July 2000

# Multiparty key agreement protocols

J. Pieprzyk
*University of Wollongong*

C. H. Li
*University of Wollongong*

# Multiparty key agreement protocols


## Abstract

A class of multiparty key agreement protocols based on secret sharing is presented. The trust infrastructure necessary to achieve the intended security goals is discussed.


## Disciplines

Physical Sciences and Mathematics


## Publication Details

This paper originally appeared as: Pieprzyk, J and Li, CH, Multiparty key agreement protocols, IEE Proceedings - Computers and Digital Techniques, July 2000, 147(4), 229-236. Copyright IEEE 2000.

# Multiparty key agreement protocols

J.Pieprzyk and C.-H.Li

**Abstract:** A class of multiparty key agreement protocols based on secret sharing is presented. The trust infrastructure necessary to achieve the intended security goals is discussed. Entity authentication is suggested to be replaced by a less expensive group authentication. Two key agreement protocols are discussed. The first is the group key agreement where all principals must be active to call the conference. The other allows a big enough subgroup (controlled by the threshold parameter $t$) to trigger the conference. It is proved that the protocols achieve key freshness, key confidentiality, group authentication and key confirmation. A discussion about possible modifications and extensions of the protocol concludes the paper.

## 1 Introduction

Most cryptographic tools can only be used if appropriate cryptographic keys are known to the parties. The intrinsic difficulty of key establishment in large computer networks has led to the invention of public-key cryptography where one of the keys can be made public, considerably simplifying the problem of key establishment. There are two well known categories of key establishment protocols: key transport and key agreement. Key transport protocols enable two communicating parties to obtain a common secret key by using pre-established secure communication channels between them and a trusted third party (TTP). Normally, the TTP is responsible for the generation of a fresh secret key and the parties gratefully accept it.

Key agreement protocols, on the other hand, allow the parties to interact with each other so that, they can derive a common secret key. Moreover, they exercise equal influence on the final form of the secret. Although the TTP is not directly involved in the protocol, its existence is crucial as it provides the public keys of the parties, typically, in the form of proper certificates (or public keys signed by the TTP). The TTP has no access to the secret agreed between the parties. That is why, in some applications, key agreement is preferred to key transport.

A natural evolution of cryptography has given rise to the so-called multiparty cryptography (also termed group or society oriented) with a key establishment protocol as its integral part. Traditionally, the multiparty key establishment is also called conference key establishment.

The very first key transport protocol was published by Needham and Schroeder [1] in 1978. There are three players in it: two principals and a TTP. The principals share secure channels with the TTP and the channels are used in the protocol to distribute the secret fresh key generated by the TTP. It turned out that even this relatively modest protocol goal was not attainable in the protocol. If a transcript of the communications together with the secret key is lost and is then found by an attacker, then the attacker may pretend to be one of the parties and the other principal has no way to tell apart messages generated in the past from the present ones. In other words, the key freshness is not guaranteed.

Public key cryptography was invented by Diffie and Hellman [2]. The validity of the concept was illustrated on an example showing how two principals may agree on a secret key via a public negotiation. This is a classical key agreement protocol. Unfortunately, the main weakness of it was the lack of principals' authentication : they can establish a secret key but they do not know with whom. This is to say that the protocol is subject to the man-in-the-middle attack. Diffie, *et al.* [3] modified the original Diffie–Hellman protocol in which both principals are able to authenticate each other. To implement it, a TTP must provide authentic parameters of the principals.

The experience with key establishment protocols illustrates, sometimes very dramatically, that the design of 'secure' protocols is not easy. The main difficulty seems to be a very vague definition of what we expect from secure protocols. The expected security characteristics of the protocol are called *security goals*. The security goals must be defined well before the construction of the protocol. There is a generic collection of security goals which typically includes key freshness, key confidentiality, principal authentication and key confirmation.

Multiparty key establishment can be seen as a generalisation of two-party key establishment. The first multiparty key agreement was published by Ingemarsson *et al.* [4] which is a generalisation of the Diffie–Hellman protocol. Fiat and Naor [5] considered key agreement in the context of broadcast encryption where the messages are to be decrypted by groups. Burmester and Desmedt [6] showed that if the group is able to structure itself into a ring, then after two broadcasts per principal, the group is able to derive a common secret key. Just and Vaudenay [7] showed that the Burmester–Desmedt protocol fails to provide entity authentication and suggested a protocol in which two-party authentication is extended into the group authentication.



*IEE Proceedings* online no. 20000531

*DOI:* 10.1049/ip-cdt:20000531

Paper first received 16th March 1999 and in revised form 29th February 2000

The authors are with the Centre for Computer Security Research, School of Information Technology and Computer Science, University of Wollongong, Northfields Avenue, Wollongong 2522, Australia
E-mail: {josef,cl14}@cs.uow.edu.au

## 2 Multiparty cryptography

Secret sharing has become an indispensable cryptographic tool whenever the control over execution of a cryptographic operation is assigned to a group rather than to an individual. Blakley [8] and Shamir [9] considered a key management system in which the secret is collectively held by a group of $n$ principals allowing any $t$ of them to recover the secret ($t \leq n$). The access structure of the secret sharing determines the collection of all subsets of principals who are authorised to recover the secret. One of the simplest access structure is a $(t, n)$ threshold scheme when any collection of $t$ or more principals are authorised to recover the secret.

Secret sharing is set up by an algorithm called the dealer. It has to be run by a trusted party. For a given secret, it produces shares of the secret and distributes them to principals via secure channels. To recover the secret, a currently active subset of shareholders pools the shares and recovers the secret if it belongs to the access structure, otherwise it fails with an overwhelming probability. The recovery of the secret is typically implemented as the combiner algorithm. It can be run collectively or by a trusted party (for example one of the active principals who is trusted by others). After secret recovery, the combiner distributes the secret via secure channels to all active principals. A good tutorial on secret sharing and the vocabulary used can be found in [10].

