Some Applications of Threshold Signature Schemes to Distributed Protocols

Vanesa Daza, Javier Herranz and Germán Sáez

Dept. Matemàtica Aplicada IV, Universitat Politècnica de Catalunya
C. Jordi Girona, 1-3, Mòdul C3, Campus Nord, 08034 Barcelona, Spain
e-mail: {vdaza,jherranz,german}@mat.upc.es

Abstract

In a threshold signature scheme, a group of players share a secret information in such a way that only those subsets with a minimum number of players can compute a valid signature. We propose methods to construct some useful and computationally secure distributed protocols from threshold signature schemes satisfying some suitable properties.

Namely, we prove that any threshold signature scheme which is non-interactive can be used to construct a metering scheme. We also design a distributed key distribution scheme from any deterministic threshold signature scheme. The security of these news schemes is reduced to the security of the corresponding threshold signature schemes. Furthermore, the constructed protocols reach some desirable properties.

1 Introduction

In a threshold signature scheme, some participants share a secret information that enables some subsets of them (those with a certain number of participants) to compute valid signatures. Security and reliability increase in collective digital signatures because the tasks performed by a single party in an individual signature scheme are now distributed among a set of players. These schemes must be secure against the action of an adversary who corrupts some dishonest players. Roughly speaking, a scheme is said to be *unforgeable* if any subset of dishonest players can not obtain any information that allows them to compute a signature. A scheme is said to be *robust* if it can detect corrupted participants, and they can not avoid honest players to generate a valid signature. The result of the process of a threshold signature scheme is a standard signature.

Metering schemes were introduced in [15] in order to measure the number of interactions between servers and clients (for example, the access of a client to a web server). Each client that visits a server must send to him some secret information. When a server has been visited by a certain number (the threshold) of clients in a period of time, he can compute a valid proof of these visits from the secret information that he has received. This proof is sent to a trusted third party, who will take it into account in order to decide on advertisement fees for web servers, for example. Some proposals of metering schemes have been done in both the information-theoretic scenario ([14, 3]) and the computationally secure one ([15, 18]). In this work, we show how any threshold signature scheme which is secure and non-interactive can be used to construct a computationally secure metering scheme. In this new scheme, anyone (for instance, the person who must pay web advertisment fees) can publicly verify the proofs computed by servers, not only a trusted third party.

Another important primitive in cryptographic protocols is the distribution and managing of secret keys. Needham and Schroeder [17] introduced a single server responsible of distributing keys. Later, some improvements of the model have been done. Naor, Pinkas and Reingold [16] presented a model where the task of a single server is distributed among a set of servers. These are the so-called *distributed key distribution schemes*. Both information theoretic ([1, 2]) and computational models have been widely studied in literature. We focus on computationally secure distributed key distribution schemes. Previously, in [16] such a scheme was proposed, based on the Decisional Diffie-Hellman Assumption. Recently, in [8] a new model, that reduces the computational effort of the users by means of increasing servers' computations, has been proposed. In this paper, we construct general distributed and computationally secure key distribution schemes following the original model in [16], from any secure and deterministic threshold signature scheme.

In the extended version of this paper, we will construct explicit metering and distributed key distribution schemes using these methods from specific threshold signature schemes.

Organization of the paper. In Section 2, we explain how a threshold signature scheme works and the requirements that one such scheme must satisfy in order to be considered secure. In Section 3, we introduce a method to construct computationally secure metering schemes from any non-interactive threshold signature scheme. In Section 4, we show how any deterministic threshold signature scheme can be used to design distributed and computationally secure key distribution schemes. We sum up the results of the paper and propose some related future research in Section 5.

2 Threshold Signature Schemes

2.1 Secret Sharing Schemes

Secret sharing schemes play an important role in distributed cryptography. In these schemes, a secret value is shared among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n players in such a way that only qualified subsets of $\mathcal{P}$ (those in the *access structure*, denoted by Γ) can reconstruct the secret from their shares. This family $\Gamma \subset 2^{\mathcal{P}}$ of authorized subsets must be *monotone*, that is, if $A_1 \in \Gamma$ and $A_1 \subset A_2 \subset \mathcal{P}$, then $A_2 \in \Gamma$.

