Key Escrow in Mutually Mistrusting Domains*

L. Chen, D. Gollmann and C.J. Mitchell

Information Security Group
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
E-mail: {liqun,dieter,cjm}@dcs.rhbnc.ac.uk

Abstract. In this paper we present a key escrow system which meets possible requirements for international key escrow, where different domains may not trust each other. In this system multiple third parties, who are trusted collectively but not individually, perform the dual role of providing users with key management services and providing authorised agencies in the relevant domains with warranted access to the users' communications. We propose two escrowed key agreement mechanisms, both 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 cryptographic key for end-to-end encryption. 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'. The first mechanism makes use of a single set of key escrow agencies moderately trusted by mutually mistrusting domains. The second mechanism uses a transferable and verifiable secret sharing scheme to transfer key shares between two groups of key escrow agencies, where one group is in each domain.

1 Introduction

1.1 Key escrow in mutually mistrusting domains

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 a key escrow agency (or a set of agencies) and later delivered to government agencies when lawfully authorised. Following the US government's Clipper proposals, [1], a number of key escrow systems have recently been proposed, and for an overview of the field, the reader is referred to [4].

When users communicate internationally, there is a potential requirement to provide the law enforcement agencies of all the relevant countries, e.g. the

* This work has been jointly funded by the UK EPSRC under research grant GR/J17173 and the European Commission under ACTS project AC095 (ASPeCT).

originating and destination countries for the communication, with warranted access to the user traffic. For example, a global mobile telecommunications system might provide an end-to-end confidentiality service to two mobile users in two different countries, and law enforcement agencies in both these countries might independently wish to intercept these communications. To make matters more complicated, these two countries will typically not trust one other (such domains are referred to as mutually distrusting countries in [6]); for example, a law enforcement agency in one country might not wish to let their counterpart in any other country know that a particular user's communications are being intercepted.

We are concerned here with international key escrow, and we assume throughout that the countries involved do not trust one another; for the maximum generality we refer to *domains* instead of countries throughout. We also refer to *interception authorities* where we mean bodies such as law enforcement agencies who may be given the right to access communications within a single domain. Finally we refer to *escrow agencies* or *Trusted Third Parties (TTPs)* who will be responsible for maintaining all the information necessary to provide access to interception agencies, when presented with the appropriate legal authorisation.

We now state our requirements for key escrow in an international (i.e. a multi-domain) context.

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.

1.2 Prior approaches

Jefferies, Mitchell and Walker [8] recently proposed a novel key escrow mechanism suitable for international use, called the 'JMW' mechanism for short. In that scheme every user has an associated TTP. If two users, communicating with each other securely by using end-to-end encryption, are located in different domains, then the relevant pair of TTPs (one in each domain) collaboratively perform the dual role of providing the users with key management services and providing the two interception agencies with warranted access to the users' communications. A session key for end-to-end encryption is established based on Diffie-Hellman key exchange [5]. An asymmetric key agreement pair for one user (the receiver) is separately computed by both TTPs (one in each domain) using a combination of a secret key shared between them and the receiver's name, and another asymmetric key agreement pair for the other user (the sender) is generated by himself. The receiver computes the session key by combining his private key (transferred securely from his own TTP) with the sender's public

key (sent with the encrypted message). The sender computes the same session key by combining his private key with the receiver's public key (obtained from the sender's own TTP). Interception agencies in each domain can retrieve the session key from the TTP in the same domain.

Note that this mechanism meets the three requirements for key escrow listed above. However, it requires the following assumptions about trust relationships among the users, TTPs and interception agencies.

1. Each user believes that their own TTP (as well as the TTPs of any other users with which they communicate) will issue proper key agreement values and certificates, and will not reveal the escrowed key illegally.
2. Each TTP believes that the user, as a sender, will provide the correct public key (matching the secret key he uses for securing messages he sends).
3. Each TTP believes that the other TTP will contribute proper key agreement values and certificates, and will not reveal the escrowed key illegally.
4. Each interception agency believes that the TTP in its domain will provide the correct escrowed key when requested.

In [6], Frankel and Yung give a different scheme for international key escrow, which requires a key escrow agency (or agencies) to be trusted by more than one domain.

1.3 Our contribution

In this paper we suppose that, in some environments where international key escrow is required, TTPs may not be trusted individually to provide proper contributions to an escrowed key and to reveal the key legally, and users also may not be trusted to provide proper contributions to an escrowed key.