Secret sharing allows groups to define (via its access structure) who is authorised to recover the secret. If a cryptographic operation is activated by a proper secret key, then to allow a group control over it, it is enough to give shares of the secret using a secret sharing scheme with a properly defined access structure. It should be no surprise that secret sharing should be of great help in designing conference key establishment protocols. Some of attractive features of secret sharing are listed below.

(1) Access structure gives a convenient way to differentiate principals and their power within the group. This could reflect the amount of trust assigned to each principal or perhaps, the place of the principal in the organisation. If all principals are equally trusted, or perhaps we are dealing with a democratic organisation, then a threshold scheme is appropriate.
(2) Delegation is possible if a principal who holds her share passes permanently or temporarily her share to a delegated person or a group of people.
(3) Secret sharing used can reflect the formal parameters of a conference, indicating how large a subset of active principals has to be to call the conference. Again, if the threshold secret sharing is acceptable, then the selection of the threshold enables one to manipulate the size of the group that is able to call on the conference.
(4) Secret sharing can be immunised against the loss of shares by making it proactive with a share refreshment protocol [11].
(5) Group authentication can replace principal authentication. Group authentication is a weaker requirement and, in general, can be less expensive to achieve. This is the case when principals do not need to know the precise composition of the currently active group but they need to be sure that the group is large enough to conduct a valid conference.
(6) Cheating detection, well developed in secret sharing, can be used to detect principals who misbehave during the protocol execution.

Clearly, secret sharing has also some characteristics which restrict its applicability for key establishment protocols. The two most serious are: first, the group who wish to call the conference must be known well ahead of the conference. The group can be composed by a trusted dealer or collectively by all participants. Secondly, shares have to be distributed to the principals via secure channels. Typically the groups involved in the conference are known well in advance and their memberships are fixed for some time so the first characteristics seems not to be a problem. Moreover, secret sharing has, already developed methods and techniques to deal with modifications of the group (enrolment and disenrollment [10]). The second feature is unavoidable but can be dealt by conversion of secret sharing into the conditionally secure setting in which shares are communicated via less expensive broadcast channels.

## 3 Trust infrastructure

A key establishment protocol can be seen as a cryptographic tool which allows one to extend an initial trust which exists between principals $A$, $B$ and their TTP to a trust between the two principals. Needless to say, the existence of trust is the necessary condition for any key establishment protocol to work correctly and to achieve the intended security goals. For instance, Needham and Schroeder [11] used a TTP who generated a fresh session key and transported it to principals via secure channels. In their key agreement protocol, Diffie *et al.* [3] assumed that any principal had the access to the other party's authenticated public keys. The authenticated public keys were displayed by a TTP in the form of certificates signed by the TTP. Anyone who knew the corresponding (authentic) public key of the TTP, could verify certificates.

There are three elements of trust infrastructure in secret sharing: the dealer, the combiner and secure channels. Now we discuss these components.

### 3.1 Dealer

The role of dealer can be considered in the context of key transport and key agreement. In key transport, the dealer algorithm can be run by a TTP or a conference chair who composes the principals into a group who are eligible to call a conference. Next the chair determines a proper access structure which reflects the position of each principal in the group and clearly identifies the smallest subgroups which are still eligible to call on the conference. Finally, the chair generates a fresh secret and divides it into shares which are secretly communicated to the principals.

In key agreement, a TTP is a passive entity whose role is restricted to the delivery of authentic parameters of the principals (their public keys) on demand. The dealer must therefore be run collectively by principals or, in other words, every principal plays the role of dealer. Each principal sets up her own secret sharing for the group of her choice. Note that each principal has full control over the access structure of her secret sharing. The shares are next distributed secretly to the other principals. After all the principals have distributed their shares, each principal possesses her own share plus shares obtained from others. Finally, each principal combines all shares into one, hoping that the resulting secret sharing has an access structure which is acceptable to all.

It is not difficult to notice that this approach can only work if all the principals use the same type of secret sharing which allows many instances of secret sharing generated locally to merge into one (without a central dealer). A broad class of secret sharing which allows one to do this is linear schemes. Even dealing with linear secret sharing does not solve the problem of different access

structures selected by individual principals. We know however, that if each principal selects a $(t, n)$ Shamir threshold scheme and distributes shares to the same collection of principals, then the resulting scheme is also a threshold scheme. It is easy to check that if each principal selects a different threshold but the collection of principals is the same for all, then the threshold of the composed sharing is the largest used by principals (with an overwhelming probability).

### 3.2 Combiner

The role of the combiner in secret sharing is to collect shares from principals and if the currently active subset belongs to the access structure, then the combiner can recover the secret and communicate it to the principals via secure channels.

In key establishment protocols, the role of the combiner needs to be redefined. Note that if we assume that the combiner is trusted, then the principals do not need to send their shares as the combiner can generate a fresh key without interaction with the principals. The purpose of secret sharing is to recover the key while in key establishment protocols any fresh key is good.

For key transport, there is a chair who designs a secret sharing of threshold 2 for a fresh secret. Each principal receives a single share while the chair holds the secret and one extra share. The extra share is used to trigger the conference by broadcasting it (broadcasting must be authenticated). Each principal, takes her share plus the one broadcast and recovers the secret key. Observe that each principal plays the role of combiner.