Shamir's secret sharing scheme was introduced in [19] and it realizes threshold access structures $\Gamma = \{A \subset \mathcal{P} \ : \ |A| \geq t\}$, for some threshold t. To share a secret k in a finite field K, the dealer chooses a random polynomial $f(z) = k + a_1 z + \cdots + a_{t-1} z^{t-1} \ \in K[z]$ of degree $t-1$. The share of participant P_i is $s_i = f(i)$, for $i = 1, \ldots, n$.

Let $A = \{P_{i_1}, \ldots, P_{i_t}\}$ be a subset of t participants. They have t different values of the polynomial $f(z)$, of degree $t-1$, so they can obtain the value $k = f(0)$. We have $k = f(0) = \sum_{P_i \in A} \lambda_{0,i}^A f(i)$, where $\lambda_{0,i}^A$ are the Lagrange interpolation coefficients.

2.2 Definitions and Previous Work

A (t,n)-*threshold signature scheme* Σ differs from a regular (individual) one because the secret information (usually, the secret key) is shared among a set of n players. This sharing is usually performed by using Shamir's secret sharing scheme or some variant of it. No coalition of less than t players can compute a valid signature. A threshold signature scheme Σ consists of three protocols (see for example [11]):

Thresh-Key-Gen: this protocol can be executed jointly by the n players themselves, or by a trusted and external authority. The input is a security parameter. The public outputs are pk (the public key of the scheme) and some verification key vk, whereas each player P_i has as his private output a share sk_i of the secret key related to pk. This protocol must be probabilistic and polynomial-time.

Thresh-Sig: public inputs of this protocol are the message m to be signed, the public key pk and the verification key vk. Each player P_i uses his private information to compute and broadcast his partial signature $\sigma_i(m)$. In some schemes, he must also broadcast a proof of correctness of his partial signature. The correctness of the partial signatures can be verified using the verification key vk. Finally, a combiner algorithm takes t valid partial signatures and produces a valid standard signature $\sigma(m)$. This protocol *Thresh-Sig* must be polynomial-time.

Verif: this protocol is executed by the recipient of the signature, and is the same as in a regular signature scheme. The inputs are the public key pk, the message m and the signature $\sigma(m)$. The output will be "yes" if the result of the verification is correct, or "no" if $\sigma(m)$ is not a valid signature on message m. This protocol must be deterministic and polynomial-time.

We say that a threshold signature scheme is *non-interactive* if, in the execution of *Thresh-Sig*, each player can compute his partial signature independently of the rest of players. Otherwise, the scheme is *interactive*. Analogously to a regular signature scheme, a threshold signature scheme can be *deterministic* (if each message has a unique valid signature) or *probabilistic* (multiple valid signatures).

Some proposals of threshold signature schemes made until now can be found in [11, 21]; they are probabilistic and interactive, based on the DSS and Schnorr schemes. On the other hand, some threshold proposals of the RSA signature scheme, which are deterministic and non-interactive, can be found in [20, 7, 9]. Finally, threshold versions of two RSA-based signature schemes [6, 13] are proposed in [5]. They are both interactive; the threshold version of Cramer-Shoup scheme [6] is probabilistic, whereas the threshold version of Gennaro-Halevi-Rabin scheme [13] is deterministic. The key generation protocol of these schemes can be performed by a trusted dealer or jointly by the own participants, by using some of the protocols in [12, 4, 10].

2.3 Security

A (t, n)-threshold signature scheme must be secure even in the presence of an adversary who corrupts and controls the behavior of $t - 1$ players. We say that such a scheme is secure if it is *robust* and *existentially unforgeable under chosen-message attacks*.

Robustness means that the scheme provides mechanisms to detect corrupted players that broadcast incorrect information. Furthermore, the protocol must produce always a valid signature from the partial signatures of the honest players. Usually, a necessary condition for robustness in a (t, n)-threshold signature scheme is $n \geq 2t - 1$.

Unforgeability under chosen-message attacks means that an adversary who corrupts $t - 1$ players (that is, he knows their private information and controls their behavior in the protocols) has negligible probability of obtaining in polynomial time a valid signature on a message m. This happens even if the adversary knows all the information broadcast by all players (corrupted or not) during the execution of the protocol *Thresh-Sig* on input messages $m_1, \ldots, m_k$ which the adversary adaptively chooses, such that $m \neq m_j$, $\forall j = 1, \ldots, k$.

