Increasing Efficiency of International Key Escrow in Mutually Mistrusting Domains

Keith M. Martin*

Katholieke Universiteit Leuven, Dept. Elektrotechniek - ESAT
Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium
keith.martin@esat.kuleuven.ac.be

Abstract. This paper is concerned with key escrow protocols for use in international communications environments, where communication domains do not necessarily trust one another. It is concerned particularly with systems where users place their trust collectively with groups of trusted third parties. We consider two different protocols, discuss and improve their efficiency and generalise the type of key splitting.

1 Introduction

There is a great deal of current interest in *key escrow* protocols, loosely defined as systems that protect data using conventional cryptographic methods but, under special circumstances, make it possible for the cryptographic protection to be circumvented allowing access to either the data itself, or some cryptographic key that protects it. Such a circumstance is when law enforcement agencies have a warrant to obtain access to certain specified communications. For an introduction to many of the existing key escrow systems see [5].

Most proposed key escrow schemes rely on the use of *trusted third parties* (TTPs). These TTPs are trusted organisations who provide network services which include acting as a trusted itermediary between network users and law enforcement agencies. In most key escrow schemes a user lodges some secret information with a TTP. This information is kept confidential during normal network use, however under special circumstances the information can either be released, or used, in order to permit access to the user's communications.

Providing a key escrow service within one *domain* (perhaps a nation state) is likely to require a network of TTPs. Each user identifies with one (or more) of these TTPs (their *home* TTPs). Within a single domain it may be possible that different TTPs have established working relationships that allow secure information to be exchanged under suitable circumstances. Key escrow in a network spanning several domains (international telecommunications) is considerably more challenging. The main problem is that it is likely that law enforcement agencies belonging to one domain will only legally be able to approach TTPs

* Supported by the European Commission under ACTS project AC095 (ASPeCT)

that also belong to that domain. This has the potential to limit the scope and efficiency of interception. Further, it may be more difficult, or indeed impossible, to establish trust relationships between TTPs belonging to different domains. In this paper we are interested in key escrow within a multiple domain environment.

For application within international telecommunications networks, one of the most studied key escrow proposals is the *JMW Protocol* [12, 13]. In the basic version of the protocol each user has one home TTP. It is assumed that this is the only TTP with which the user can communicate. The JMW Protocol can be used to establish a session key between the two users (see Sect. 2.1).

One, of several, generalisations of the JMW Protocol is to weaken the requirement that a user fully trusts their single home TTP. Instead, the trust is spread among a *group* of home TTPs in such a way that the user need only *jointly trust* certain sets of TTPs [4]. We recall two of the main protocols from [4] (Sect. 2) and generalise the key splitting techniques used in each variant (Sects. 3 and 4). We then reduce several different communication costs for the second variant (Sects. 5 and 6) and consider in detail the case of threshold splitting (Sect. 7).

2 The JMW Protocol and Splitting Variants

2.1 The JMW Protocol

We describe the basic protocol proposed in [12]. We assume that user A wants to establish a session key with user B. Let the home TTP of A be TA, and the home TTP of B be TB. Let p be a prime and g be a primitive element modulo p. Each TTP is equipped with a signature and verification algorithm pair, which we denote by $\mathrm{sig}_{\mathrm{TA}}(\cdot)$, $\mathrm{ver}_{\mathrm{TA}}(\cdot)$ and $\mathrm{sig}_{\mathrm{TB}}(\cdot)$, $\mathrm{ver}_{\mathrm{TB}}(\cdot)$. The TTPs TA and TB share a secret key K_{TAB} and a key generation function f, which takes as input a user identification number and K_{TAB}, and outputs a *private receive key* for that user. We assume that prior to communication, B has received a copy of their *private receive key* $b = f(\mathrm{B}, K_{\mathrm{TAB}})$. The protocol runs as follows:

1. User A sends a message to TA that they want to communicate with user B.
2. TA chooses a *private send key* a for A and computes $b = f(\mathrm{B}, K_{\mathrm{TAB}})$. TA then sends the values a, $\mathrm{sig}_{\mathrm{TA}}(g^a)$, and g^b, to A.
3. A computes $K_{\mathrm{AB}} = g^{ba} \bmod p$ and sends $\mathrm{sig}_{\mathrm{TA}}(g^a)$ to B.
4. B verifies A's public send key g^a and computes $K_{\mathrm{AB}} = g^{ab} \bmod p$.

Thus users A and B can now communicate using session key K_{AB}. Further, both TA and TB can provided interception authorities with either a private send or a private receive key for A or B, or a copy of session key K_{AB}.

2.2 JMW Protocol Variants Permitting Key Splitting

We now consider weakening the requirement that a user has just one home TTP in which they must place full trust. The formulation of a system by means of which trust can be spread across a group of TTPs is achieved using *key splitting*

techniques (more commonly known as *secret sharing*). Secret sharing was first suggested for use in key escrow systems in [14].