Assume that there is no chair and the trusted dealer does not participate in conferences but sets up a secret sharing with a fresh key. If the secret sharing has the threshold $n+1$ and the number of all shares is $3n$ ($n$ is the number of all eligible principals) and each principal is assigned three shares, then to call on a conference, it is enough if $n$ principals broadcast their shares. Knowing $n$ shares, each principal can recover the secret key using her second share. The third share can be applied to verify the validity of the secret (or cheating detection). Clearly, a misbehaving principal can broadcast two or three shares instead of one. If a principal broadcasts two shares, she can recover the secret but cannot verify it. If she announces three shares, she cannot participate in the conference.

Consider the role of combiner in the context of key agreement protocols. Assume that a $(n+1, 2n)$ secret sharing is set up collectively by all $n$ principals so the threshold is $(n+1)$ and each principal holds two shares. To call a conference, it is enough for the principals to broadcast their single shares. After the announcement of $n$ shares, each principal can apply the second share to recover the secret (the threshold is $(n+1)$).

The situation becomes more interesting if the call for a conference can be made by any $t$ out of $n$ principals.

### 3.3 Communication channels

Interaction among the principals is done via different communication channels.

Confidentiality channels are very expensive to set up and use. Messages are encrypted before transmission so that any outsider who gains access to the channel is unable to understand them. Confidentiality channels can be implemented using symmetric or asymmetric cryptography. In the case of symmetric cryptography, both the sender and the receiver know the same cryptographic key. In asymmetric (or public key) cryptography, the sender key is public but the receiver's key is secret. Note that the sender must make sure that the key is the authentic public key of the intended receiver.

Authenticity channels are typically less expensive as messages are communicated in plain and a relatively short authentication string is attached to them. Typically, the receiver can detect whether or not a message comes from the correct source and has not been tampered with during transmission. Authentication strings can be implemented using either digital signatures (every one can verify whether or not the signatures match the messages and their alleged sender), or message authentication code (MAC) where only the holders of the secrets which were used to produce the MAC can verify the validity of pairs: messages and their MACs.

Broadcast channels are relatively cheap to implement. The sender may set up her publicly accessible billboard (a web page) on which she announces messages allowing the interested parties to fetch messages when they need them. Clearly, messages displayed on the billboard can be authenticated by appending signatures to them.

A secure channel is one that provides both secrecy and authenticity. All interactions in key transport protocols are performed via secure channels implemented using either secret-key or public-key cryptosystems. This was the case for Needham–Schroeder protocols and their successors [1]. Key agreement protocols are normally supported by public-key cryptosystems and broadcasting seems to be the predominant way of message communication [12].

## 4 Security goals

Security goals may vary but there is a relatively small collection of goals which are achievable in most conference key establishment protocols. Additionally, it is desirable that the protocol can be executed efficiently. The main collection of security goals for key establishment protocols are [12]: first, key freshness. The key has not been generated or used before in any other conference. Typically, to ensure key freshness it is sufficient to generate the key at random from a large population so the probability of reusing some already generated keys, is negligible. Secondly, entity authentication, this is a confirmation process which allows one principal to correctly identify the others involved in the protocol. Typically, it allows a principal to check whether other principals are active (alive) at the time when the protocol is being executed. This requirement can be relaxed by defining group authentication in which every principal is sure that all the principals are alive and present. This allows any principal to identify the group rather than individuals. Weak group authentication means that all currently active principals are sure that there is a large enough group of active principals. In most circumstances, a conference is considered to be valid if a quorum of principals is present. The access structure (or the threshold parameter) conveniently determines the size of a large enough group. Thirdly, key confirmation, this is a property of the protocol which allows one principal to make sure that the other parties possess the same common key. This is typically achieved by using the so-called handshaking or challenge-response interaction. The idea is to generate a random challenge, encrypt it using the key which needs to be confirmed, and to expect from the other party the correct response, which is agreed before public transformation (say squaring the modulo of some prime). Forthly, implicit key authentica-

tion this provides an assurance to principals that no one except specific other parties could have gained access to the common key. Implicit key authentication can be also viewed as key confidentiality. Finally, explicit key authentication means that both implicit key authentication and key confirmation hold.

In key agreement protocols, one would expect that the control over the final form of the key is distributed over the principals and there is no way that a subgroup of conspiring principals can force others to accept the key of their choice. This goal can be termed group key control.

## 5 A new multiparty key agreement protocol

The proposed key agreement applies secret sharing generated independently by principals. Assume that there are $n$ principals $P_1, \ldots, P_n$ who are eligible to participate in a conference. Each principal $P_i$; $i = 1, \ldots, n$, creates her own Shamir secret sharing defined by a polynomial $f_i(z)$ and distributes shares to the members of the group using prearranged secure channels. The combined secret sharing is defined by the polynomial

$$F(z) = \sum_{i=1}^{n} f_i(z)$$

Note that the secret $F(0)$ generated collectively by the group is not known until the principals decide to pool their shares to recover the secret. Moreover, the principals contribute 'equally' to the fresh secret. The protocol progresses through three major phases:

(1) Registration: each principal who wants to join the conference registers herself with a trusted registry
(2) Initialisation: each principal creates her private secret sharing scheme and distributes shares to all other principals
(3) Call for conference: principals broadcast their shares and therefore enable themselves to recover a common secret key.

### 5.1 Assumptions

Our assumptions are listed as follows:

(i) there are $n$ principals $\{P_1, \ldots, P_n\}$ who want to joint the conference,
(ii) there exists a trusted registry (R) who manages the registration of principals. In particular, the registry keeps a list of public keys of principals,
(iii) public information accessible from the registry is authenticated by the registry. Typically, information is accessible in a form of certificates signed by R,
(iv) secure channels provide both secrecy and authentication. Broadcast channels deliver authenticated messages to all principals (messages can be read by all but nobody can modify them without detecting the modification).