Usually, unforgeability of a threshold signature scheme is proved by reducing it to the unforgeability of the regular signature scheme in which the threshold one is based. The reasoning is that a successful forger against the threshold scheme could be used as a sub-routine by a successful forger against the regular scheme. Thus, if the regular scheme is assumed to be unforgeable, then the threshold scheme must be unforgeable, too. The security of some signature schemes (regular or threshold) is proved in the standard cryptographic model, whereas other schemes are proved secure in the random oracle model.

3 New Computational Metering Schemes

Metering schemes are designed to measure the interaction between servers and clients during a certain number of time frames. One application of metering schemes is the measure of number of web accesses to servers in the Internet, in order to decide on advertisement fees for web servers.

In a metering scheme, an *audit agency* distributes some secret information c_i to each client C_i in $\mathcal{C} = \{C_1, \ldots, C_n\}$. When a client C_i visits a server S_j in $\mathcal{S} = \{S_1, \ldots, S_m\}$ during a time frame $\lambda = 1, \ldots, \tau$, he gives some piece of information c_{ij}^{λ} to him. Once a server S_j has been visited by a subset of t clients, he can compute the proof p_j^{λ}. With this proof, he can demonstrate to the audit agency that at least t clients have visited him in time frame λ.

Some proposals of unconditionally secure metering schemes have been given [15, 14, 3]. These metering schemes are secure against an infinitely powerful adversary. On the other hand, only few proposals have been done in a computational setting. In [15] Naor and Pinkas propose a computationally secure scheme under the computational Diffie-Hellman assumption using bivariable polynomials. Ogata and Kurosawa propose in [18] a computationally secure scheme based on the same assumption, using polynomials in three variables.

Now we present a new method to construct a computationally secure metering scheme from any threshold signature scheme that is non-interactive (Section 2.2). This happens in the threshold signature schemes proposed in [20, 7, 9].

3.1 Metering Schemes from Threshold Signature Schemes

Let Σ be a non-interactive (t, n)-*threshold signature scheme*. A metering scheme is usually divided in three different phases: an *initialization phase*, which involves the audit agency and clients; a *regular operation*, in which clients visit servers; and the *proof computation phase*, in which servers obtain the proof that they have been visited by at least t clients. We denote as Σ-*metering scheme* the metering scheme constructed as follows.

Initialization phase: in this phase, the audit agency executes the protocol *Thresh-Key-Gen* of Σ to generate public outputs pk (the public key of the signature scheme), verification key vk, and private outputs sk_i. Then the audit agency sends secretly the information $c_i = sk_i$ to each client C_i.

Regular operation: the idea behind this protocol is that clients produce partial signatures on message $h(S_j, \lambda)$ when they visit server S_j in a time frame λ. Here h must be an injective function whose outputs are valid messages. Since Σ is non-interactive, the client C_i does not need the other clients to compute this partial signature $c_{ij}^{\lambda} = \sigma_i(h(S_j, \lambda))$ and send it to server S_j, together with some proof of correctness. Server S_j checks if the partial signature is correct using the public verification key (here Σ is assumed to be a robust threshold signature scheme); if not, he denies the access to C_i.

Proof computation phase: server S_j uses the combiner algorithm of the protocol *Thresh-Sig* of Σ to produce a valid standard signature $p_j^{\lambda} = \sigma(h(S_j, \lambda))$

from t valid partial signatures. The audit agency (or anyone) uses the public key pk to verify the validity of the proof p_j^λ using the protocol *Verif* of Σ.

Observe that in our Σ-metering scheme there is not a prefixed number of time frames for which the scheme can be used. This is true if the map h is an injection. On the other hand, the amount of the secret information distributed among clients and servers, and the complexity of the computations are independent from the number of time frames. These properties (satisfied also by other computational metering schemes as the one in [18]) are an improvement with respect to the unconditionally secure proposals ([15, 14, 3]).

In our proposal, the audit agency is not needed in the verification phase, because the validity of the proofs (signatures) computed by a server can be verified publicly by anyone (for example, the advertisers who must pay the corresponding fees) from the public key of the scheme. So the presence of the audit agency is only necessary for the generation of the initial secret and public informations of the scheme. This can be seen as an improvement with respect to the previous proposals of metering schemes. For example, in the computational scheme proposed in [18], the audit agency is the only one who can check the validity of the proofs.