We consider two related key escrow mechanisms with the following properties.

1. *The schemes use a set of moderately trusted third parties instead of a single TTP, in an effort to prevent a single TTP from corrupting an escrowed key.* For the purposes of this paper, moderately trusted third parties are trusted collectively, but not individually, by users, interception agencies and another set of TTPs.
   Key splitting schemes have previously been used for splitting an escrowed key into n shares escrowed by n agencies in proposed key escrow systems (e.g. see [4, 9, 11, 12]); we also make use of a k out of n threshold scheme. Such a scheme allows any subset of k of the n escrow agencies to affect the recovery of a complete key, but prohibits any group of fewer than k agencies from recovering a complete key.
2. *They use a verifiable secret sharing scheme in order to prevent deviant users from subverting the secret sharing scheme by providing improper shares.* Such a scheme has previously been adopted in a key escrow system to let a group of key escrow agencies verify that they have valid shares [9].
3. *They use an affine expansible verifiable secret sharing scheme to let users and third parties jointly generate an escrowed key, thus preventing deviant users from obtaining a 'shadow-key' (not available to the escrow agency).*

4. *The second scheme makes use of a transferable verifiable secret sharing scheme to transfer shares between two sets of key escrow agencies which may not trust each other.*

The remainder of the paper is subdivided as follows. In section 2, we present a transferable verifiable secret sharing scheme and an affine expansible verifiable secret sharing scheme based on the Shamir secret sharing scheme, [15], and the Pedersen verifiable secret sharing scheme, [13]. We then propose two mechanisms for international key escrow in section 3. The first, which incorporates Frankel and Yung's idea, [6], makes use of a single group of key escrow agencies moderately trusted by mutually mistrusting domains. The second scheme, which is an alternative to the JMW mechanism, adopts the transferable and verifiable secret sharing scheme to transfer shares between two sets of moderately trusted key escrow agencies, one set within each of two mutually mistrusting domains. In both mechanisms, users and key escrow agencies jointly generate an escrowed key by using the affine expansible verifiable secret sharing scheme.

In section 4, we consider possible trust relationships among the three types of entity involved in an international key escrow system, namely moderately trusted third parties, potentially untrustworthy users and multiple mistrusting domains. We conclude by giving two open questions.

2 Verifiable Secret Sharing

In this section we first briefly describe the Shamir secret sharing scheme [15] and the Pedersen verifiable secret sharing scheme [13]. We then discuss how to transfer a shared secret between two domains, and also how to share an affine function of a shared secret, using modifications of the Shamir and Pedersen schemes. This work will provide the basis for the key escrow schemes described subsequently.

2.1 The Shamir scheme

A (k, n)-threshold secret sharing scheme is a protocol in which a *dealer* distributes partial information (a *share*) about a secret to each of n participants such that

- ◇ no group of fewer than k participants can obtain any information about the secret, and
- ◇ any group of at least k participants can compute the secret.

We now describe the Shamir (k, n)-threshold secret sharing scheme, [15]. Suppose p and q are large primes such that q divides $p-1$, and g is an element of order q in $\mathcal{Z}_p^*$. It is assumed that p, q and g are publicly known. These parameters will be used throughout this paper. Unless otherwise stated all arithmetic will be computed modulo p.

Let the secret s be an element of $\mathcal{Z}_\Pi$. In order to distribute s among P_1, ..., P_n (where $n < q$) the dealer chooses a polynomial of degree $k-1$:

$$f(x) = a_0 + a_1 x + ... + a_{k-1} x^{k-1},$$

where $f \in \mathcal{Z}_\Pi[\S]$ and $a_0 = s$. Each participant P_i $(1 \leq i \leq n)$ receives $s_i = f(x_i)$ as his private share, where $x_i \in \mathcal{Z}_\Pi - \{\prime\}$ is public information about P_i ($x_i \neq x_j$, for $i \neq j$).

