INFORMATICA, 2009, Vol. 20, No. 4, 591–612
© 2009 *Institute of Mathematics and Informatics, Vilnius*

Adaptively Secure Threshold Signature Scheme in the Standard Model

Zecheng WANG[1,2], Haifeng QIAN[1], Zhibin LI[1]
[1]*Department of Computer Sci. & Tech., East China Normal University, 200062 Shanghai, China*
[2]*Department of Computer Sci. & Tech., Anhui University of Finance and Economics 233041 Bengbu, China*
e-mail: w52051201006@hotmail.com; {hfqian, lizb}@cs.ecnu.edu.cn

Received: June 2007; accepted: June 2008

Abstract. We propose a distributed key generation protocol for pairing-based cryptosystems which is adaptively secure in the erasure-free and secure channel model, and at the same time completely avoids the use of interactive zero-knowledge proofs. Utilizing it as the threshold key generation protocol, we present a secure (t, n) threshold signature scheme based on the Waters' signature scheme. We prove that our scheme is unforgeable and robust against any adaptive adversary who can choose players for corruption at any time during the run of the protocols and make adaptive chosen-message attacks. And the security proof of ours is in the standard model (without random oracles). In addition our scheme achieves optimal resilience, that is, the adversary can corrupt any $t < n/2$ players.

Keywords: threshold signature, distributed key generation, computational Diffie–Hellman problem, adaptively secure, provable security.

1. Introduction

Since threshold cryptography was introduced by the works of Boyd (1986), Croft and Harris (1989), Desmedt (1988), and Desmedt and Frankel (1990), it has received considerable attention. Many threshold cryptosystems, including many kinds of threshold signature schemes, have been proposed (Chien *et al.*, 2003; Desmedt, 1994; Long *et al.*, 2006; Qian *et al.*, 2005; Shoup, 2000; Tsai *et al.*, 2003). And a lot of techniques were developed.

As part of distributed cryptography, there are also two adversary models in threshold cryptography, one is static adversary and the other is adaptive adversary (Canetti *et al.*, 1999). In both cases the adversary is allowed to corrupt any subset of players up to some threshold. However, in the case of an adaptive adversary, the adversary can choose which players to corrupt at any time and based on any information he sees during the run of the protocol. In contrast, in the case of a static adversary, the adversary fixes the players that will be corrupted before the protocol starts. It is known that the adaptive adversary is strictly stronger than the static one (Canetti *et al.*, 1996, 2000; Cramer *et al.*, 1999). Since the adaptive adversary model appears to better capture real threats, designing and proving threshold signature schemes secure in the adaptive adversary model has been focused in recent years.

Some techniques have been proposed to achieve adaptively secure for threshold signature schemes. Canetti *et al.* (1999) and Frankel *et al.* (1999a, 1999b) achieved adaptive security respectively by developing and utilizing many protocol designing and proving techniques, such as using additive sharing instead of polynomial sharing, Pedersen's commitment and zero-knowledge proofs, erasing private values, rewinding the adversary, single-inconsistent-player *etc.* in the secure channel model. Jarecki and Lysyanskaya (2000) and Lysyanskaya and Peikert *et al.* (2000, 2001) improved the schemes to work in the erasure-free model and to remain secure under concurrent composition by developing a novel construction tool of a committed zero-knowledge proof and a new analytical tool of single persistently inconsistent player. Furthermore, they implemented the secure channels in the adaptive erasure-free model by devising a receiver-non-committing encryption scheme. Thus, their threshold cryptosystems could be implemented in the non-secure channel model. Though their schemes achieved more security and functionality, their schemes still heavily depended on zero-knowledge proofs. Abe *et al.* (2004) implemented two adaptively secure Feldman VSS (Feldman, 1987) schemes, one is in the non-secure channel model and the other is in the secure channel model. Based on the one in non-secure channel, they proposed adaptively secure distributed discrete-log key generation protocol in the erasure-free model, which was also proved secure in the single-inconsistent-player UC model[1]. They also proposed a fully UC threshold Schnorr signature scheme, a fully UC threshold DSS signature scheme and other adaptively secure protocols. And they avoided the use of interactive zero-knowledge proofs.

On the other hand, many efficient digital signature schemes (Bellare and Rogaway, 1996; Boneh *et al.*, 2001) and their threshold versions (Shoup, 2000; Boldyreva, 2003) are proved secure in the random oracle model (Bellare and Rogaway, 1993). However, the result from Canetti *et al.* (1998) shows that there exists an encryption scheme which is secure in the random oracle model (RO model), but is not secure in the complexity-theoretic model (named CT model or standard model), no matter the instantiation of the RO. This leads to focus on constructing secure cryptosystems proved without random oracles, e.g., in the standard model.