3.2 Security of Σ-metering schemes

First of all we will define the security of a computational metering scheme, and then we will proof the security of Σ-metering scheme, according to this definition. The idea is that a server will not be able to compute a valid proof for a time frame if he has been visited by less than t clients during this period of time, even if he receives information from the other servers.

Definition 1. *We say that a metering scheme is* computationally secure *if any computationally bounded (polynomial time) adversary who corrupts at most $t-1$ clients and all servers, is not able to obtain a valid proof $p_{j_0}^T$ corresponding to a server S_{j_0} who has not been visited during time frame T by any client distinct from the corrupted clients. This must happen even if all clients C_i have visited all servers S_j in all previous time frames $\lambda \leq T$, excluding obviously the case $(j, \lambda) = (j_0, T)$.*

Theorem 2. *If Σ is a non-interactive threshold signature scheme existentially unforgeable under chosen-message attacks, then Σ-metering scheme is computationally secure.*

Proof. We prove this theorem by assuming that the Σ-metering scheme is not computationally secure and then proving that Σ is existentially forgeable against a chosen-message attack.

Let us assume, therefore, that the Σ-metering scheme is not computationally secure. That is, there exist $t-1$ clients (we suppose without loss of generality that they are $C_1, \ldots, C_{t-1}$), a server S_{j_0}, a time frame T and a forger algorithm $\mathcal{F}_1$ that takes as input the secret information of clients $C_1, \ldots, C_{t-1}$

and all the information that all clients $C_i \in \mathcal{C}$ give when they visit all servers $S_j \in \mathcal{S}$ in all previous time frames $\lambda \leq T$, excluding the case $(j, \lambda) = (j_0, T)$. With non-negligible probability and in polynomial time, $\mathcal{F}_1$ outputs a valid proof $p_{j_0}^T = \sigma(h(S_{j_0}, T)))$. Next, we construct a forger algorithm $\mathcal{F}_2$ that can be used by an adversary who corrupts the clients $C_1, \ldots, C_{t-1}$ to forge the threshold signature scheme Σ with a chosen-message attack.

When the initialization phase of the Σ-metering scheme is executed, the adversary knows the real shares $c_i = sk_i$, for $i = 1, \ldots, t-1$, of the secret key of the threshold signature scheme Σ. In order to perform a chosen-message attack, the adversary chooses the messages $h(S_j, \lambda)$, for $j = 1, \ldots, m$, $\lambda \leq T$, and $(j, \lambda) \neq (j_0, T)$, to be signed with the protocol *Thresh-Sig* of Σ, obtaining all the public information (signatures and partial signatures) broadcast during these executions. This information is exactly the information that all servers S_j would obtain from Σ-metering scheme if they were visited by all clients in time frames $\lambda \leq T$, provided $(j, \lambda) \neq (j_0, T)$.

The adversary runs the forger algorithm $\mathcal{F}_1$ with inputs the information obtained from these executions of the protocol *Thresh-Sig* of Σ and the secret information of clients $C_1, \ldots, C_{t-1}$. Since $\mathcal{F}_1$ outputs $p_{j_0}^T = \sigma(h(S_{j_0}, T)))$, $\mathcal{F}_2$ obtains, in polynomial time and with non-negligible probability, a valid signature for a message $h(S_{j_0}, T)$ different from the messages $h(S_j, \lambda)$ that were signed in the chosen executions of the threshold signature protocol. This completes the proof. $\square$

Note that if the threshold signature scheme Σ is proved to be secure in the random oracle model, then Σ-metering scheme is computationally secure in the same model.

Ogata and Kurosawa propose in [18] a computationally secure metering scheme basing its security on the computational Diffie-Hellman assumption. They prove that an adversary who corrupts $t-1$ clients and d servers can not compute the proof p_j^λ, where d is a security parameter. Since a Σ-metering scheme is secure even if the adversary corrupts all the servers, we can in some way assure that it is more secure than the scheme in [18].