Any k participants (without loss of generality we assume that they are P_1, P_2, ..., P_k) can find $f(x)$ by the interpolation formula,

$$f(x) = \sum_{i=1}^{k} (\prod_{h \neq i} \frac{x - x_h}{x_i - x_h}) f(x_i) = \sum_{i=1}^{k} (\prod_{h \neq i} \frac{x - x_h}{x_i - x_h}) s_i.$$

Thus

$$s = f(0) = \sum_{i=1}^{k} (\prod_{h \neq i} \frac{x_h}{x_h - x_i}) s_i.$$

2.2 The Pedersen scheme

Assume that a dealer has a secret $s \in \mathcal{Z}_\Pi$ and corresponding public value $h = g^s$. This secret can be distributed to and verified by P_1, ..., P_n, in the following way:

1. The dealer computes shares s_i using the Shamir secret sharing scheme by first choosing a polynomial $f(x) = a_0 + a_1 x + ... + a_{k-1} x^{k-1}$ over $\mathcal{Z}_\Pi$ satisfying $a_0 = s$ and then computing $s_i = f(x_i)$ $(1 \leq i \leq n)$. Here x_i is public information about P_i as previously.
2. The dealer sends the share s_i secretly to P_i $(1 \leq i \leq n)$ and broadcasts a verification sequence

   $$V = (g^{a_0}, g^{a_1}, ..., g^{a_{k-1}})$$

   to all n participants.
3. Each P_i $(1 \leq i \leq n)$ computes

   $$h_i = \prod_{j=0}^{k-1} (g^{a_j})^{(x_i)^j},$$

   and verifies whether

   $$h_i = g^{s_i}.$$

   If this does not hold then P_i broadcasts s_i and stops. Otherwise P_i accepts the share.
4. Any k participants, who have accepted their shares, can find s as described in the Shamir secret sharing scheme above.

2.3 Transferable verifiable secret sharing

We now consider how to transfer a shared secret between two groups of participants. We start by stating our requirements for a (k, m, n)-transferable verifiable secret sharing scheme, where k, m, and n are positive integers satisfying $1 < k \leq min\{m, n\}$.

- A secret s shared by m participants P_1, ..., P_m needs to be transferred to, and then shared by, another n participants Q_1, ..., Q_n.
- The participants Q_j $(1 \leq j \leq n)$ must be able to verify their own private shares without communicating with other participants in the same domain.
- Any group of at least k participants in Q_1, ..., Q_n, who have accepted their shares, can compute s.
- No group of fewer than k participants in Q_1, ..., Q_n can obtain any information about s.

We now present a transferable verifiable secret sharing scheme based on the Shamir and Pedersen schemes.

Algorithm 1 *Assume that m participants P_i $(1 \leq i \leq m)$ share a secret $s \in Z_{11}$ using the Pedersen scheme. This secret can be transferred to and verified by another n participants Q_j $(1 \leq j \leq n)$, in the following way:*

1. *Each P_i $(1 \leq i \leq m)$ computes new shares s_{ij} $(1 \leq j \leq n)$ using the Shamir secret sharing scheme by:*
   - *first choosing a polynomial $f_i(x) = a_{i0} + a_{i1}x + ... + a_{i(k-1)}x^{k-1}$ over Z_{11} satisfying $a_{i0} = s_i$, and*
   - *then computing $s_{ij} = f_i(x_j)$. Here x_j is public information about Q_j.*
2. *P_i $(1 \leq i \leq m)$ sends s_{ij} secretly to Q_j $(1 \leq j \leq n)$ and broadcasts a verification sequence*

$$V_i = (g^{a_{i0}}, ..., g^{a_{i(k-1)}})$$

   to all n participants Q_1, ..., Q_n.
3. *On receipt of s_{ij} and V_i $(1 \leq i \leq m)$, Q_j $(1 \leq j \leq n)$ computes*

$$h_{ij} = \prod_{l=0}^{k-1} (g^{a_{il}})^{(x_j)^l},$$

   and verifies whether

$$h_{ij} = g^{s_{ij}}.$$

   If this does not hold, Q_j broadcasts s_{ij} and stops. Otherwise Q_j accepts the share.

Theorem 2 *The above algorithm has the following properties.*

1. *Any group of at least k participants in Q_1, ..., Q_n, who have accepted their shares following* **Algorithm 1**, *can find s_i $(1 \leq i \leq m)$, and hence compute s.*
2. *No group of fewer than k participants in Q_1, ..., Q_n can obtain any information about s_i $(1 \leq i \leq m)$ and s.*
3. *Each Q_j $(1 \leq j \leq n)$ can verify s_{ij} $(1 \leq i \leq m)$ and g^s without communicating with other participants in the same domain.*

Proof

All three parts of the theorem hold by using precisely the same arguments as used to prove the same statements for the Pedersen scheme. □

This scheme will be used to transfer a partial escrowed key from a set of TTPs in one domain to another set of TTPs in a second domain in **Mechanism 7** described in the next section. The two groups of participants do not have to trust each other. If fewer than k participants in any domain follow the scheme, the secret transfer cannot be successful, but no one can subvert the algorithm by forcing anyone else to accept a fraudulent secret.