Currently, most practical signature schemes proved secure without random oracles are based on the Strong RSA assumption (Cramer and Shoup, 2000; Gennaro *et al.*, 1999a) or the Strong Diffie–Hellman assumption (Boneh and Boyen, 2004). Recently, Waters proposed an efficient digital signature scheme (Waters, 2005) based on the Computational Diffie–Hellman assumption which can be proved secure without random oracles. This is the first signature scheme based on the more standard computational complexity assumption.

Based on the short digital signature scheme proposed in Boneh and Boyen (2004), Wang *et al.* (2005) proposed a threshold signature scheme proved secure without random oracles. But their scheme is based on the Strong Diffie–Hellman assumption, which

[1]Universally Composable security model is proposed by Canetti (2001) which defines stronger security notion for cryptographic protocols.

is a stronger assumption than the Computational Diffie–Hellman assumption. Based on Waters' provably signature scheme, Xu proposed a provably secure threshold signature scheme without random oracles (Xu, 2006). As Waters' scheme, the security of Xu's scheme is based on the Computational Diffie–Hellman assumption and can tolerate $t < n/4$ malicious parties. But her scheme is not proved adaptively secure.

In this paper, utilizing Abe's adaptively secure Feldman VSS scheme in secure channel model and other construction and analytical techniques, we present an adaptively secure distributed key generation (DKG) protocol for pairing based cryptosystems. As an application of the DKG protocol, based on Waters' signature scheme, we present a provably secure threshold signature scheme without random oracles. We prove security of our schemes by exhibiting a direct reduction of its security to the hardness of the Computational Diffie–Hellman Problem. Our scheme achieves optimal resilience, that is, the adversary can corrupt any $t < n/2$ players. Furthermore, both the DKG protocol and the threshold signature generation protocol achieve the adaptive security in the secure channel model, without data erasure and zero knowledge proofs. This is the first adaptively secure threshold signature scheme reached optimal resilience in the erasure-free secure channel model without random oracles and zero knowledge proofs.

The rest of this paper is organized as follows. In Section 2, we summarize the communication and adversary models and the definition of security for the threshold signature schemes. In Section 3, we give a brief review of Waters' signature scheme. In Section 4, we propose our distributed key generation protocol and prove its security against adaptive adversary. In Section 5, we present our threshold Waters' signature scheme and prove its security. Finally, Section 6 is our conclusions.

2. Preliminaries

In this section, we briefly review the computation, communication and adversary models for our threshold signature scheme. We also briefly review the definition of threshold signature scheme and its security. More details can be found in Canetti *et al.* (1999), Jarecki and Lysyanskaya (2000), Gennaro *et al.* (2001, 2003).

2.1. *Computation, Communication, and Adversary Models*

Computation Model. The computation proceeds among a set of n players $P_1, \dots, P_n$ modelled by probabilistic polynomial-time Turing machines (PPT TM), and an adversary $\mathcal{A}$, also modelled as a PPT TM. In addition, the players do not need to erase local data once it is no longer needed.

Communication Model. We assume that the players are connected by a complete network of private (i.e., untappable) and authenticated point-to-point channels. In addition, the players have access to a dedicated broadcast channel. By dedicated we mean that if a player broadcasts a message, it is received by every other player and recognized as coming from that player. The dedicated broadcast can be implemented for example by Cachin and Poritz (2002). We assume that the communication channels provide a *partially synchronous* message delivery, i.e., that computation proceeds in synchronized rounds and that the messages are received by their recipients within some specified time bound. To guarantee this round synchronization, and for simplicity of discussion, we assume that the players are equipped with synchronized clocks.

The Adversary Model. We assume the adversary $\mathcal{A}$ is *adaptive*, that is he can choose any player to corrupt at any time, based on any information he sees during the run of the protocols. He can corrupt up to t of the n players, for any value of $t < n/2$, which is the best achievable threshold. In addition he can cause the corrupted players to arbitrarily divert from the specified protocol, that is the adversary is *malicious*. On the other hand, existential unforgeability under adaptive chosen-message attacks (EUF-CMA; Goldwasser *et al.*, 1998) is a widely accepted standard notion for the security of digital signature schemes. It fits for threshold signature schemes also. Thus the adversary is permitted to request a threshold signature on any message of his choice and get it.

2.2. *The Definition of Threshold Signature Scheme and Its Security*

DEFINITION 1. Let $\mathcal{S}$ = (Key-Gen, Sig, Ver) be a signature scheme. A (t, n)-threshold signature scheme $\mathcal{TS}$ for $\mathcal{S}$ is a triple of protocols (Thresh-Key-Gen, Thresh-Sig, Ver) for the set of players $\{P_1, \ldots, P_n\}$.

Thresh-Key-Gen is a distributed key generation protocol used by the players to jointly generate a pair (x, y) of private/public keys. At the end of the protocol the private output of player P_i is a value x_i which is a secret sharing of x. This share may be a polynomial share or an additive share with respect to the threshold signature scheme. The public output of the protocol contains the public key y.

