptosystem [5,6,7]. Public verifiability comes from the ElGamal encryption and the proof of knowledge of Chaum and Pedersen [8]. Other two properties achieved from the joint generation of the session key.

Escrow Secrecy For any group of less than t TTPs it must be infeasible to recover the session key.

Public Verifiability Everybody, e.g. any network provider, can verify the correctness in recoverability of the session key.

Robustness The faulty behavior of any reasonably sized coalition of TTPs can be tolerated. In key escrow protocols this includes that no user can disrupt the key escrow scheme; in other words, any cheating user can be detected and discarded.

Resistance to Shadow Public Key Attack An escrowed key in the proposed protocols is a function of contributions from the user and all relevant TTPs. This property prevents two users abusing the mechanism by using the 'shadow-public-key' attack proposed by Kilian and Leighton [9].

Session Key Freshness If any entity updates its contribution using a fresh and random number, the key will be fresh and random. Any entity, which has updated his contribution to the session key can verify the freshness of the key.

1.2 Previous Work

The proposed protocols build directly upon international key escrow protocols of Chen, Gollman, Mitchell [1] and Martin [4], where a verifiable secret sharing (VSS) scheme used as an important primitive. In their protocol, each user A and B, located in mutually mistrusting domains, distributes shares of the private key to the TTPs using VSS respectively. A general problem with such a solution is that TTPs can verify the validity of only their own shares, but they cannot know whether other TTPs have also received valid shares. It may open the possibility for disputes: On the one hand a dishonest user may just skip sending a message to a TTP, while on the other hand a dishonest TTP may claim not to have received a message.

Their protocol lacks of explicit mechanisms to prevent malicious TTPs, e.g, TTPs submitting incorrect shares during the recovery procedure. This problem is usually dealt with implicitly though. In the schemes of [1,4] it suffices that the TTPs simply releases their shares. Subsequently the released shares may be verified by anybody against the output of the distribution protocol.

In their scheme major communication costs are generation of shares of a secret sharing scheme, and communication between users and their home TTPs and between TTPs from different domains. For security parameter k, each user has to perform $O(nk)$ modular multiplication and the TTPs also have to perform $O(nk)$ modular multiplication to transfer secrets to the other set of TTPs in a different domain using VSS.

Recently, Young and Yung [10] presented the scenario and solution for software key escrow. In their approach each user generates its own key pair, and registers with a certification authority. The user must also encrypt shares of the secret key for a specified group of trustees. The key pair is only accepted, if the user also provides a proof that the encrypted shares indeed correspond to the registered public key. Wenbo [11] described a partial key escrow scheme based on Stadler's verifiable encryption [12] and the robust threshold cryptosystem [7].

1.3 Our Contributions

The main contribution of this paper is a simple and fair international key escrow protocol based on robust threshold cryptosystem [5,6,7] and publicly verifiable encryption. To this end, we will employ fault-tolerant threshold cryptosystems instead of verifiable secret sharing scheme. In our scheme there will be only one public key for which the matching private key is shared among the TTPs using threshold cryptography techniques. Each user gives to the home TTPs a single escrowed encryption of the private key and decryption must be carried out in threshold collectively by a set of TTPs. Unlike previous schemes based on Chen's approach, however we will achieve robustness w.r.t. The correctness of the decryption will be assured, even in the presence of malicious TTPs. To achieve robustness against faulty TTPs we used the proof of knowledge of Chaum and Pedersen [8].

Recall that [1,4] requires the availability of private channels from the user to each of the home TTPs individually. However, communication over the private channels is clearly not publicly verifiable. We replaced the private channels by the ElGamal public key encryption. In our scheme, Each user sends an ElGamal encryption of the private key and a proof that TTPs can recover the private key. This is achieved via applying an ElGamal encryption with the proof of knowledge. The proof prevents the users from casting invalid encryption, and should be such that no information whatsoever leaks about the actual private key contained in an encryption.

Another contribution of this paper is a fair key escrow scheme in which the complexity of the user's protocol is linear in the security parameter k. This comprises the computational as well as the communication complexity (in bits). Each user needs to communicate only $O(k)$ bits and to perform $O(k)$ modular multiplication. Moreover, the dominating factor for the work of a TTP is k. Compared to the scheme of [1,4], we thus achieve a reduction of the work for each participant by a factor of n.

A session key for end-to-end encryption is established based on Diffie-Hellman key exchange. An escrowed key is a function of contributions from the user and all relevant TTPs to prevents two users abusing the mechanism by using the 'shadow-public-key' attack as shown in [1].