2.4 Affine expansible verifiable secret sharing

We now consider an *affine expansion* of threshold secret sharing. We start by stating our requirements for 'affine expansion'.

- A secret $s \in Z_q$ is shared by m participants P_1, ..., P_m. Its affine function $w = as + b$, where $a, b \in Z_q$ and $a \neq 0$, needs to be shared by the same participants. Here a and b are public information about P_i $(1 \leq i \leq m)$.
- No group of fewer than k participants can obtain any information about w.
- Any group of at least k participants can compute w.

We now present an affine expansible verifiable secret sharing scheme based on the Shamir and Pedersen schemes.

Algorithm 3 *Assume that m participants P_i $(1 \leq i \leq m)$ share a secret $s \in \mathcal{Z}_\Pi$ using the Pedersen scheme, and know public information $a \in Z_q - \{0\}$ and $b \in Z_q$. A new secret $w = as + b \in Z_q$ can be shared and verified by the same m participants without communicating with one another. The new shares w_i are*

$$w_i = as_i + b.$$

The corresponding public keys are

$$g^{w_i} = g^{as_i+b} = (g^{s_i})^a g^b, and$$

$$g^w = g^{as+b} = (g^s)^a g^b.$$

Theorem 4 *The above algorithm has the following properties.*

1. *It meets the requirements for affine expansible secret sharing.*
2. *P_i ($1 \leq i \leq m$) can verify w_i ($1 \leq i \leq m$) and g^w without communicating with other participants.*

Proof

This theorem again follows using precisely the same arguments as are used to establish the properties of the Pedersen scheme. □

This scheme will be used to let third parties provide an contribution to an escrowed key in **Mechanism 5** and **Mechanism 7** described below. Because the contribution is not known to users, it is difficult for the users to subvert the escrowed key by using a hidden 'shadow-public-key', the corresponding 'shadow-private-key' of which cannot be computed by using a real key pair and 'shadow-public-key' [9].

3 Escrowed key agreement

3.1 Assumptions

We make the following assumptions for our model of an international key escrow system.

- ◇ Two entities A and B, located in mutually mistrusting domains, want to communicate securely with each other. For this purpose they need to verify one another's identity and establish a shared session key K_{AB}, although before the authentication and key distribution processing starts they do not share any secret.
- ◇ The communications between A and B have to meet potential legal requirements for warranted interception. Interception agencies in each domain are not actively involved in the authentication and key distribution procedures, but may require access to the session key K_{AB}.
- ◇ In the first scheme (**Mechanism 5**) a single set of TTPs $\{T_1, ...,T_m\}$ are used as both multiple authentication servers for the users, and key escrow agencies for the interception agencies in both domains. In the second scheme (**Mechanism 7**) two sets of TTPs $\{T_1, ...,T_m\}$ and $\{U_1, ..., U_n\}$, one group in each domain, are used as multiple authentication servers for the users and key escrow agencies for the interception agencies. In both cases they are responsible for verifying A's and B's identities, establishing a session key K_{AB}, and escrowing the session key. They are trusted by both the users and interception agencies collectively, but not individually.

3.2 Mechanism 1

This escrowed key agreement scheme is based on Diffie-Hellman key exchange [5] and the verifiable secret sharing schemes described in section 2. In the mechanism, A and B are users in separate domains, and m moderately TTPs T_1, ...,

T_m work for both users as authentication servers, and for interception agencies in both domains as key escrow agencies. We assume that A and B have authenticated channels with T_i $(1 \leq i \leq m)$. As in the JMW mechanism, these m TTPs agree a commonly held secret key $K(T_1, ..., T_m)$ and a function f. This function f shall take as input the shared secret key and the names of A and B, and generate a private integer S_{TAB}. The scheme is designed so that for some positive integer k $(k \leq m)$, any set of k TTPs can compute the session key established between A and B, but no group of $k-1$ or less TTPs can derive any useful information about this session key.