Let $p$ and $q$ denote large primes such that $q$ divides $p - 1$. Let $G_q$ be a subgroup of $Z_p^*$ of order $q$ and $g$ be a generator of $G_q$.

### 5.2 Registration

The principal $P_i$ chooses her own private key $x_i \in Z_q^*$ and submits her public key $h_i = g^{x_i} \pmod p$ for $i = 1, \ldots, n$ to the registry R. When all the principals have completed their registration, the registry R displays a read-only list of public keys together with principals' names. Additionally, R generates a random integer $r \in_R Z_q^*$ on demand and displays $\alpha = g^r$ for a short period of time. Normally, the value is generated whenever a need for conference arises (indicated by the principals who wish to call a conference). This value is erased after some time (when the conference has finished). The same value $\alpha$ is never used in two different conferences.

Registration serves three purposes. The first one is that each principal knows the other principals who are to join the conference. The second one is that the public keys can be used to implement secure channels between the principals. For example, the information provided by the registry is enough to encrypt a message using the ElGamal cryptosystem. Assume that $m \in Z_q^*$ and $P_i$ wants to send the message to $P_j$ in encrypted form. First, $P_i$ chooses a random integer $v \in Z_q^*$ and computes $g^v$, $h_j^v$ and $m \times h_j^v$. The pair $(g^v, m \times h_j^v)$ is sent to $P_j$. The receiver $P_j$ takes the pair and computes $(g^v)^{x_j} = g^{vx_j}$ which can later be used to extract the message $m = m \times h_j^v \times g^{-vx_j}$. The third purpose is to supply the principals with fresh (random) elements $\alpha$ which are later used in the protocol.

### 5.3 Initialisation

This phase of the protocol is executed independently by each principal and proceeds as follows:

(1) $P_i$ designs a $(n+1, 2n)$ Shamir thresold scheme, i.e. a scheme with $2n$ shares and with threshold $n+1$. Let the scheme be defined by a random polynomial $f_i(z)$ of degree at most $n$. Suppose that

$$f_i(z) = a_{i,0} + a_{i,1}z + \ldots + a_{i,n}z^n$$

where coefficients $a_{i,j} \in Z_q^*$ are chosen at random for $j = 1, \ldots, n$. As usual in Shamir scheme, shares are computed for $2n$ public $z$ coordinates. We assume that $P_i$ is assigned a pair of coordinates $z_i = (2i - 1, 2i)$.
(2) Next $P_i$ prepares pairs of shares $s_{i,j} = f_i(z_j) = (s_{i,j}^{(1)} = f_i(2j - 1), s_{i,j}^{(2)} = f_i(2j))$.
(3) Finally, $P_i$ communicates $s_{i,j}^{(2)}$ to the principal $P_j$; $j = 1, \ldots, n; j \neq i$ via a secure channel. In effect, $P_i$ obtains a sequence of $n$ elements $(s_{1,i}^{(2)}, \ldots, s_{n,i}^{(2)})$ and computes her secret share $S_i^{(2)} = \sum_{j=1}^{n} s_{j,i}^{(2)}$ where $S_i^{(2)} = F(2i)$ and the polynomial $F(z) = \sum_{i=1}^{n} f_i(z)$.

Note that the secret $s = F(0) = \sum_{i=1}^{n} a_{i,0}$ is never exposed to principals. From now on $s = F(0)$ will be called a seed to differentiate it from a fresh secret key obtained by all the principals involved in the conference.

### 5.4 Call for a conference

To start the conference, the principals execute the following steps:

(1) $P_i$ contacts the registry and fetches necessary parameters including $\alpha = g^r$ (the registry selects $r$ at random). If the element $\alpha$ is not on display, $P_i$ asks R for one
(2) $P_i$ prepares public shares $\beta_{i,j} = \alpha^{s_{i,j}^{(1)}}$ for $j = 1, \ldots, n$
(3) $P_i$ broadcasts $\beta_{i,j}$ to all the principals $j = 1, \ldots, n$
(4) After $P_i$ has obtained $\beta_{j,i}$ from other principals, she recovers $n$ public shares

$$\alpha^{S_j^{(1)}} = \alpha^{F(2j-1)} = \alpha^{\sum_{i=1}^{n} s_{i,j}^{(1)}} = \prod_{i=1}^{n} \beta_{i,j}$$

for $j = 1, \ldots, n$
(5) $P_i$ uses $n$ public shares and her secret share, $S_i^{(2)}$ to recover the common secret $S = \alpha^{F(0)} = \alpha^s$. Note that the principals still use the Lagrange interpolation but for

exponents so

$$S = \alpha^{F(0)} = (\alpha^{S_1^{(2)}})^b \prod_{j=1}^{n} (\alpha^{S_2^{(1)}})^{b_j}$$

where

$$b = \prod_{j=2,4\ldots,2n;j\neq 2i} \frac{j}{j-2i}$$

and

$$b_j = \prod_{\ell=1,3,\ldots,2n-1;\ \ell\neq 2j-1} \frac{\ell}{\ell-2j+1}$$

are Lagrange coefficients

(6) $P_i$ takes the secret $S$, her name $id_i$, and $\alpha$ and prepares a string $\sigma_i = H(S\|id_i\|\alpha)$ where $H$ is a cryptographically strong, collision resistant hash function with a public description. The triplet $(\sigma_i, id_i, \alpha)$ is broadcast (note that the broadcasting channel is assumed to provide authentication).

(7) $P_i$ collects $(\sigma_j, id_j, \alpha)$ from other principals, checks their authenticity and verifies them using her own secret $S$. If the checks hold, $P_i$ is ready for the conference. Otherwise, $P_i$ announces the error and aborts the protocol.