The main scheme presented in the paper is based on the security of the ElGamal encryption scheme (which is related to the difficulty of the Decision Diffie-Hellman problem).

1.4 Organization of the Paper

The remainder of the paper is subdivided as follows. In Section 2, verifiable encryption techniques are described. In Section 3, We explain the robust threshold cryptosystem. We propose a new key escrow protocol in Section 4. In our protocol two sets of TTPs, one group in each domain, are used as escrow authorities for two users. In Section 5, we conclude by giving the brief review on proposed protocols.

2 Verifiable Encryption

In this section, we will describe a verifiable encryption technique based on the ElGamal encryption and the proof of knowledge by Chaum and Pedersen.

2.1 Bulletin Board

The communication model required for our key escrow protocol is a public broadcast channel with memory, which is called a bulletin board. All communication through the bulletin board is public and can be read by any party (including passive observers). No party can erase any information form the bulletin board, but each active participant can append message to its own designated section.

2.2 Proofs of Knowledge for Equality of Discrete Logarithms

We will use the protocol by Chaum and Pedersen [8] as a subprotocol to prove that $\log_g x = \log_h y$, where by a prover shows possession of an Z_q satisfying $x = g$ and $y = h$.

1. The prover sends $a = g^w$ and $b = h^w$ to the verifier, with $w \ _R \ _q$.
2. The verifier sends a random challenge $c \ _R \ _q$ to the prover.
3. The prover responds with $r = w \quad c$
4. The verifier checks that $a = g^r x^c$ and $b = h^r y^c$.

It is well known that the above protocol is zero-knowledge only against the honest verifier. In order to make the protocol non-interactive, the verifier will be implemented using the Fiat-Shamir heuristic [13] which requires a hash function. In this case security is obtained for the random oracle model.

2.3 Double Exponentiation and Double Discrete Logarithms

Let p be a large prime so that $q = (p \quad 1)/2$ is also prime, and let $h \quad _p$ be an element of order q. Let further G be a group of order p, and let g be a generator of G so that computing discrete logarithms to the base g is difficult.

Our scheme will make use of double exponentiation. By double exponentiation with bases g and h we mean the function

$$_p \quad G : x \quad g^{(h^x)}$$

By the double discrete logarithm of g to the bases g and h we mean the unique $x \quad _q$ with

$$y = g^{(h^x)}$$

if such an x exists.

2.4 Verifiable Encryption of Discrete Logarithms

Our key escrow scheme is identical ElGamal's public key system [2], which is a variation of the Diffie-Hellman key-exchange protocol [3].

First, each TTP randomly chooses a secret key $z \quad _q$ and publishes his public-key $y = h^z \pmod p$. To encrypt a message $m \quad _p^*$ with the public-key y, the user randomly chooses $\quad _q$ and calculates the pair

$$\left(h \quad , y \quad m \quad ^1\right) \pmod p .$$

The ciphertext (A, B) can be decrypted by the recipient by calculating

$$m = A^z / B \pmod p .$$

Let us now describe a protocol for verifying that a pair (A, B) encrypts the discrete logarithm of a public element $P = g^s$ of the group G. It is based on the fact that if (A, B) is equal to $\left(h \quad , y \quad s \quad ^1\right) \pmod p$ for any $\quad _q$ then

$$P^B = g^{sB} = g^{(y \quad)} .$$

By the discrete logarithm of $g^{\{y\}}$ to the base g, we can now immediately obtain an efficient proof of knowledge for the following relation:

$$\log_h A = \log_y \left(\log_g P^B\right).$$

To prove that a pair (A, B) encrypts the discrete logarithm of a public key P, we have the efficient proof of knowledge by Chaum and Pedersen, described above.

3 Robust Threshold ElGamal Cryptosystem

The robust threshold cryptosystems described here is a slight variation from [5,6,7]. The main protocols of a threshold system consists of a key generation protocol to generate the private key jointly by the receivers, and decryption protocol to jointly decrypt a ciphertext without explicitly reconstructing the private key.

Key Generation The TTPs will execute a key generation protocol due to Pedersen [5] or rather improvement by [14]. The result of the key generation protocol is that each TTP T_i will possess a share $z_i \quad Z_q$ of a secret z. The TTPs are committed to these shares as the values $y_i = h^{z_i}$ are made public. Furthermore, the shares z_i are such that the secret z can be reconstructed from any set of z shares using appropriate Lagrange coefficients, say:

$$z = \sum_i z_i \, \lambda_{i,\ldots,} \quad , \quad \lambda_{i,\ldots,} = \prod_{l \setminus \{i\}} \frac{l}{l-i}.$$

The public key $y = h^z$ is announced to all participants in the system.