A *secret sharing scheme* is a protocol for protecting a secret among a group of entities (*participants*) in such a way that only certain sets of participants can compute the secret. Each participant is issued with a *share* of the secret. The *access structure* Γ is the collection of participants desired to be able to compute the secret. We assume that Γ is *monotone* (given a set of participants in the access structure, all larger sets that include the given set are also in the access structure). An access structure Γ is uniquely determined by its *minimal sets* (those sets that do not strictly contain another set in the access structure).

The most well-known access structures are the (k, n)-*threshold* access structures, which consist of all subsets of n participants of size at least k [2, 15]. It is possible to construct secret sharing schemes for *any* access structure.

We now describe two JMW Protocol variants that permit key splitting. We assume that each user has a collection of home TTPs. Let the home TTPs of A be $\mathrm{T}_1, \ldots, \mathrm{T}_m$ and the home TTPs of B be $\mathrm{U}_1, \ldots, \mathrm{U}_n$.

First Key Splitting Variant [13]. In this scheme, A has a distinct modulus p_A and base g_A. We assume that every pair of TTPs $(\mathrm{T}_i, \mathrm{U}_j)$ share a secret key K_{ij} and assume the existence of a function f as before. This f can be used by any pair $(\mathrm{T}_i, \mathrm{U}_j)$ to compute $b_{ij} = f(\mathrm{B}, K_{ij})$. Prior to communication, B is issued with a copy of their private receive key b as follows: each U_j sends b_{ij} $(i = 1, \ldots, m)$ to B; user B computes $b_i = \sum_{j=1}^{n} b_{ij}$; user B computes $b = \sum_{i=1}^{m} b_i$.

Protocol 1. The protocol runs as follows:

1. User A sends a message to each T_i that they want to communicate with B.
2. Each T_i chooses a_i for A, computes b_{ij} $(j = 1, \ldots, n)$ and then $g_\mathrm{A}^{b_i} = \prod_{j=1}^{n} g_\mathrm{A}^{b_{ij}}$. Each T_i then sends back to A the values a_i, $\mathrm{sig}_{\mathrm{T}_i}(g_\mathrm{A}^{a_i})$, and $g_\mathrm{A}^{b_i}$.
3. A computes $a = \sum_{i=1}^{m} a_i$, $g_\mathrm{A}^{b} = \prod_{i=1}^{m} g_\mathrm{A}^{b_i}$, and $K_\mathrm{AB} = g_\mathrm{A}^{ba} \bmod p$. A then sends $\mathrm{sig}_{\mathrm{T}_i}(g_\mathrm{A}^{a_i})$ $(i = 1, \ldots, m)$ to B.
4. B verifies components $g_\mathrm{A}^{a_i}$, and computes g_A^{a} and $K_\mathrm{AB} = g^{ab} \bmod p$.

Thus users A and B can now communicate using session key K_AB. Further, $\mathrm{T}_1, \ldots, \mathrm{T}_m$ jointly (an (m, m)-threshold access structure), or $\mathrm{U}_1, \ldots, \mathrm{U}_n$ jointly (an (n, n)-threshold access structure), can provided interception authorities with the information to construct either a private send or a private receive key for A or B, or a copy of session key K_AB (see [13] for details).

Second Key Splitting Variant [4] (Mechanism 7). This time we assume that 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 and a secret TTP key, and outputs an integer. Let $f(A, B, K_\mathrm{T}) = s_\mathrm{TAB}$ and $f(A, B, K_\mathrm{U}) = s_\mathrm{UAB}$.

In this protocol any k out of $\mathrm{T}_1, \ldots, \mathrm{T}_m$ and any k out of $\mathrm{U}_1, \ldots, \mathrm{U}_n$ are collectively trusted, where $k \leq \min m, n$. We describe a simplified version of the

protocol (omitting verification information and the explicit transfer of any public keys). From here on, all addition is modulo p, where p is a large prime. Further $p-1$ has a large prime divisor q and g is an element of order q in $\mathbf{Z}_p$.

Protocol 2. With the setting as above, users A and B proceed as follows:

1. User A chooses s_{A}, publishes $g^{s_{\mathrm{A}}}$, and generates shares $s_{\mathrm{A}_1}, \ldots, s_{\mathrm{A}_m}$ of a (k,m)-threshold scheme with secret s_{A}. For $i = 1, \ldots, m$, A sends s_{A_i} to T_i.
2. User B chooses s_{B}, publishes $g^{s_{\mathrm{B}}}$, and generates shares $s_{\mathrm{B}_1}, \ldots, s_{\mathrm{B}_n}$ of a (k,n)-threshold scheme with secret s_{B}. For $j = 1, \ldots, n$, B sends s_{B_j} to U_j.
3. T_i publishes $g^{s_{\mathrm{A}} s_{\mathrm{TAB}}}$, and generates shares $s_{\mathrm{A}_{i1}}, \ldots, s_{\mathrm{A}_{in}}$ of a (k,n)-threshold scheme with secret s_{A_i}. For $j = 1, \ldots, n$, T_i sends $s_{\mathrm{A}_{ij}} s_{\mathrm{TAB}}$ to U_j.
4. U_j publishes $g^{s_{\mathrm{B}} s_{\mathrm{UAB}}}$, and generates shares $s_{\mathrm{B}_{j1}}, \ldots, s_{\mathrm{B}_{jm}}$ of a (k,m)-threshold scheme with secret s_{B_j}. For $i = 1, \ldots, m$, U_j sends $s_{\mathrm{B}_{ji}} s_{\mathrm{UAB}}$ to T_i.
5. Each TTP T_i sends $g^{s_{\mathrm{B}} s_{\mathrm{UAB}} s_{\mathrm{TAB}}}$ to user A.
6. Each TTP U_j sends $g^{s_{\mathrm{A}} s_{\mathrm{TAB}} s_{\mathrm{UAB}}}$ to user B.