### 5.5 Security analysis

The following theorem describes which security goals are achievable by the protocol.

*Theorem 1:* Assume that the protocol is run by a group of honest principals, then the protocol attains the following security goals: (1) key freshness, (2) key confidentiality, (3) group authentication, (4) key confirmation.

*Proof:* (1) The registry displays an integer $\alpha = g^r$ for a random $r \in_R Z_q^*$. Note that the common secret key $S = g^{rs} = \alpha^s$ is fresh as long as $r$ is fresh. The freshness is probabilistic.

(2) Key confidentiality holds as after broadcasting the shares $\alpha^{s_{i,j}^{(1)}}$, all outsiders know $n$ public shares only. As the Shamir scheme is perfect, it means that $n$ shares do not provide any information about the secret when the threshold is $n+1$. The perfectness argument can only be used if the secret sharing is used once. For a multiple use, which is the case, the principals should be sure that the threshold of the group secret sharing is exactly $(n+1)$. A simple way to decide whether the threshold is $(n+1)$, is to check if the secret derived according to the protocol by a principal is the same as the value obtained by the Lagrange interpolation of public shares only. If the two values are the same the threshold is not equal to $(n+1)$. It is easy to verify that the probability of the threshold being $(n+1)$ is $(1-q^{-1})$.

(3) Group authentication (by contradiction). Assume that the protocol has been successful and the group authentication does not hold. From this assumption we will derive that instances of the discrete logarithm are easy to invert (which is the requested contradiction). From our assumptions we know that there is at least one principal, say $P_j$, who has not participated in the protocol. As the threshold of the secret sharing is $(n+1)$, somebody had to broadcast the prescribed collection of public shares $\beta_{j,i} = \alpha^{s_{j,i}^{(1)}}; i = 1, \ldots, n$, on behalf of $P_j$. This can be done only if either the shares $s_{j,i}^{(1)}$ can be extracted from the public shares announced in the previous runs or the random $r$ can be extracted from $\alpha = g^r$. This leads us to the conclusion that the discrete logarithm instances used are easy, which is a contradiction.

(4) After the conference has been called, every principal can check whether other principals are holding the same secret by verifying the triplets $(\sigma, id_j, \alpha)$ for all $j \neq i$. The key confirmation is satisfied. This completes the proof.

What if a subgroup of the principals does not follow the protocol? Let us consider the following possibilities:

(1) At the initialisation stage, the subgroup can intentionally lower the thresholds used in their private secret sharing schemes. This does not effect the work of the protocol as if at least one principal is honest, the threshold will be random and equal to $(n+1)$ with the probability $(1-q^{-1})$. The subgroup of conspirators can establish a conference but they compromise the confidentiality of their secret (see Theorem 2).

(2) At the initialisation stage, the subgroup can intentionally increase the threshold of their private schemes, then at the call for conference stage, the principals who are honest will recover inconsistent secrets and will abort the conference (see Theorem 3).

(3) At the call for conference stage, the subgroup can broadcast modified shares of their private schemes. This will be detected by honest principals when the secret is verified.

(4) A disobedient principal $P_i$ can make public her secret sharing scheme (the polynomial $f_i(z)$). The conference can still be called but without involvement of $P_i$. This is another way of saying call conference whenever you wish. $P_i$ can still participate in conferences if her share $S_i^{(2)}$ remains secret. If $P_i$ goes further and discloses $S_i^{(2)}$, then the secret key becomes public if the rest of the principals follows the protocol. Otherwise if some principals refrain from broadcasting their public shares, the conference will not go ahead.

*Theorem 2.* Given a group of n principals $P_1, \ldots, P_n$ who participate in the protocol. Assume that there is a subgroup $P_1, \ldots, P_k$ of principals who lowered the threshold of their private secret sharing schemes so

$$\deg f_i(z) = k$$

for $i = 1, \ldots, k$ and $k < n$. Then the subgroup can work with their own secret sharing based on the polynomial

$$G(z) = \sum_{i=1}^{k} f_i(z)$$

with the secret $S' = \alpha^{G(0)}$. This secret is known to the whole group $P_1, \ldots, P_n$ and indeed to all outsiders.

*Proof:* If all principals broadcast their public shares, then each principal can compute the secret $S = \alpha^{F(0)}$ where $F(z) = \sum_{i=1}^{n} f_i(z)$. A principal $P_i$; $i = 1, \ldots, n$, can compute $S' = \alpha^{G(0)}$ by simply ignoring all information obtained from participants not belonging the the subgroup. In particular, $P_i$ computes her secret share

$$\tilde{S}_i^{(2)} = \sum_{j=1}^{k} s_{j,i}^{(2)}$$

and at the call for conference stage, calculates the following $n$ public shares

$$\alpha^{\tilde{S}_j^{(1)}} = \prod_{i=1}^{k} \beta i, j.$$

As the secret sharing of the subgroup has the threshold $(k+1)$, the Lagrange interpolation gives the same secret $S' = \alpha^{G(0)}$ for any subset of $k$ public shares. The subgroup

can agree on the secret which is known to the whole group. Moreover, an outsider can recover the secret $S' = \alpha^{G(0)}$ from any $(k+1)$ public shares. The subgroup may have a conference which is public.

*Theorem 3* Given a group of $n$ principals $P_1, \ldots, P_n$ who participate in the protocol. Assume that there is a dishonest principal $P_1$ who deviates from the protocol by selecting her random polynomial $f_1(z)$ of degree $(n+1)$. Then honest principals $P_i$; $i = 2, \ldots, n$ detect this at the first run of the protocol during the key confirmation stage.