Mechanism 5 *A set of TTPs T_1, ..., T_m assist two users A and B in establishing a session key K_{AB}, and escrow the key collectively.*

1. *A secretly chooses and stores its private key agreement value S_A, and computes the corresponding public value P_A $(= g^{S_A})$, the private shares S_{A_i} $(1 \leq i \leq m)$ of S_A as defined in subsection 2.1, and the public verification sequence V_A as defined in subsection 2.2, and then sends S_{A_i} and V_A to T_i $(1 \leq i \leq m)$.*
2. *B follows the same procedure as A (choosing S_B, creating private shares S_{B_i}, a verification sequence V_B, and sending S_{B_i} and V_B to T_i $(1 \leq i \leq m)$).*
3. *T_i $(1 \leq i \leq m)$ verifies S_{A_i}, P_A, and S_{B_i}, P_B as described in subsection 2.2. If the verification fails, T_i broadcasts the suspect share value and stops; otherwise T_i accepts the share.*
4. *T_i $(1 \leq i \leq m)$ does the following:*
   - *obtains S_{TAB} by using the function f with $K(T_1, ..., T_m)$, A and B,*
   - *calculates P_{AT} $(= P_A^{S_{TAB}})$ and P_{BT} $(= P_B^{S_{TAB}})$, and*
   - *sends P_{AT} to B and P_{BT} to A.*
5. *A and B separately compute a session key as:*

$$K_{AB} = (P_{AT})^{S_B} = (P_{BT})^{S_A} = g^{S_A S_B S_{TAB}}.$$

Theorem 6 *The above mechanism has the property that any group of at least k TTPs can compute K_{AB} (which is what is required for escrow purposes).*

Proof

Any group of at least k TTPs can compute S_A and S_B (by the properties of the Shamir scheme discussed in subsection 2.1 above). Hence they can compute

$$K_{AB} = g^{S_A S_B S_{TAB}}$$

and the result follows. □

The mechanism has been designed to make it difficult for A and B to prevent K_{AB} from being escrowed 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, and no group of fewer than k third parties can obtain any information about K_{AB}.

The method used to compose a set of key escrow agencies, who are moderately trusted by mutually mistrusting domains, depends on the requirements for international secure telecommunications. The set could consist of TTPs licensed by domains other than the two domains being served, or by a 'super-domain' including the two domains, or one or other of the two domains.

It would be desirable if S_{TAB} could be changed from time to time (which will mean that K_{AB} also changes). This could be achieved by including a date-stamp in the function f used to compute S_{TAB}.

Compared with a number of other proposed key agreement schemes, such as, letting the two users choose the key (see [7]), letting a set of TTPs generate the key (see [3]), and letting one user and two TTPs generate the key (see [8]), this mechanism forces all involved entities, i.e. both users and the set of TTPs, to jointly generate the key, so that it may be more difficult for users and TTPs to subvert the key.

3.3 Mechanism 2

This escrowed key agreement scheme is based on Diffie-Hellman key exchange [5] and the transferable verifiable secret sharing scheme described in section 2. In this mechanism, A and B are users in different domains. There are m TTPs T_1, ..., T_m working for A as authentication servers (in A's domain), and n TTPs U_1, ..., U_n working for B as authentication servers (in B's domain). These servers also operate as key escrow agencies for the interception agencies in their respective domains. Each set of third parties is moderately trusted by their users and interception agencies. Users and interception agencies do not communicate with TTPs outside their domain. TTP T_i $(1 \leq i \leq m)$ can communicate with U_j $(1 \leq j \leq n)$. Again, we assume that A has an authenticated channel with each T_i, and B has an authenticated channel with each U_j. Each group of TTPs agree a secret key $K(T_1, ..., T_m)$ or $K(U_1, ..., U_n)$ and a function f. This function f shall take as input the shared secret keys and the names of A and B, and generate private integers S_{TAB} and S_{UAB} respectively. The scheme is designed so that for some positive integer k $(k \leq min\{m, n\})$, any set of k TTPs from one or other of the two domains can compute the session key established between A and B, but no group of $k-1$ or less TTPs can derive any useful information about this session key.