Users A and B can now both compute the session key $K_{\mathrm{AB}} = g^{s_{\mathrm{A}} s_{\mathrm{B}} s_{\mathrm{TAB}} s_{\mathrm{UAB}}}$. In either domain the key K_{AB} can be recovered by at least k home TTPs.

2.3 Efficiency of Split Escrow Variants

There are two different types of overhead encountered when implementing the protocols. Firstly, the advance agreement of information between TTPs on protocol initiation, and secondly, the performance of operations during protocol execution. In likely order of increasing communication cost, common protocol operations are generation of shares of a secret sharing scheme, and communication between users and their home TTPs, between TTPs from the same domain, and between TTPs from different domains.

Note that Protocol 1 requires every pair of TTPs from different domains to agree a common key on initiation, but during operation does not involve communication between any TTPs. In contrast, Protocol 2 makes extensive use of all but the third of the operations, but only requires each set of home TTPs (in the same domain) to agree a secret key in advance.

Another difference is that in Protocol 1 the type of key splitting used is (n,n)-threshold, whereas in Protocol 2 it is (k,n)-threshold. We proceed by generalising the type of key splitting used in both protocols, and in doing so pay particular attention to, and often improve, the resulting efficiency.

3 Generalising the First Key Splitting Variant

We now assume the existence of sets of home TTPs $\mathrm{T}_1, \ldots, \mathrm{T}_m$ and $\mathrm{U}_1, \ldots, \mathrm{U}_n$, as before, but allow the trusted subsets of TTPs to take a more general form, given by access structures Γ_{T} and Γ_{U}. Protocol 1 makes use of the fact that an (n,n)-threshold scheme can be set up without the assistance of a TTP (or *Mutually Trusted Authority*). To achieve this for more general access structures, we need to use MTA-free secret sharing [10, 11], which works as follows:

1. a subset $\{P_1, \ldots, P_k\} \in \Gamma$ of users establish a (k, k) threshold scheme among themselves (without needing a TTP), where user P_i has share s_i;
2. each user P_i generates shares of a secret sharing scheme with secret s_i and distributes these shares among a subset of the users.

The idea is to choose the various access structures, and on which subsets of users they are defined, in such a way that the resulting secret sharing scheme has the desired access structure. This is best illustrated by means of an example.

Example 1. In order to set up an MTA-free $(2, 3)$-threshold scheme among the users A,B and C, a possible construction is as follows:

1. A generates s_{A}, and B generates s_{B}. The secret is taken to be $s_{\mathrm{A}} + s_{\mathrm{B}}$;
2. A generates shares $s_{\mathrm{AB}}, s_{\mathrm{AC}}$ of a $(2, 2)$-threshold scheme with secret s_{A} and distributes s_{AB} to B and s_{AC} to C. User B sends s_{B} to C.

In [11] some protocols are given for the design of suitable MTA-free secret sharing schemes. Several efficiency measures were proposed, one being the *linkage* of a protocol, which was defined to be the total number of communications between different users necessary to enable the scheme. In Example 1 the linkage is $2+1 = 3$, which can be shown to be optimal [11].

We now apply this technique to Protocol 1. Firstly, A and $\mathrm{T}_1, \ldots, \mathrm{T}_m$, and B and $\mathrm{U}_1, \ldots, \mathrm{U}_n$, both agree to publicly designate one set $\{\mathrm{T}_1, \ldots, \mathrm{T}_s\} \in \Gamma_{\mathrm{T}}$ and $\{\mathrm{U}_1, \ldots, \mathrm{U}_t\} \in \Gamma_{\mathrm{U}}$, respectively, to be *nominated sets*. The nominated sets of TTPs have a special role to play in the protocol and for reasons of efficiency (see Sect. 2.3) the best choice of nominated set is a minimal set of smallest size. The generalised protocol set-up is similar to Protocol 1 except that the private receive key b is computed as follows: each nominated U_j sends b_{ij} $(i = 1, \ldots, s)$ to B; user B computes $b_i = \sum_{j=1}^{t} b_{ij}$; user B computes $b = \sum_{i=1}^{s} b_i$. Also prior to communication, each nominated TTP U_j computes $b'_j = \sum_{i=1}^{s} b_{ij}$. The nominated TTPs $\mathrm{U}_1, \ldots, \mathrm{U}_t$ establish an MTA-free secret sharing scheme for Γ_{U} by generating shares that correspond to the secrets $b'_1, \ldots, b'_t$, and distributing these shares appropriately among $\mathrm{U}_1, \ldots, \mathrm{U}_n$.