*Proof:* Without the loss of generality, we ignore exponentiation so, in other words, in the call for conference stage the principals broadcast their shares $s_{i,j}^{(1)}$ (instead of prescribed $\alpha^{s_{i,j}^{(1)}}$). The proof is conducted by contradiction. Assume that there are two principals who recover the same secret. Let them be $P_2$ and $P_3$. The secret sharing created by the group in the initialisation stage is defined by polynomial

$$F(z) = \sum_{i=1}^{n} f_i(z).$$

As $\deg f_1(z) = n+1$, the degree of $F(z)$ is also $(n+1)$. $P_2$ applies the Lagrange interpolation for the following $(n+1)$ points:

$$(4, S_2^{(2)}), (1, S_1^{(1)}), \ldots, (2n-1, S_n^{(1)})$$

where $S_j^{(1)} = F(2j-1)$ and $S_2^{(2)} = F(4)$ and finds a polynomial $G_2(z)$. Similarly, $P_3$ knows

$$(6, S_3^{(2)}), (1, S_1^{(1)}), \ldots, (2n-1, S_n^{(1)})$$

where $S_3^{(2)} = F(6)$ and determines a unique polynomial $G_3(z)$ which contains the points. Both $G_2(z)$ and $G_3(z)$ are of degree at most $n$. If both $P_2$ and $P_3$ recover the same secret, it means that $G_2(0) = G_3(0)$. As the polynomials $G_2(z)$ and $G_3(z)$ contains $(n+1)$ common points ((1, $S_1^{(1)}$), ..., $(2n-1, S_n^{(1)})$ and $(0, G_2(0))$), they have to be identical so $G_2(z) = G_3(z)$. On the other hand, knowing $(n+2)$ points

$$(4, S_2^{(2)}), (6, S_2^{(2)}), (1, S_1^{(1)}), \ldots, (2n-1, S_n^{(1)})$$

one could find $F(z)$ using the Lagrange interpolation. Note that these points also belong to $G_2(z)$ so $F(z) = G_2(z)$. This implies that deg $F(z) = n$ which is our requested contradiction. This completes the proof.

We claim that the protocol can be used repeatedly to call conferences as the seed $s$ remains secret and to recover the fresh secret key $S$, the principals need to use secret sharing to compute it.

Recall that the discrete logarithm (DL) problem is defined as follows. Given the modulus $N$, the element $g$ and $h = g^x \bmod N$. What is $x$. ?

Assume that the principals have been running the protocol $\ell$ times. We define a view $V_i(\ell)$ of principal $P_i$ which specifies the information available to $P_i$ after $\ell$ successful executions of the protocol. It is easy to verify that

$$\begin{aligned} V_i(\ell) = \{ & f_i(z), S_i^{(2)} \rightarrow \text{(setup stage)} \\ & \alpha_1, \alpha_1^{S_i^{(1)}}, \ldots, \alpha_1^{S_n^{(1)}}, \alpha_1^{s} \rightarrow \text{(first run)} \\ & \vdots \\ & \alpha_\ell, \alpha_\ell^{S_i^{(1)}}, \ldots, \alpha_\ell^{S_n^{(1)}}, \alpha_\ell^{s} \rightarrow (\ell\text{th run}) \\ & + \text{public information}\} \end{aligned}$$

where $\alpha_1, \ldots, \alpha_\ell$ are random values generated by R. Note that the strings $\sigma_i$, generated for key confirmation purpose, are omitted from the view. The reason is that the assumption that the hash function is cryptographically strong is not enough to draw any conclusions about the overall security of the protocol. It is expected that hash function must not share any homomorphic property with exponentiation [12].

*Theorem 4.* Given the protocol without key confirmation. If the principals honestly follow the protocol and run it successfully $\ell$ times and the applied discrete logarithm instances are intractable, then the seed $s$ remains unknown to principals (and outsiders).

*Proof:* The proof is by contradiction. Suppose that the seed can be obtained from a view $V_i(\ell)$ using a polynomial-time probabilistic algorithm $A$ which takes the view as an input and returns the seed, or $A(V_i(\ell)) = s$. Consider an intractable instance of the DL problem defined by the pair $(h, g)$ such that $h = g^x$. For this instance, we create a view $V_{DL}(\ell)$. To do this, we need to design a secret sharing for $g^x$. We select at random integer $U_{n+1} \in_R Z_q^*$ and and $\delta_1 = g^{U_1}, \ldots, \delta_{n-1} = g^{U_{n-1}}$. Points $(1, g^{U_1})$, $(3, g^{U_2}) \ldots (2n-3, g^{U_{n-1}})$ together with $(0, g^x)$ and $(2, g^{U_{n+1}})$ uniquely determine the point $(2n-1, g^{U_n})$ using the Lagrange interpolation for exponents. The view generated for the DL instance has the following form:

$$\begin{aligned} V_{DL}(\ell) = \{ & f'(z), U_{n+1}, \\ & \alpha_1 = g^{r_1}, \alpha_1^{U_1}, \ldots, \alpha_1^{U_n}, \alpha_1^{x} = h^{r_1} \rightarrow \text{(first run)} \\ & \vdots \\ & \alpha_\ell = g^{r_\ell}, \alpha_\ell^{U_1}, \ldots, \alpha_\ell^{U_n}, \alpha_\ell^{x} = h^{r_\ell} \rightarrow (\ell\text{th run}) \\ & + \text{public information}\} \end{aligned}$$

where $f'(z)$, and $(\alpha_1, \ldots, \alpha_\ell)$ are random elements generated according to the protocol specification. We argue that views $V_i(\ell)$ and $V_{DL}(\ell)$ are statistically indistinguishable [13]. This is true as all elements are selected randomly and uniformly. Now we can input the view $V_{DL}(\ell)$ into the algorithm $A$. If the algorithm works for the view $V_i(\ell)$, it must also work for the view $V_{DL}(\ell)$ as both views are statistically indistinguishable. This means that $A$ returns $x$ and solve the intractable instance of the DL problem. This is requested contradiction which proves the claim. This completes the proof.