Mechanism 7 *Two sets of TTPs* $\{T_1, ..., T_m\}$ *and* $\{U_1, ..., U_n\}$ *assist two users A and B (respectively) to establish a session key* K_{AB}*. Each set of third parties escrow the key collectively.*

1. *A secretly chooses and stores its private key agreement value* S_A*, and computes the following values:*
   - *the corresponding public value* P_A $(= g^{S_A})$*,*
   - *the private shares* S_{A_i} $(1 \leq i \leq m)$ *as defined in subsection 2.1, and*
   - *the public verification sequence* V_A *as defined in subsection 2.2, and then sends* S_{A_i} *and* V_A *to* T_i $(1 \leq i \leq m)$*.*

2. *T_i $(1 \leq i \leq m)$ verifies S_{A_i} and P_A as described in subsection 2.2. If the verification fails then T_i broadcasts the suspect share value and stops; otherwise T_i accepts the share.*
3. *B secretly chooses and stores its private key agreement value S_B, and computes the following values:*
   - *the corresponding public value P_B $(= g^{S_B})$,*
   - *the private shares S_{B_j} $(1 \leq j \leq n)$ as defined in subsection 2.1, and*
   - *the public verification sequence V_B as defined in subsection 2.2, and then sends S_{B_j} and V_B to U_j $(1 \leq j \leq n)$.*
4. *U_j $(1 \leq j \leq n)$ verifies S_{B_j} and P_B as described in subsection 2.2. If the verification fails then U_j broadcasts the suspect share value and stops; otherwise U_j accepts the share.*
5. *T_i $(1 \leq i \leq m)$ does the following:*
   - *obtains S_{TAB} by using the function f with $K(T_1, ..., T_m)$, A and B,*
   - *calculates P_{AT} $(= P_A^{S_{TAB}})$,*
   - *calculates $S_{A_{ij}}$ $(1 \leq j \leq n)$ from S_{A_i} as defined in subsection 2.3,*
   - *computes the 'private shares' $S_{A_{ij}}S_{TAB}$, and their corresponding public values $g^{S_{A_{ij}}S_{TAB}}$ as defined in subsection 2.4, and the public verification sequence V_{A_i} as defined in subsection 2.2.*
   - *Finally, T_i sends $S_{A_{ij}}S_{TAB}$, V_{A_i} and P_{AT} to U_j $(1 \leq j \leq n)$.*
6. *U_j $(1 \leq j \leq n)$ verifies $S_{A_{ij}}S_{TAB}$, V_{A_i} and P_{AT} as described in subsection 2.3. If the verification fails then U_j broadcasts the suspect share value and stops, otherwise U_j accepts the share.*
7. *U_j $(1 \leq j \leq n)$ does the following:*
   - *obtains S_{UAB} by using the function f with $K(U_1, ..., U_n)$, A and B,*
   - *calculates P_{ATU} $(= P_{AT}^{S_{UAB}})$ and sends it to B,*
   - *calculates P_{BU} $(= P_B^{S_{UAB}})$,*
   - *calculates $S_{B_{ji}}$ $(1 \leq i \leq m)$ from S_{B_j} as defined in subsection 2.3,*
   - *computes the 'private shares' $S_{B_{ji}}S_{UAB}$, and their corresponding public values $g^{S_{B_{ji}}S_{UAB}}$ as defined in subsection 2.4, and the public verification sequence V_{B_j} as defined in subsection 2.2, and, finally,*
   - *sends $S_{B_{ji}}S_{UAB}$, V_{B_j} and P_{BU} to T_i $(1 \leq i \leq m)$.*
8. *T_i $(1 \leq i \leq m)$ verifies $S_{B_{ji}}S_{UAB}$, V_{B_j} and P_{BU} as described in subsection 2.3. If the verification fails then T_i broadcasts the suspect share value and stops, otherwise T_i accepts the share, calculates P_{BTU} $(= P_{BU}^{S_{TAB}})$ and sends it to A.*
9. *A and B can now separately compute the session key:*

$$K_{AB} = (P_{BTU})^{S_A} = (P_{ATU})^{S_B} = g^{S_A S_B S_{TAB} S_{UAB}}.$$

Theorem 8 *The above mechanism has the property that any group of at least k TTPs (in either domain) can compute K_{AB}.*

Proof

The proof follows immediately from the results in subsection 3.2 above. □

In this mechanism, the two sets of third parties in both domains do not have to trust each other, as mentioned in subsection 2.3. For the same reasons as in the previous mechanism, it is suggested that S_{TAB} and S_{UAB} should be changed as often as required.

4 Further considerations

In a key escrow system, the differing requirements of users and interception authorities are further complicated by the introduction of the key escrow agencies (or TTPs). The key escrow agencies are responsible to the interception agencies for preventing criminal users from abusing escrowed keys. Both the users and interception agencies should be in a position to check that the key escrow agencies cannot reveal escrowed keys illegally. In international key escrow, the relationships amongst these three groups of entities becomes still more complicated because more than one domain is involved. The key escrow agencies in one domain have a potential requirement to check that the key escrow agencies in the other domain cannot subvert the escrowed keys.