Protocol 3. The protocol runs as follows:

1. A informs each nominated T_i that they want to communicate with B.
2. Each nominated T_i chooses a_i for A, computes b_{ij} $(j = 1, \ldots, t)$ and $g_{\mathrm{A}}^{b_i} = \prod_{j=1}^{t} g_{\mathrm{A}}^{b_{ij}}$. Each T_i sends a_i, $\mathrm{sig}_{\mathrm{T}_i}(g_{\mathrm{A}}^{a_i})$, and $g_{\mathrm{A}}^{b_i}$, to A.
3. $\mathrm{T}_1, \ldots, \mathrm{T}_s$ establish an MTA-free secret sharing scheme for Γ_{T} by generating shares that correspond to secrets $a_1, \ldots, a_s$ and distributing these shares appropriately among $\mathrm{T}_1, \ldots, \mathrm{T}_m$.
4. Each nominated T_i computes b_i. TTPs $\mathrm{T}_1, \ldots, \mathrm{T}_s$ establish an MTA-free secret sharing scheme for Γ_{T} by generating shares that correspond to secrets $b_1, \ldots, b_s$ and distributing these shares appropriately among $\mathrm{T}_1, \ldots, \mathrm{T}_m$.
5. A computes $a = \sum_{i=1}^{s} a_i$, $g_{\mathrm{A}}^{b} = \prod_{i=1}^{s} g_{\mathrm{A}}^{b_i}$, and $K_{\mathrm{AB}} = g_{\mathrm{A}}^{ba} \bmod p$. User A then sends $\mathrm{sig}_{\mathrm{T}_i}(g_{\mathrm{A}}^{a_i})$ $(i = 1, \ldots, s)$ to B.

6. B verifies $g_{\mathrm{A}}^{a_i}$, and computes g_{A}^{a} and $K_{\mathrm{AB}} = g^{ab} \bmod p$.

Users A and B can now communicate using key K_{AB}. Further, any set of TTPs belonging to Γ_{T} can jointly compute a and b, and any set of TTPs belonging to Γ_{U} can jointly compute b. The correctness of Protocol 3 follows from the correctness of Protocol 1. As with Protocol 1 there is no need for any communication between TTPs from different domains, however our generalisation of the splitting access structure comes at the expense of some communication between TTPs from the same domain. This communication can be minimised by using MTA-free secret sharing schemes with low linkages (see [11] and Sect. 7). For many applications this TTP communication could be off-line. Note that it is only strictly necessary that nominated TTP pairs $(\mathrm{T}_i, \mathrm{U}_j)$ agree on a secret key.

4 Generalising the Second Key Splitting Variant

We now consider generalising the key splitting in Protocol 2. In fact the generalisation is straightforward, as long as secret sharing schemes are used that offer *verifiability* (it should be possible for each participant to detect the submission of a wrong share by another participant), and *linearity* (if x is a share of secret s then $ax + b$ is a share of secret $as + b$).

Verification capabilities can be provided for any secret sharing scheme which has secret and shares $s, s_1, \ldots, s_n \in \mathbf{Z}_p$ (p prime) by the dealer broadcasting $(g^s, g^{s_1}, \ldots, g^{s_n})$, where g is publicly known. Thus, providing verification is not a problem, and as before we omit it from further protocol descriptions.

Secret sharing schemes that are linear are also easy to construct. In particular schemes that are *homomorphic* [1, 9] are useful, because schemes that are homomorphic to themselves (informally, two sets of shares, and secret, can be combined to obtain a third set of shares, and secret, in the same scheme) are linear [11]. Of particular interest are schemes that are homomorphic to themselves with respect to addition over $\mathbf{Z}_p$. Important and useful examples are *geometric schemes* (constructed using projective geometry). Geometric schemes can be constructed for all access structures [16] and are equivalently described in terms of vector spaces [3] and linear error-correcting codes [8].

Having observed that generalising Protocol 2 is fairly straightforward, we note that as stated the protocol is not very efficient. Perhaps the most serious drawback is the number of separate communications that needs to take place between TTPs from different domains, namely $2mn$. In the remaining sections we discuss several methods for improving efficiency.

5 Reducing Communication between Different Domains

5.1 A Simple Communication Reduction

Communication between TTPs from different domains can be reduced (compared to Protocol 2) by the following simple modification.

Protocol 4. Replace Steps 3 and 4 in Protocol 2 with:

3. Select the smallest minimal set $\{\mathrm{T}_1, \ldots, \mathrm{T}_k\} \in \Gamma_\mathrm{T}$. For $i = 1, \ldots, k$, each T_i publishes $g^{s_\mathrm{A} s_\mathrm{TAB}}$, and generates shares $s_{\mathrm{A}_{i1}}, \ldots, s_{\mathrm{A}_{in}}$ of a secret sharing scheme for Γ_U with secret s_{A_i}. For $j = 1, \ldots, n$, T_i sends $s_{\mathrm{A}_{ij}} s_\mathrm{TAB}$ to U_j.
4. Select the smallest minimal set $\{\mathrm{U}_1, \ldots, \mathrm{U}_l\} \in \Gamma_\mathrm{U}$. For $j = 1, \ldots, l$, each U_j publishes $g^{s_\mathrm{B} s_\mathrm{UAB}}$, and generates shares $s_{\mathrm{B}_{j1}}, \ldots, s_{\mathrm{B}_{jm}}$ of a secret sharing scheme for Γ_T with secret s_{B_j}. For $i = 1, \ldots, m$, U_j sends $s_{\mathrm{B}_{ji}} s_\mathrm{UAB}$ to T_i.