Consider the efficiency of the protocol. The first part in which the principals design their private secret sharing schemes is not computationally intensive. The reconstruction of the secret key $S$ and the key verification constitute the main computational overhead. To reconstruct the secret key, the principals have to first compute their public shares and later use the Lagrange interpolation to recover the polynomial $\alpha^{F(z)}$ and the secret $S = \alpha^{F(0)}$.

The communication overhead for the protocol consists of two components. The first one involves confidential delivery of the shares $s_{i,j}^{(2)}$ from any single principal to the others. This consumes $(n-1)$ confidential transmissions for every principal. The second component consists of broadcasting shares $\beta_1, = \alpha^{s_{i,j}^{(1)}}$. This takes $n$ broadcast transmissions for all principals. Table 1 summarises the communication and computation overhead for the protocol.

Our protocol compares favourably with other key agreement protocols. For example, the protocols by Burmester and Desmedt [6] are designed with a specific network configuration in mind. The most evident weakness of

**Table 1: Communication and computation requirements for the main protocol**

| | | Communication (message sent by each principal) | Computation (calculations done by each principal) |
|---|---|---|---|
| Registration | | 1 message sent to registry | 1 exponentiation |
| Setup | preparation | | $\approx 2n^2$ multiplications and additions for computations of shares |
| | distribution | $n$ messages sent to other principals via secure channel | $n$ exponentiations |
| | share broadcasts | $n$ broadcasts | $n$ exponentiations |
| Call for conference | key calculation | | $(n+1)$ exponentiations (Lagrange interpolation) |
| | key confirmation | 1 message broadcasts | Hashing of a single message and 1 exponentiation for authentication |

their protocols seems to be the lack of principal authentication. Just and Vaudenay [7] incorporated the authentication of principals into the Burmester–Desmedt protocols but the authentication can be achieved with the neighbouring principals only.

## 6 A (*t*, *n*) Multiparty key agreement

In general, conferences do not need the whole group to be active. Assume that out of $n$ members of the group, $t$ are allowed to call on the conference ($t < n$). The straightforward application of the previous protocol will not work. Note that after the principals collectively set up the $(t+1, 2n)$ secret sharing, any group of active principals larger than $t$ will compromise the confidentiality of the agreed key. One could solve this problem by requesting the principals who are about to broadcast their public shares to first contact the registry and ask it for permission. In this case, the registry keeps account of who has already contacted it and obtained permission. Some other possibilities are: to introduce an additional phase in which participants announces the intention of broadcasting a public shares by showing a random number. After the phase lapses, the principals order the numbers. Only the $t$ top ones are allowed to participate in the share broadcasting. Another possibility is to broadcast public shares in encrypted form. Later active principals order their encrypted shares and only the $t$ top principals broadcast the corresponding decryption keys. This can be an attractive option as encryption can be based on a fast block ciphet such as the DES algorithm.

The solution we present here is a modification of the main protocol which preserve the overall structure and more importantly, security evaluation obtained for the main protocol is also valid here. The registration phase is as before. In the initialisation phase, all the principals have to be active and each principal $P_i$ designs her own $(t+1, t+1)$ Shamir secret sharing defined by the polynomial $f_i(z)$ and computes $(t+1)$ shares $s_{i,j} = f_i(j)$ where $j = 1, \ldots, t+1$. She prepares $n$ auxillary shares $\varepsilon_{i,j}$ such that

$$\begin{bmatrix} \varepsilon_{i,1} \\ \vdots \\ \varepsilon_{i,n} \end{bmatrix} = X \times \begin{bmatrix} s_{i,1} \\ \vdots \\ s_{i,t} \end{bmatrix}$$

where $X$ is public $(n \times t)$ matrix whose any collection of $t$ rows constitutes a nonsingular matrix. She also distributes auxiliary shares to her fellow principals so $P_j$ obtains $\varepsilon_{i,j}$ for $j = 1, \ldots, n$; $j \neq i$ via secure channels and additionally $s_{i,t+1}$ is communicated securely to all other principals. She composes the auxiliary shares into a single auxiliary share

$$\varepsilon_i = \sum_{j=1}^{n} \varepsilon_{i,j}$$

and merges shares $s_{i,t+1}$ so

$$S_{t+1} = \sum_{i=1}^{n} s_{i,t+1}$$

this share is common for all principals.

The principals collectively set up a $(t+1, t+1)$ Shamir secret sharing with the underlying polynomial

$$F(z) = \sum_{i=1}^{n} f_i(z)$$

Denote shares $S_i = F(i)$. Any collection of $t$ or more different auxiliary shares allows a principal to recover the shares $(S_1, \ldots, S_t)$. Again $S_{t+1}$ is known to all principals.

To trigger the conference, there must be at least $t$ active principals. Without the loss of generality, assume that the active set $\mathcal{A} = \{P_1, \ldots, P_t\}$ and each $P_i \in \mathcal{A}$ performs the following steps:

(1) Contacts the registry and fetches necessary parameters including $\alpha = g^r$ (the registry selects $r$ at random). If the element $\alpha$ is not on display, $P_i$ asks R for one
(2) Prepares public share $\beta_i = \alpha^{\varepsilon_i}$ and broadcasts it
(3) Computes first shares $(\alpha^{S_1}, \ldots, \alpha^{S_t})$ using Lagrange interpolation (this step is identical to that used to recover the secret for Shamir scheme, note that the computations are done on exponents) and later calculates a common key $S = \alpha^{F(0)}$ using the complete set of shares $(\alpha^{S_t}, \ldots, \alpha^{S_{t+1}})$
(4) $P_i$ takes the secret $S$, her name $id_i$, and $\alpha$ and prepares a string $\sigma_i = H(S \| id_i \| \alpha)$ where $H$ is a cryptographically strong, collision resistant hash function with a public description. The triplet $(\sigma_i, id_1, \alpha)$ is broadcast (note that the broadcasting channel is assumed to provide authentication).
(5) $P_i$ collects $(\sigma_i, id_j, \alpha)$ from the other principals, checks their authenticity and verifies them using her own secret $S$.