In this section, we discussion some aspects of the trust relationships between the various entities involved.

4.1 Moderately trusted third parties

There are two major reasons why we make use of moderately trusted third parties in this paper.

- ◇ If interception agencies are not actively involved in session key establishment for possibly deviant users and do not store every session key themselves, key escrow agencies are required to provide a valid key when lawfully authorised. Although the key escrow agencies may not be trusted individually, a group of them might be collectively trusted by the interception agencies.
- ◇ If two users sharing no secret want to communicate security with each other, they need an authentication service provided by authentication servers. Although the servers may not be trusted individually, a group of them might be collectively trusted by their users.

Four kinds of key splitting schemes based on secret sharing schemes (e.g. [2, 15]) have been used for splitting an escrowed key into n shares escrowed by n agencies in previously proposed key escrow systems. The first approach involves 'splitting' with an n out of n scheme, where all n components are needed to restore a given key [12]. The second approach uses splitting with an k out of n threshold scheme, which allows any subset of k of the n escrow agencies to affect the recovery of a complete key, but prohibits any group of fewer than k agencies from recovering a complete key [9]. The third approach involves splitting with an (n, t, u)-escrow scheme, which allows a subset of the escrow agencies to recover a key, where t escrow agencies could conspire without compromising a key, and $n - u$ agencies could 'withhold' their components without interfering with key

recovery ($t < u \leq n$) [11]. The last approach involves splitting with a 'general monotone access structure', which allows for the specification of arbitrary subsets of escrow agencies that can work together to restore a key [4].

We have used the second approach in the mechanisms described here. Actually, any one of these four splitting schemes could have been chosen, and in practice the choice would depend on the requirements for establishing a set of moderately trusted third parties. Note also that the idea of Reiter et al. regarding secure group implementation in [14] can be used to establish such a set of moderately trusted third parties and their group key.

4.2 Untrustworthy users

We now consider the case where users are not trustworthy in a key escrow system. Kilian and Leighton gave a 'shadow-public-key' attack in [9], and we now see how this attack might work in a key escrow system based on Diffie-Hellman key exchange. Each normal user generates a pair (P, S), publishes P and gives an interception authority the ability to reconstruct S, so that both the users and interception authority can compute an escrowed key by using Diffie-Hellman key agreement. In this attack, each of two attackers instead generates two key pairs (P, S) and (P', S'), where (P, S) is a proper (public- key, private-key) pair, (P', S') is a 'shadow' key pair, and $P' = f(P)$ where f is an easily computed and publicly known function. Each of them uses (P, S) in the same way as would an ordinary user, but keeps S' reserved as his shadow private key. Both attackers separately compute a 'shadow-escrowed-key' by using his 'shadow-private-key' and the other's 'shadow-public-key'. If it is infeasible to obtain S' by knowing P, P' and S, the interception authority cannot obtain the 'shadow-escrowed-key'. Furthermore the interception authority may not detect this cheating.

We presented an affine expansible verifiable secret sharing scheme in subsection 2.4, which provided the basis of users and third parties jointly generating an escrowed key in order to prevent the users from using a hidden 'shadow-key'. Note that it only makes sense to prevent criminal users from obtaining a S' which is in feasibly computed by using P, P' and S.

The further problem is how in practice to prevent criminal users from abusing the key escrow system by using improper keys, for examples, using an old escrowed key instead of a current one, using a modification of the escrowed key, e.g. which may be a publicly known function of the real escrowed key, and using a 'shadow-public-key', where S' may feasibly be computed by knowing P, P' and S. Although these abuses are all detectable, a key escrow mechanism may never check for such abuses, giving deviant users greater leeway in their abuses.

A number of approaches could be used to prevent the above abuses, such as, keeping all old escrowed keys in a valid period to check if they are used again, and monitoring all communication channels between suspected criminal users [10]. Unfortunately, these approaches may not be practical, particularly, in complicated mobile telecommunications systems.

In fact, it is impossible for a key escrow system to force two users to use only the current escrow key if the users share a secret or can use their own security

system. For the purposes of this paper, we suppose that two users, who want to communicate securely with each other, have to get assistance from key escrow agencies in order to authenticate one another's identity and establish a shared session key. We assume it is detectable if the users subvert key escrow systems by using an old escrowed key or a modified escrowed key. However we have not answered the question of how to force users to use only the current escrowed key.