The only difference with respect to Protocol 2 is that not all the TTPs are involved in communication exchange during Steps 3 and 4. For the threshold splitting case considered in Protocol 2, the number of communications between TTPs is reduced from $2mn$ to $kn + lm$. The minor disadvantage with Protocol 4 is that some agreements have to be reached in advance regarding which minimal set is used in Steps 3 and 4, but if the communication saving is significant then this extra procedure will be worth the effort.

5.2 Using MTA-free Secret Sharing

Recall the idea of MTA-free secret sharing from Sect. 3. We propose a protocol which can be used if the following condition holds:

[C1] there exists an integer k such that Γ_T has a minimal set of size k and Γ_U contains a set of size k.

Protocol 5. Proceed as in Protocol 2 except replace Step 3 with:

3.1. Select the smallest minimal set $\{\mathrm{T}_1, \ldots, \mathrm{T}_k\} \in \Gamma_\mathrm{T}$ such that there exists a set $\{\mathrm{U}_1, \ldots, \mathrm{U}_k\} \in \Gamma_\mathrm{U}$. For $i = 1, \ldots, k$, each T_i publishes $g^{s_\mathrm{A} s_\mathrm{TAB}}$ and sends $s_{\mathrm{A}_i} s_\mathrm{TAB}$ to U_i;
3.2. $\mathrm{U}_1, \ldots, \mathrm{U}_k$ establish an MTA-free secret sharing scheme for Γ_U by generating shares that correspond to secrets $s_{\mathrm{A}_1} s_\mathrm{TAB}, \ldots, s_{\mathrm{A}_k} s_\mathrm{TAB}$ and distributing these shares appropriately among $\mathrm{U}_1, \ldots, \mathrm{U}_n$.

The number of communications from TTPs in A's domain to TTPs in B's domain has thus been reduced to k. If an equivalent condition applies in the reverse direction, namely that there exists an integer l such that Γ_U has a minimal set of size l and Γ_T contains a set of size l, then Step 4 can be similarly replaced. If, as in Protocol 2, Γ_T is (k, m)-threshold and Γ_U is (k, n)-threshold, then both conditions hold and the number of communications between TTPs in different domains is reduced from $2mn$ to $2k$.

The reduction of communications between TTPs from different domains in Protocol 5 comes at a cost of introducing communication between TTPs from the same domain. The number of such communications is given by the sum of the linkages of the MTA-free secret sharing schemes used in Steps 3 and 4. Thus to minimise this number we should again use MTA-free secret sharing schemes with low linkages (see [11] and Sect. 7).

We note that to achieve minimal linkages it is sometimes necessary for a user to broadcast his share to the other users [11]. This translates in Protocol 5 to one TTP U_i broadcasting $s_{\mathrm{A}_i} s_{\mathrm{TAB}}$ to the other TTPs U_j. As a broadcast is insecure we assume that all other parties can obtain $s_{\mathrm{A}_i} s_{\mathrm{TAB}}$. This information would seem of litle use to any other party except for user A who knows s_{A_i} and could thus determine s_{TAB}. It is not clear whether this is a problem because the main role of s_{TAB} is to provide a mask for the passing of s_{A_i} to the TTPs from the other domain. However, should it not be desirable that A can learn the value of s_{TAB} in this manner then the broadcasts, where appropriate, should be replaced by secure transmissions and hence the number of communications between TTPs from the same domain will be increased slightly.

In the event that [C1] only fails because $k > n$ then Protocol 5 can still be used. The extra TTPs T_i can either broadcast their shares in Step 3.1, or should send their information to some predetermined U_j, who either broadcasts the shares or securely distributes them among the other TTPs U_j. The latter method should be used if broadcasting raises the same previous concerns.

5.3 Factoring Access Structures

If [C1] is not satisfied then in certain cases it is still possible to make improvements. Firstly we recall a convenient representation of an access structure [16].

Example 2. Let Γ be $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\}, \{\mathrm{A}, \mathrm{B}, \mathrm{D}\}, \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$. The minimal sets are $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$ and $\{\mathrm{A}, \mathrm{B}, \mathrm{D}\}$ and we represent this as $\Gamma = \mathrm{ABC}+\mathrm{ABD}$. This notation is convenient because we can replace the right hand side by any equivalent logical expression and use this as an equally valid representation of the access structure. So, for instance, we can also write $\Gamma = \mathrm{AB}(\mathrm{C} + \mathrm{D})$.