4 New Computational and Distributed Key Distribution Schemes

First of all, we briefly describe how the threshold and computationally secure key distribution scheme proposed in [16] works. They consider a set of servers $\mathcal{S} = \{S_1, \ldots, S_n\}$ and a group of users $\mathcal{U} = \{U_1, \ldots, U_m\}$ (they also refer to them as clients). Let $\mathcal{C} \subset 2^{\mathcal{U}}$ be a family of sets of users, the *conferences*, who want to communicate securely among them. In the initialization phase, each server S_i receives a share α_i of some random secret α, shared (jointly or using a trusted authority) among the servers by means of Shamir secret sharing scheme. When a user U in a conference $C \in \mathcal{C}$ needs the key of this conference (each conference C is related to a public value h_C), he contacts with at least $2t-1$ servers and asks them for the key of the conference C. Afterwards, every contacted server verifies that U belongs to C and, if so, computes the value $h_C^{\alpha_i}$ and sends it to him through their private channel.

After receiving the information from the servers, the user detects corrupted servers. As these dishonest servers are assumed to be less than t, the user is able to compute the conference key κ_C by using the correct values received from t or more honest servers. Without loss of generality, we assume these honest servers are $S_1, \ldots, S_t$. The computation of κ_C is performed as follows: $\kappa_C = h_C^{\alpha} = \prod_{i=1}^{t} (h_C^{\alpha_i})^{\lambda_i}$, where λ_i are the Lagrange interpolation coefficients.

Recently, a new model has been proposed [8] in order to reduce the amount of computations the user must perform to obtain the key. The usefulness of this model relies on those situations where servers have main part of the computational power.

Next, we present a new method to construct computationally secure and distributed key distribution schemes, following the model introduced in [16], from a threshold signature scheme. This construction will be possible as long as the considered signature scheme is deterministic (Section 2.2), as it happens in the proposals of [20, 7, 9] and the threshold version of the Gennaro-Halevi-Rabin scheme [13] proposed in [5].

4.1 Distributed Key Distribution Schemes from Threshold Signature Schemes

Let Σ be a deterministic (t,n)-*threshold signature scheme*. As it has been done in some other recent works [2], we divide a distributed key distribution scheme in three different phases: an *initialization phase*, which involves only the servers; a *key-request phase*, in which users ask for keys to servers; and a *key-computation phase*, in which users retrieve keys from the messages received from the servers contacted during the key request phase. We refer to the distributed key distribution scheme constructed as follows as Σ-*DKDS*.

Initialization phase: in this phase, the protocol *Thresh-Key-Gen* of the signature scheme Σ is performed, taking as participants the set of servers. The protocol can be executed jointly by the n servers themselves, or by a trusted and external authority. At the end of this phase, each server S_i has a share sk_i of the secret key.

Key request phase: a user U_j in a conference $C \in \mathcal{C}$ contacts with a group of at least $2t-1$ servers requiring a key for the conference C. We denote such a key κ_C. After checking for membership of U_j in C, every contacted server S_i performs first part of the *Thresh-Sig* protocol of Σ. In this way, he uses his private information to compute the partial signature $\sigma_i(h(C))$ and sends it privately to the user. Here h is a public cryptographic hash function. The server must also send to the user a proof of correctness of his partial signature.

Key computation phase: once having received the answers from the contacted servers, user U_j checks the proof of correctness of partial signatures he has received in the previous phase (here we are assuming that the threshold scheme Σ is robust). By means of using the combiner algorithm of Σ, U_j takes t valid partial signatures (where by valid we refer to those signatures that pass the checking of the proof of correctness) and produce a standard signature $\sigma(h(C))$ on the message $h(C)$. The key for the conference C is $\kappa_C = \sigma(h(C))$. The user can verify the correctness of this key by using the protocol $Verif$ of Σ, checking therefore if κ_C is a valid signature of message $h(C)$.

Note that all users in the conference C obtain the same key κ_C, since the threshold signature scheme Σ is deterministic.

4.2 Security of Σ-DKDS

We base the security of the scheme Σ-DKDS on the security of the yielding threshold signature scheme Σ under chosen-message attacks.

Before giving a formal definition of the security of a distributed and computational key distribution scheme, we introduce some notation that will be useful throughout the rest of the section. Given a conference $C \in \mathcal{C}$, let $W_C = \mathcal{U} - C$ be the set of users not in the conference C. We also define the sets of pairs of users and conferences $P_C = \{(U, C) \mid U \in C\}$ and $N_C = \{(U, C') \mid U \in W_C \text{ and } U \in C'\}$.