Decryption To decrypt a ciphertext $(A, B) = \left(h \quad , y \quad m^{-1}\right)$ without reconstructing the secret z, each TTP broadcasts $w_i = A^{z_i}$ and proves in zero-knowledge that

$$\log_h y_i = \log_A w_i$$

as described in Section 2.2. Let denote any subset of t TTPs who passed the zero-knowledge proof. Now the plaintext can be recovered as

$$m = \prod_i w_i^{i,} \Big/ B .$$

Note that no single participant learns the secret z, and that the value of z is only computationally protected. The above protocol assures that the decryption is correct and successful even if up to $n \quad t$ TTPs are malicious or fail to execute the protocol.

4 The Key Escrow Protocol

Given the primitives of the previous section we now assemble a simple and efficient key escrow protocol, where two sets of TTPs, one group in each domain, are used as multiple authentication servers for the users and key escrow agencies for the interception agencies.

In this protocol, A and B are users in different domains. There are n TTPs $T_1,\cdots,T_n$ working for A as escrow authorities (in A's domain), and l TTPs $U_1,\cdots,U_l$ working for B as escrow authorities (in B's domain). Users and interception agencies do not communicate with TTPs outside their domain. Each set of home TTPs has agreed a common secret key (K_T and K_U respectively) and a key generation function f. This function takes as input the identity of two users, a clock C and a secret TTP key, and outputs a random secret. Let $f(A,B,C,K_T) = s_T$ and $f(A,B,C,K_U) = s_U$. We assume that clocks are synchronized among the set of TTPs. Before the protocol starts users do not share any secret.

In the following protocol, two sets of TTPs $T_1,\cdots,T_n$ and $U_1,\cdots,U_l$ assist two users A and B respectively to establish a session key K_{AB}. Each set of third parties escrow the key collectively.

Initialization As part of the initialization the designated parties generate the system parameters p,q,g,G as described in Section 2.3. Two sets of TTPs $T_1,\cdots,T_n$ and $U_1,\cdots,U_l$ will execute a key generation protocol respectively as described in Section 3.

Key Establishment The protocol consists of the following steps:

1. A chooses a random private key s_A, generates an encryption (x_A, y_A) of s_A using the public key of T_i, $1 \quad i \quad n$ accompanied by a proof of knowledge. She also publishes the public key $P_A \ (= g^{s_A})$.
2. B chooses a random private key s_B, generates an encryption (x_B, y_B) of s_B using the public key of U_j, $1 \quad j \quad l$ accompanied by a proof of knowledge. She also publishes the public key $P_B \ (= g^{s_B})$.
3. T_i, $1 \quad i \quad n$ verifies the encryption (x_A, y_A). T_i, $1 \quad i \quad n$ generates a secret s_T, calculates $P_{AT} \ (= P_A^{s_T})$, and sends P_{AT} to U_j, $1 \quad j \quad l$.

4. U_j, $1 \quad j \quad l$ verifies the encryption (x_B, y_B). U_j, $1 \quad j \quad l$ generates a secret s_U, calculates $P_{BU} \ (= P_B{}^{s_U})$, and sends P_{BU} to T_i, $1 \quad i \quad n$.
5. T_i, $1 \quad i \quad n$ calculates $P_{BUT} \ (= P_{BU}{}^{s_T})$ and sends P_{BUT} to A.
6. U_j, $1 \quad j \quad l$ calculates $P_{ATU} \ (= P_{AT}{}^{s_U})$ and sends P_{ATU} to B.
7. Finally, A and B separately compute a session key as:

$$K_{AB} = (P_{BUT})^{s_A} = (P_{ATU})^{s_B} = g^{s_A s_T s_U s_B}.$$

Key Recovery Any set of t TTPs in each domain can recover the session key separately:

1. T_i, $1 \quad i \quad n$ jointly execute the decryption protocol as described in Section 3 for (x_A, y_A) to obtain s_A. T_i, $1 \quad i \quad n$ can recover the session key K_{AB} with the knowledge of P_{BUT} and s_A.
2. U_j, $1 \quad j \quad l$ jointly execute the decryption protocol as described in Section 3 for (x_B, y_B) to obtain s_B. U_j, $1 \quad j \quad l$ can recover the session key K_{AB} with the knowledge of P_{ATU} and s_B

In this protocol any set of t TTPs in each domain can compute the session key K_{AB} established between A and B, but no group of $t \quad 1$ or less TTPs can recover the session key. The fact that all entities are involved in the key generation process helps make it more difficult for deviant users to subvert the escrowed key by using a hidden 'shadow-key'. In addition, no third party can force A or B to accept a wrong message unless all the third parties are colluding.