Let Γ be an access structure that can be represented in the form $\Gamma = \Gamma_1 \Gamma_2 \ldots \Gamma_r$, where each Γ_i is an access structure (defined on some subset of participants). We refer to the Γ_i as *factors* of Γ. In Example 2, AB and $\mathrm{C} + \mathrm{D}$ are factors of Γ. We can use the factors of an access structure to obtain an efficient protocol if the following condition holds:

[C2] there exists a minimal set in Γ_{T} of size r and a factorisation of Γ_{U} of the form $\Gamma_{\mathrm{U}} = \Gamma_{\mathrm{U}_1} \ldots \Gamma_{\mathrm{U}_r}$.

Protocol 6. Proceed as in Protocol 2 except replace Step 3 with:

3. Select a minimal set $\{\mathrm{T}_1, \ldots, \mathrm{T}_k\} \in \Gamma_{\mathrm{T}}$ such that $\Gamma_{\mathrm{U}} = \Gamma_{\mathrm{U}_1} \ldots \Gamma_{\mathrm{U}_k}$. For $i = 1, \ldots, k$, each TTP T_i publishes $g^{s_{\mathrm{A}} s_{\mathrm{TAB}}}$ and generates shares $s_{\mathrm{A}_{i1}}, \ldots, s_{\mathrm{A}_{ir_i}}$ of a secret sharing scheme with access structure Γ_{U_i} and secret s_{A_i}. For $j = 1, \ldots, r_i$, T_i sends $s_{\mathrm{A}_{ij}} s_{\mathrm{TAB}}$ to each U_{i_j}, where Γ_{U_i} is defined on $\{\mathrm{U}_{i_1}, \ldots, \mathrm{U}_{i_{r_i}}\}$;

If the equivalent of [C2] holds from Γ_{U} to Γ_{T} then we can introduce a similar improvement to Step 4.

Example 3. Let Γ_{T} be $(2,3)$-threshold and Γ_{U} be $(3,3)$-threshold. Then we can write $\Gamma_{\mathrm{U}} = \Gamma_{\mathrm{U}_1}\Gamma_{\mathrm{U}_2}$, where Γ_{U_1} is $(2,2)$-threshold on $\{\mathrm{U}_1, \mathrm{U}_2\}$ and Γ_{U_2} is $(1,1)$-threshold on U_3. Then Step 3 becomes:

3. Choose $\{\mathrm{T}_1, \mathrm{T}_2\}$. TTP T_1 generates shares $s_{\mathrm{A}_{11}}, s_{\mathrm{A}_{12}}$ of a $(2,2)$-threshold scheme with secret s_{A_1}, sends $s_{\mathrm{A}_{11}} s_{\mathrm{TAB}}$ to U_1 and sends $s_{\mathrm{A}_{12}} s_{\mathrm{TAB}}$ to U_2. TTP T_2 sends $s_{\mathrm{A}_2} s_{\mathrm{TAB}}$ to U_2.

For the access structures in Example 3, the number of communications from TTPs T_i to TTPs U_j using Protocols 2, 4, and 6, are 9, 6 and 3, respectively.

6 Reducing Other Protocol Operations

6.1 Reducing User to TTP Communication

In Protocol 2, communication between users and their home TTPs takes place in Steps 1, 2, 5 and 6. The only way of 'reducing' the communication in Steps 1 and 2 is to make sure that m and n are chosen to be as small as possible. Note that if Γ_{T} and Γ_{U} are threshold access structures, as in Protocol 2, then the size of m and n is not an indication of the security level (this is determined by the threshold value k), but is rather an indication of the system flexibility.

In Steps 5 and 6 of (the generalised) Protocol 2, and in all other variants discussed in this paper, a trivial saving in user to TTP communication overheads can be made by nominating just one of the TTPs T_i and U_j to pass on the final value to users A and B respectively. For Protocol 2 this reduces the number of communications between users and their TTPs from $2(m+n)$ to $m+n+2$.

6.2 Reducing the Generation of Secret Sharing Schemes

In this section we consider minimising the generation of secret sharing schemes in Protocol 2. Note that these protocols do not necessarily involve less communication between TTPs from different domains than the protocols of Sect. 5.

In the simple case that $\Gamma_{\mathrm{T}} = \Gamma_{\mathrm{U}}$ we can remove the need for generation of any secret sharing schemes in Steps 3 and 4 of Protocol 2 as follows:

Protocol 7. If $\Gamma_{\mathrm{T}} = \Gamma_{\mathrm{U}}$ it is sufficient to replace Steps 3, 4 of Protocol 2 with:

3. Each TTP T_i computes s_{TAB}, publishes $g^{s_{\mathrm{A}} s_{\mathrm{TAB}}}$ and sends $s_{\mathrm{A}_i} s_{\mathrm{TAB}}$ to U_i.
4. Each TTP U_j computes s_{UAB}, publishes $g^{s_{\mathrm{B}} s_{\mathrm{UAB}}}$ and sends $s_{\mathrm{B}_j} s_{\mathrm{UAB}}$ to T_j.