Definition 3. *We say that a distributed key distribution scheme is* computationally secure *if for every conference $C \in \mathcal{C}$, any computationally bounded (polynomial time) adversary who corrupts at most $t-1$ servers and all users in W_C, is not able to obtain κ_C, even if the protocol is previously performed for all pairs in P_C and N_C.*

Note that, in the model that we consider, execution of the protocol for the pairs in P_C does not give any new information to the adversary, because he does not control the users of conference C. Therefore, the information that the honest servers send secretly to these honest users remains unknown to the adversary.

Theorem 4. *If Σ is a deterministic threshold signature scheme existentially unforgeable under chosen-message attacks, then Σ-DKDS is computationally secure.*

Proof. In order to prove this theorem, we will prove that if Σ-DKDS is not computationally secure, then the threshold signature scheme is existentially forgeable against a chosen-message attack.

In effect, if Σ-DKDS is not computationally secure, then there exist a conference $C \in \mathcal{C}$ and a forger algorithm $\mathcal{F}_1$ that takes as input the secret information of $t-1$ corrupted servers (without loss of generality, $S_1, \ldots, S_{t-1}$) and all the new information that users in W_C and servers $S_1, \ldots, S_{t-1}$ can obtain from the executions of Σ-DKDS for pairs in P_C and N_C (as we have said before, in this model only executions for pairs in N_C could give some useful information to the adversary). The output of $\mathcal{F}_1$ is $\kappa_C = \sigma(h(C))$). Next, we show how an adversary who corrupts $t-1$ servers can construct a forger algorithm $\mathcal{F}_2$ that forges the scheme Σ with a chosen-message attack.

Once the initialization phase of Σ-DKDS is performed, the adversary obtains the real shares of the secret key of the threshold signature scheme Σ corresponding to servers $S_1, \ldots, S_{t-1}$. By definition of chosen-message attack, the adversary can choose some messages to be signed with the protocol *Thresh-Sig* of Σ, obtaining all the public information (signatures and partial signatures) broadcast during these executions. In this case, the adversary chooses the messages $h(C')$ for all conferences $C' \neq C$ such that there exists some $U \in W_C$ with $(U, C') \in N_C$. Then the adversary runs the forger algorithm $\mathcal{F}_1$ with input the secret information of servers $S_1, \ldots, S_{t-1}$ and the information obtained from these executions of the protocol *Thresh-Sig* of Σ (which is exactly the information that users in W_C would obtain if Σ-DKDS was executed for pairs in N_C). $\mathcal{F}_1$ obtains $\kappa_C = \sigma(h(C))$ with non-negligible probability, so $\mathcal{F}_2$ obtains, also with non-negligible probability, a signature for a message $h(C)$ different from the messages $h(C')$ that were signed in the chosen executions of the threshold signature protocol. $\square$

Again, if the security of the scheme Σ can be proved only in the random oracle model, then Σ-DKDS is computationally secure also in this model.

5 Conclusions and Future Work

In this paper we have proposed two methods to construct computationally secure metering schemes and distributed key distribution schemes, from any secure threshold signature scheme. In the case of metering schemes, the considered threshold signature scheme must be non-interactive, whereas in the case of distributed key distribution schemes, the threshold signature scheme must be deterministic. The two constructions provide schemes which are as secure as the corresponding threshold signatures schemes.

To the best of our knowledge, the only threshold signature schemes proposed until now that satisfy the properties of determinism and non-interaction are the schemes in [20, 7, 9]. So we can construct from them a metering scheme as well as a distributed key distribution scheme. On the other hand, the threshold version in [5] of the signature scheme in [13] is deterministic but interactive, so we can use it only to construct a distributed key distribution scheme. In general, the use of non-interactive threshold signature schemes in our construction of distributed key distribution schemes would be preferable, because in this case the servers must not communicate among them in the computation of the conference keys.

All these proposed schemes run only with threshold structures, so an interesting open problem is to find distributed signature schemes running with more general access structures, and satisfying some of the properties (determinism or non-interaction) that are required in this work.