The performance of the protocol is as follows. The work for an user is clearly linear in k, independent of the number of TTPs. The work for the TTP is $O(nk+k)$ for T_i, $1 \quad i \quad n$ and $O(lk+k)$ for U_j, $1 \quad j \quad l$ respectively. Since key generation protocol is an one time operation of system setup procedure, the work for the TTPs is actually $O(k)$.

Our result can be summarized in the following theorem.

Theorem 1 If the ElGamal cryptosystem is semantically secure, then our key escrow protocol provides escrow secrecy, public verifiability, robustness and resistance to shadow-public-key attack.

Proof Escrow secrecy is guaranteed by the security of the ElGamal cryptosystem used to encrypt the private key. This is true because we assume that no more than $t \quad 1$ TTPs conspire, since t TTPs can reconstruct the private key used in the scheme. Any group of at least t TTPs can compute the user's private key (by the properties of the threshold ElGamal cryptosystem discussed in Section 3 above).

Public verifiability is achieved because any observer can check the proof of knowledge for the encryption, since those are made non-interactive.

Robustness with respect malicious users is achieved by means of the verifiable encryption technique, which ensures that users cannot submit bogus encryption. Ro-

bustness with respect to at most $n - t$ malicious authorities is inherited from the robustness of the key generation and decryption protocols.

Finally, shadow-public-key attack is prevented by joint generation of the session key. □

For the non-interactive version of the scheme based on the Fiat-Shamir heuristic, the result holds in the random oracle model.

5 Concluding Remarks

We have shown secure and efficient key escrow protocols for mobile communications based on the robust threshold cryptosystem and publicly verifiable encryption. In our scheme the work for the user is minimal and independent of the number of TTPs. The work for the TTPs is reduced accordingly. Another important difference is that due to the use of a threshold cryptosystem we achieve robustness in a strong sense. Proposed escrowed key agreement protocol, which meets possible requirements for international key escrow, where different domains do not trust each other.

We must note that it is difficult for any key escrow system to force two users to use only the current escrow key for their end-to-end encryption if the users share a secret or can use their own security system.

References

1. L. Chen, D. Gollmann, and C.J. Mitchell, "Key escrow in mutually mistrusting domain," Cambridge Workshop on Security Protocols, LNCS, vol.1189, pp.139-153, Springer-Verlag, 1997.
2. T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions. on Information Theory, vol.IT-31, no.4, pp.469-472, 1985.
3. W. Diffie, and M. Hellman, "New Directions in Cryptography," IEEE Transactions on Information Theory, November, 1976.
4. K. M. Martin, "Increasing efficiency of international key escrow in mutually mistrusting domains," 6th IMA Conference on Cryptography and Coding, LNCS, vol.1355, pp.221-232, Springer-Verlag, 1997.
5. T. Pedersen, "A threshold cryptosystem without a trusted party," In Advances in Cryptology-Eurocrypt '91, vol.547 LNCS, pp.522-526, Springer-Verlag, 1991.
6. T. P. Pedersen. Distributed Provers and Verifiable Secret Sharing Based on the Discrete Logarithm Problem, PhD thesis, Aarhus University, Computer Science Department, Aarhus, Denmark, March 1992.
7. R. Cramer, R. Gennaro and B. Schoenmakers. "A secure and optimally efficient multi-authority election scheme," In Advances Cryptology – Eurocrypto'97, LNCS vol.1233, pp.103-118, Springer-Verlag, 1997.
8. D. Chaum, and T. P. Pedersen, "Wallet databases with observers" In Advances in Cryptology-Crypto'92, vol.740 LNCS, pp.89-105, Springer-Verlag, 1993.
9. J. Kilian, and F. T. Leighton, "Fair Cryptosystem Revisited," In Advances Cryptology – Crypto'95, LNCS 963, pp.208-221, Springer-Verlag, 1995.

10. A. Young, and M. Yung, "Auto-Recoverable Auto-Certifiable Cryptosysem," In Advances Cryptology – Eurcrypt'98. LNCS, Springer-Verlag. 1998.
11. W. Mao, "Publicly Verifiable Partial Key Escrow," Proc. ACISP, pp.240-248, Springer-Verlag, 1997.
12. M. Stadler. "Publicly Verifiable Secret Sharing," In Advances Cryptology – Eurocrypt'96 LNCS vol.1070, pp.190-199, Springer-Verlag, 1996.
13. A. Fiat, and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," In Advances Cryptology-Crypto '86, vol.263 LNCS, pp.186-194, Springer-Verlag, 1987.