If the checks hold, $P_i$ is ready for the conference. Otherwise, $P_i$ announces the error and aborts the protocol.

The protocol achieves the same security goals as the main protocol. If a run of the protocol is successful, all currently active principals know that there are at least $t$ of them in the group and the last steps allow them to identify them.

The modified protocol has the following remarkable properties.

(1) A principal who does not belong to $\mathcal{A}$, can always join the conference later by using the public shares and key confirmation strings

(2) A principal not in $\mathcal{A}$ can attend the conference passively, i.e. collect all public information which allows her to obtain the secret key. Later she can read all the information exchanged during the conference without others knowing that she is present

(3) It is possible to add a new principal to the conference (enrollment). It is sufficient for a newcomer to design her private secret sharing and distribute her shares to other members and other members give her their shares.

## 7 Conclusions

Principals involved in the protocols generate their own secret sharing schemes and use secure channels to distribute shares among themselves. In effect, they jointly create the group secret sharing. For a single run, principals use broadcasting to announce their public shares and use secret shares to compute the secret key. Note that the $(n+1, 2n)$ secret sharing used in the first protocol can be replaced by $(n+1, n+1)$ secret sharing. In this case, each principal has two shares and one of them is common for all principals.

Note that the second version of the protocol which can be run by any $t$ out on $n$ principals, uses a variant of secret sharing in which any $t$ active principals are able to reconstruct $t$ public shares (and recover the key using the common secret share).

Assume that a group of principals already collectively holds a secret via a $(t, n)$ secret sharing. Is it possible to design a protocol which enables the group to agree on a key? The answer seems to be in the affirmative. Consider a possible solution based on the concept of divisible shares [14]. The principals take their shares and 'divide' them into two parts: one will be used to produce public shares (using exponentiation) and the second is used to compute the common key. The effective threshold would in general, be, much higher than $t$. If the principals decide collectively to split their shares in two, then the effective threshold would be $2t-1$ as if $2t-1$ principals pool their share halves together, they will know $(t-1/2)$ public shares. This together with their secret halves allows them to recover the common key.

## 8 References


1 NEEDHAM, R.M., and SCHROEDER, M.D.: 'Using encryption for authentication in large networks of computers', *Commun. ACM*, 1978, **21**, (12), pp. 993–999

2 DIFFIE, W., and HELLMAN, M.E.: 'New directions in cryptography', *IEEE Trans.*, 1976, **IT-22**, pp. 644–654

3 DIFFIE, W., VAN OORSCHOT, P., and WIENER, M.: 'Authentication and authenticated key exchanges', *Des. Codes, Cryptogr.*, 1992, **2**, pp. 107–125

4 INGEMARSSON, I., TANG, D., and WONG, C.: 'A conference key distribution system', *IEEE Trans.*, 1982, **IT-28**, pp. 714–720

5 FIAT, A., and NAOR, M.: 'Broadcast encryption', *in* STINSON, D.R. (Ed.): 'Advances in cryptology–CRYPTO'93, Lecture Notes in Computer Science No. 773 (Springer, 1994), pp. 480–491

6 BURMESTER, M., and DESMEDT, Y.: 'A secure and efficient conference key distribution system', *in* DE SANTIS, A. (Ed.): 'Advances in cryptology–EUROCRYPT'94, Lecture Notes in Computer Science No. 950 (Springer, 1995), pp. 275–286

7 JUST, M., and VAUDENAY, S.: 'Authenticated multi-party key agreement', *in* KIM, K. and MATSUMOTO, T. (Eds.): 'Advances in cryptology –ASIACRYPT'96', Lecture Notes in Computer Science No. 1163 (Springer, 1996), pp. 36–49

8 BLAKLEY, G.R.: 'Safeguarding cryptographic keys'. Proceedings of the AFIPS 1979 national computer conference, (AFIPS, 1979), pp. 313–317

9 SHAMIR, A.: 'How to share a secret', *Commun. ACM*, 1979, **22**, pp. 612–613

10 STINSON, D.R.: 'An explication of secret sharing schemes', *Des. Codes Cryptogr.*, 1992, **2**, pp. 357–390

11 HERZBERG, A., JARECKI, S., KRAWCZYK, H., and YUNG, M.: 'Proactive secret sharing or: how to cope with perpetual leakage', *in* COPPERSMITH, D. (Ed.): 'Advances in Cryptology –CRYPTO'95, Lecture Notes in Computer Science No. 963' (Springer, 1995), pp. 339–352

12 MENEZES, A., VAN OORSCHOT, P., and VANSTONE, S.: 'Handbook of applied cryptography' (CRC Press, Boca Raton, Florida, 1997)

13 STINSON, D.R.: 'Cryptography: theory and practice' (CRC Press, Boca Raton, Florida, 1995)

14 MARTIN, K., PIEPRZYK, J., SAFAVI-NAINI, R., and WANG, H.: 'Changing thresholds in the absence of secure channel'. Proceedings of the fourth Australasian conference on *information Security and Privacy* (ACISP99), 1999, Springer-Verlag, **1587**, pp. 177–191