The difference between Protocol 7 and the use of Protocol 4 for transfer between two (k,n)-threshold schemes is that Protocol 7 involves n communications between TTPs from different domains (just k for Protocol 4), but no communication between TTPs from the same domain, or generation of secret sharing schemes ($kn - (1/2)k(k+1)$ and k respectively for Protocol 4).

Let Γ be an access structure defined on a set $\mathcal{P}$. Then a *cumulative map* for Γ is a set $\mathcal{S}$ and a mapping α from $\mathcal{P}$ to the collection of subsets of $\mathcal{S}$, such

that for any $A \subseteq \mathcal{P}$, $\cup_{\mathrm{P} \in A}\, \alpha(\mathrm{P}) = \mathcal{S} \iff A \in \Gamma$. Let Γ^+ denote the collection of maximal sets that do not belong to Γ. Then in [16] it was shown that there exists a unique *minimal* cumulative map for Γ, where *minimal* is used in the sense that all other cumulative maps contain the minimal solution. The minimal cumulative map for Γ can be defined by letting $\mathcal{S} = \Gamma^+$ and then for any $\mathrm{P} \in \mathcal{P}$, $\alpha(\mathrm{P}) = \{B \in \Gamma^+ \,|\, \mathrm{P} \notin B\}$.

Example 4. Let $\Gamma = \mathrm{AB} + \mathrm{BCD}$. Then Γ^+ consists of the sets $B_1 = \{\mathrm{A}, \mathrm{C}, \mathrm{D}\}$, $B_2 = \{\mathrm{B}, \mathrm{D}\}$, $B_3 = \{\mathrm{B}, \mathrm{C}\}$. Thus let $\mathcal{S} = \{B_1, B_2, B_3\}$. The minimal cumulative map for Γ is $\alpha(\mathrm{A}) = \{B_2, B_3\}$, $\alpha(\mathrm{B}) = \{B_1\}$, $\alpha(\mathrm{C}) = \{B_2\}$, $\alpha(\mathrm{D}) = \{B_3\}$.

Note that we can easily construct a cumulative map for Γ which has $|\mathcal{S}| > |\Gamma^+|$ from the minimal cumulative map for Γ. The following protocol does not need the generation of any secret sharing schemes in Step 3. It can be used if:

[C3] there exists a minimal set in Γ_{T} that is at least the size of Γ_{U}^+.

Protocol 8. Proceed as in Protocol 2 except replace Step 3 with:

3. Select the smallest minimal set $\{\mathrm{T}_1, \ldots, \mathrm{T}_k\} \in \Gamma_{\mathrm{T}}$ such that $k \geq |\Gamma_{\mathrm{U}}^+|$. Let $\mathcal{S}$ (of size k) and α be a cumulative map for Γ_{U}. For $i = 1, \ldots, k$, each T_i publishes $g^{s_{\mathrm{A}} s_{\mathrm{TAB}}}$ and sends $s_{\mathrm{A}_i} s_{\mathrm{TAB}}$ to each U_i that belongs to $\alpha(\mathrm{U}_i)$;

If the equivalent of [C3] applies from Γ_{U} to Γ_{T}^+ then we can introduce a similar change to Step 4. As the number of TTPs U_i increases, the number of sets in Γ_{U}^+ generally increases at an exponential rate, so Protocol 8 is likely to be useful only if m is large, relative to n. For *connected* access structures Γ (every participant belongs to at least one minimal set of Γ), the number of sets in Γ^+ is at least the number of participants. Thus, generally [C3] only holds if [C1] holds. It follows that Protocol 8 is an alternative to Protocol 5 for use when minimising the generation of secret sharing schemes, rather than as an option for cases when [C1] does not hold.

7 Threshold Transferral

In this section Γ_{T} and Γ_{U} are threshold access structures. We aim to first minimise the number of communications between TTPs from different domains, and describe solutions that achieves this, while also keeping communication between TTPs from the same domain, and generation of secret sharing schemes, to a minimum. We recall a result from [11]:

Result 9. The minimum linkage for an MTA-free (k, n)-threshold scheme is $nk - (1/2)k(k+1)$, and can be realised by the *Contraction Construction* [11].

Example 1 has the minimum possible linkage of 3 for a $(2, 3)$-threshold scheme and is an example of application of the Contraction Construction. There are three cases of threshold transferral to consider:

Case 1. (k,m)-*threshold to* (n,n)-*threshold.* Use Protocol 6. This needs a total of $\max k, n$ communications between TTPs from different domains, involves no communications between TTPs from the same domain, and needs generation of one secret sharing scheme (by one of the sending TTPs) if $k < n$.

Case 2. (k,m)-*threshold to* (l,n)-*threshold,* $(k \geq l,\ l < n)$. Use Protocol 5. This needs k communications between TTPs from different domains. Using the Contraction Construction it needs $nk - (1/2)k(k-1)$ communications between (receiving) TTPs from the same domain, and needs generation of k secret sharing schemes (by receiving TTPs).

Case 3. (k,m)-*threshold to* (l,n)-*threshold,* $(k < l < n)$. Use Protocol 4. This needs kn communications between TTPs from different domains, involves no communication between TTPs from the same domain, and needs generation of k secret sharing schemes (by sending TTPs).