In our new metering schemes, anyone can publicly verify the validity of the proofs, not only the audit agency. However, this figure is still necessary in the initialization phase, because a jointly generation of the secret information held by the clients would be interactive and expensive. This does not fit in with a real situation where clients are users of the Internet.

With respect to distributed key distribution schemes, a different model from the one in [16] has been recently proposed in [8], reducing the computational effort of the users in the calculation of the conference key. Another open problem is to discuss if we can find methods to construct, from threshold signature schemes, distributed key distribution schemes computationally secure in this different model.

References

[1] C. Blundo and P. D'Arco. An Information Theoretic Model for Distributed Key Distribution. *Proceedings of IEEE ISIT'2000*, p. 267 (2000).

[2] C. Blundo, P. D'Arco, V. Daza and C. Padró. Bounds and constructions for unconditionally secure distributed key distribution schemes with general access structures. *Proceedings of ISC'2001*, LNCS 2200, pp. 1-17 (2001).

[3] C. Blundo, S. Martín, B. Masucci and C. Padró. A Linear Algebraic Approach to Metering Schemes. Preprint available at `http://eprint.iacr.org/2001/087/`

[4] D. Boneh and M. Franklin. Efficient Generation of Shared RSA Keys. *Advances in Cryptology-Crypto'97*, LNCS **1294**, pp. 425-439 (1997).

[5] D. Catalano, R. Gennaro and S. Halevi. Computing inverses over a shared secret modulus. *Advances in Cryptology-Eurocrypt'00*, LNCS **1807**, pp. 190-206 (2000).

[6] R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In *ACM Conference on Computer and Communications Security*, pp. 46-51 (1999).

[7] I. Damgård and M. Koprowski. Practical threshold RSA signatures without a trusted dealer. *Advances in Cryptology-Eurocrypt'01*, LNCS **2045**, pp. 152-165 (2001).

[8] V. Daza, J. Herranz, C. Padró and G. Sáez. A Distributed and Computationally Secure Key Distribution Scheme. Preprint available at `http://eprint.iacr.org/2002/069/`

[9] P.A. Fouque and J. Stern. Fully distributed threshold RSA under standard assumptions. *Proceedings of Asiacrypt'2001*, LNCS **2248**, pp. 310-330 (2001).

[10] Y. Frankel, P. MacKenzie and M. Yung. Robust efficient distributed RSA-key generation. *Proceedings of STOC'98*, pp. 663-672 (1998).

[11] R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin. Robust Threshold DSS Signatures. *Advances in Cryptology-Eurocrypt'96*, LNCS **1070**, pp. 354-371 (1996).

[12] R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. *Advances in Cryptology-Eurocrypt'99*, LNCS **1592**, pp. 295-310 (1999).

[13] R. Gennaro, S. Halevi and T. Rabin. Secure hash-and-sign signature without the random oracle. *Advances in Cryptology-Eurocrypt'99*, LNCS **1592**, pp. 295-310 (1999).

[14] B. Masucci and D.R. Stinson. Metering Schemes for General Access Structures. *ESORICS'2000*, LNCS **1895**, pp. 72-87 (2000).

[15] M. Naor and B. Pinkas. Secure and Efficient Metering. *Advances in Cryptology-Eurocrypt'98*, LNCS **1403**, pp. 576-590 (1998).

[16] M. Naor, B. Pinkas and O. Reingold. Distributed pseudo-random functions and KDCs. *Advances in Cryptology-Eurocrypt'99*, LNCS **1592**, pp. 327-346 (1999).

[17] R. M. Needham and M. D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, vol. **21**, pp. 993-999 (1978).

[18] W. Ogata and K. Kurosawa. Provably Secure Metering Scheme. *Proceedings of Asiacrypt'2000*, LNCS **1976**, pp. 388-398 (2000).

[19] A. Shamir. How to share a secret. *Communications of the ACM*, vol. **22**, pp. 612-613 (1979).

[20] V. Shoup. Practical Threshold Signatures. *Advances in Cryptology-Eurocrypt'00*, LNCS **1807**, pp. 207-220 (2000).

[21] D.R. Stinson and R. Strobl. Provably Secure Distributed Schnorr Signatures and a (t, n) Threshold Scheme for Implicit Certificates. *Proceedings of ACISP'2001*, LNCS **2119**, pp. 417-434 (2001).