4.3 Multiple mistrusting domains

So far we have discussed key escrow in mutually mistrusting domains. However, some modern secure communications may cover more than two domains. For example, in a global mobile telecommunications system, two users, respectively, are citizens of countries C and D, work for countries E and F, are registered with two mobile companies belong to countries G and H, and are roaming in two countries I and J. Their traffic might conceivably need to be intercepted by agencies in any of countries $C - J$, and hence it may be necessary to try and devise an international key escrow system which provides all governments involved with warranted access to user communications. To make matters more complicated, the countries involved may not all trust each other.

Our first mechanism, as described in subsection 3.2, could be used for this purpose. However, whether or not a set of key escrow agencies could be set up which are moderately trusted by multiple mistrusting domains, depends on political considerations beyond the scope of this paper. The second mechanism, as described in subsection 3.3, could also be used for this purpose, at least in theory. The problem is that each set of key escrow agencies in each domain involved have to collaborate to provide contributions to the escrowed key. This may not be practical, particularly when the number of domains involved is quite large.

5 Conclusions

We have described a key escrow system using moderately trusted third parties in mutually mistrusting domains and analysed its use.

The following open questions are of potential practical importance.

- ◇ Do there exist practical key escrow systems forcing users to use only the current escrowed session key?
- ◇ Can a practical key escrow scheme be designed for the case where more than two domains are involved, and where escrow agencies are not permitted to span more than one domain?

References

1. National Institute of Standards and Technology. FIPS Publication 185: Escrowed Encryption Standard. February 1994.

2. G.R. Blakley. Safeguarding cryptographic keys. In *the Proceedings of AFIPS 1979 NCC, Vol. 48, Arlington, Va.*, pages 313–317, June 1979.
3. L. Chen, D. Gollmann, and C. Mitchell. Key distribution without individual trusted authentication servers. In *Proceedings: the 8th IEEE Computer Security Foundations Workshop*, pages 30–36. IEEE Computer Society Press, Los Alamitos, California, June 1995.
4. D.E. Denning and D.K. Branstad. A taxonomy for key escrow encryption systems. *Communications of the ACM*, 39(3):34–40, 1996.
5. W. Diffie and M.E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, November 1976.
6. Y. Frankel and M. Yung. Escrow encryption systems visited: attacks, analysis and designs. In D. Coppersmith, editor, *Lecture Notes in Computer Science 963, Advances in Cryptology — CRYPTO '95*, pages 222–235. Springer – Verlag, 1995.
7. L. Gong. Increasing availability and security of an authentication service. *IEEE Journal on Selected Areas in Communications*, 11:657–662, 1993.
8. N. Jefferies, C. Mitchell, and M. Walker. A proposed architecture for trusted third party services. In E. Dawson and J. Golić, editors, *Lecture Notes in Computer Science 1029, Cryptography: Policy and algorithms Conference*, pages 98–104. Springer-Verlag, 1996.
9. J. Kilian and T. Leighton. Fair cryptosystems, revisited. In D. Coppersmith, editor, *Lecture Notes in Computer Science 963, Advances in Cryptology - CRYPTO '95*, pages 208–221. Springer – Verlag, 1995.
10. A.K. Lenstra, P. Winkler, and Y. Yacobi. A key escrow system with warrant bounds. In D. Coppersmith, editor, *Lecture Notes in Computer Science 963, Advances in Cryptology - CRYPTO '95*, pages 197–207. Springer – Verlag, 1995.
11. S. Micali and R. Sidney. A simple method for generating and sharing pseudo-random functions, with applications to clipper-like key escrow systems. In D. Coppersmith, editor, *Lecture Notes in Computer Science 963, Advances in Cryptology - CRYPTO '95*, pages 185–196. Springer – Verlag, 1995.
12. J. Nechvatal. A public-key based key escrow system. Journal of Systems and Software, to appear October 1996.
13. T.P. Pedersen. Distributed provers with applications to undeniable signatures. In D. W. Davies, editor, *Lecture Notes in Computer Science 547, Advances in Cryptology: Proc. Eurocrypt '91*, pages 221–238. Berlin: Springer-Verlag, 1991.
14. M.K. Reiter, K.P. Birman, and R. van Renesse. A security architecture for fault-tolerant systems. *ACM Transactions on Computer Systems*, 12:340–371, 1994.
15. A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.

This article was processed using the LaTeX macro package with LLNCS style