Given that the complete version of any of the described protocols involves a transfer from the TTPs T_i to the TTPs U_j and then from the TTPs U_j to the TTPs T_i, we identify in Table 1 five cases for consideration over a complete protocol (where in each direction one of Cases 1, 2 and 3 applies). For each Type

Table 1. Types of Threshold Transferral

Type	**Transfer between thresholds**			**Conditions**
1	(m,m)	and	(n,n)	$m \neq n$
2	(m,m)	and	(l,n)	$l < m,\ l < n$
3	(m,m)	and	(l,n)	$m < l < n$
4	(k,m)	and	(l,n)	$k < m,\ l < k,\ l < n$
5	(k,m)	and	(k,n)	

shown in Table 1 we compare the use of the Protocols suggested in Cases 1, 2 and 3 with Protocol 2. The necessary number of communications are given in Table 2. As can be seen from Table 2, for each type the number of communications

Table 2. Efficiencies of Threshold Transferral

	Using Protocol 2			**Using Cases 1,2,3**		
Type	*Diff Dom*	*Same Dom*	*SSS Gen*	*Diff Dom*	*Same Dom*	*SSS Gen*
1	$2mn$	0	$m+n$	$2\max m, n$	0	1
2	$2mn$	0	$m+n$	$m + \max l, m$	$ln - (1/2)l(l+1)$	$l+1$
3	$2mn$	0	$m+n$	$mn + \max l, m$	0	m
4	$2mn$	0	$m+n$	$k+lm$	$ln - (1/2)l(l+1)$	$2l$
5	$2mn$	0	$m+n$	$2k$	$k(m+n-k-1)$	$2k$

between TTPs from different domains when using Cases 1, 2 and 3 is no more, and usually significantly less, than if Protocol 2 had been used. Further the

total number of communications between TTPs (from the same and different domains) is generally less than for Protocol 2. The number of secret sharing schemes needing generated is always less than for Protocol 2.

Note that the use of Protocol 8 is not recommended for transfer between threshold access structures (when [C3] applies). The reason is that the sizes of set $\mathcal{S}$ and the sizes of the sets $\alpha(P_i)$ are relatively large for threshold access structures (compared to other access structures on the same number of participants). In all cases the simple change suggested in Sect. 4.1 can be used to reduce communication between users and their home TTPs.

References

1. J. Benaloh. Secret sharing homomorphisms: keeping shares of a secret secret. *Adv. in Cryptology - CRYPTO'86, Lecture Notes in Comput. Sci*, **263** (1987) 251–260.
2. G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS 1979 National Computer Conference*, **48** (1979) 313–317.
3. E. F. Brickell. Some ideal secret sharing schemes. *J. Combin. Math. Combin. Comput.*, **9** (1989) 105–113.
4. L. Chen, D. Gollmann and C.J. Mitchell. Key escrow in mutually mistrusting domains. *Proceedings of 1996 Cambridge Workshop on Security Protocols, Lecture Notes in Comput. Sci.*, **1189** (1997) 139–153.
5. D.E. Denning and D.K. Branstad. A taxonomy for key escrow encryption systems. *Communications of the ACM*, **39**(3) (1996) 34–40.
6. Y. G. Desmedt and Y. Frankel. Homomorphic zero-knowledge threshold schemes over any finite abelian group. *SIAM J. Disc. Math.*, **4** (1994) 667–679.
7. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, **22** (1976) 644–654.
8. M. van Dijk, On the information rate of perfect secret sharing schemes, *Des. Codes Cryptogr.*, **6** (1995) 143–169.
9. Y. Frankel and Y. Desmedt. Classification of ideal homomorphic threshold schemes over finite Abelian groups. *Adv. in Cryptology – EUROCRYPT'92, Lecture Notes in Comput. Sci.*, **658** (1992) 25–34.
10. I. Ingemarsson and G. J. Simmons. A protocol to set up shared secret schemes without the assistance of a mutually trusted party. *Adv. in Cryptology – EUROCRYPT'90, Lecture Notes in Comput. Sci.*, **473** (1991) 266–282.
11. W.-A. Jackson, K.M. Martin and C.M. O'Keefe. Mutually Trusted Authority free Secret Sharing Schemes. *J. Cryptology*, to appear.
12. N. Jefferies, C. Mitchell and M. Walker. A proposed architecture for trusted third party services. *Cryptography: Policy and Algorithms, Lecture Notes in Comput. Sci.*, **1029** (1996) 98–104.
13. N. Jefferies, C. Mitchell and M. Walker. Practical solutions to key escrow and regulatory aspects. *Proceedings of Public Key Solutions '96*, Zurich, September 1996.
14. S. Micali. Fair cryptosystems. *MIT Technical Report*, MIT/LCS/TR-579.c, (1994).
15. A. Shamir. How to share a secret. *Comm. ACM*, **22**(11) (1979) 612–613.
16. G. J. Simmons, W.-A. Jackson, and K. Martin. The geometry of shared secret schemes. *Bull. Inst. Combin. Appl.*, **1** (1991) 71–88.

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