Thresh-Sig is the distributed signature protocol. The private input of P_i is the value x_i. The public inputs consist of a message m and the public key y. The output of the protocol is a value $\sigma \in Sig(m, x)$.

Ver is the verification algorithm, which is the same as in the regular signature scheme $\mathcal{S}$.

DEFINITION 2. A (t, n)-threshold signature scheme $\mathcal{TS}$ = (Thresh-Key-Gen, Thresh-Sig, Ver) is t-threshold secure if it is both t-threshold unforgeable and t-threshold robust. $\mathcal{TS}$ is t-threshold unforgeable means no malicious adversary who corrupts at most t players can produce, with non-negligible probability, the signature on any new (i.e., previously unsigned) message M, given the view of the protocol Thresh-Key-Gen and of the protocol Thresh-Sig on input messages $M_1, \ldots, M_k$ which the adversary adaptively chose. $\mathcal{TS}$ is t-threshold robust means both Thresh-Key-Gen and Thresh-Sig complete successfully except for negligible probability, even if in the presence of an adversary who corrupts maliciously at most t players.

3. Brief Review of Waters' Signature Scheme

In this section, we first present some background on groups with efficiently computable bilinear maps and the definition of Computational Diffie–Hellman problem and assumption. Then, we recall the definition of existentially unforgeable signatures. Finally, we recall the Waters' signature scheme.

3.1. *Groups and Complexity Assumption*

We briefly review the necessary facts about bilinear maps and bilinear map groups. For more detail, see, e.g., Galbrait (2005); Paterson (2005). Consider the following setting:

– $\mathbb{G}$ and $\mathbb{G}_\mathrm{T}$ are two (multiplicative) cyclic groups of prime order p;
– the group actions on $\mathbb{G}$ and $\mathbb{G}_\mathrm{T}$ can be computed efficiently;
– g is a generator of $\mathbb{G}$;
– e: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_\mathrm{T}$ is an efficiently computable map with the following properties:
    - bilinear: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_\mathrm{p}$, $e(u^a, v^b) = e(u, v)^{ab}$;
    - non-degenerate: $e(g, g) \neq 1$.

We say that $\mathbb{G}$ is a bilinear group if it satisfies these requirements. In recent years, many cryptographic schemes were proposed based on some computational hard problems in the bilinear groups (Boneh *et al.*, 2001; Boldyreva, 2003; Boneh and Boyen, 2004; Qian *et al.*, 2005; Waters, 2005; Wang *et al.*, 2005; Long *et al.*, 2006; Lu *et al.*, 2006; Xu, 2006; Kancharla *et al.*, 2007; Huang *et al.*, 2007; Chang *et al.*, 2007; Tseng *et al.*, 2008).

The security of our scheme relies on the hardness of the Computational Diffie–Hellman (CDH) problem in the bilinear groups. We state the problem and the assumption as follows.

DEFINITION 3 (CDHP on $\mathbb{G}$). Given $(g, g^a, g^b) \in_R \mathbb{G}^3$ for some unknown $a, b \in \mathbb{Z}_\mathrm{p}$, compute $g^{ab} \in \mathbb{G}$.

Define the success probability of an algorithm $\mathcal{A}$ in solving the CDHP on $\mathbb{G}$ as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{cdh}} \overset{\mathrm{def}}{=} \mathbf{Pr}\big[\mathcal{A}(g, g^a, g^b) = g^{ab}\colon\ a, b \xleftarrow{R} \mathbb{Z}_\mathrm{p}\big].$$

The probability is over the uniform random choice of g from $\mathbb{G}$, of a, b from $\mathbb{Z}_\mathrm{p}$, and the coin tosses of $\mathcal{A}$. We say that an algorithm $\mathcal{A}$ (t, ε)-breaks CDHP on $\mathbb{G}$ if $\mathcal{A}$ runs in time at most t, and $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{cdh}}$ is at least ε.

DEFINITION 4 ($(\mathbf{t}, \varepsilon)$-CDHA on $\mathbb{G}$). Given $(g, g^a, g^b) \in_R \mathbb{G}^3$ for some unknown $a, b \in \mathbb{Z}_\mathrm{p}$, no adversary (t, ε)-breaks CDHP on $\mathbb{G}$.

3.2. *Security Definition of Signature Schemes*

A signature scheme is made up of three algorithms, *Gen*, *Sign*, and *Ver*, for generating keys, signing, and verifying signatures, respectively.

Existential Unforgeability under adaptive Chosen-Message Attacks (EUF-CMA) (Goldwasser *et al.*, 1988) is a widely accepted standard notion for the security of digital signature schemes.

DEFINITION 5. An adversary $\mathcal{A}$ (t, q_s, ε)-breaks a signature scheme if $\mathcal{A}$ runs in time t, makes at most q_s signature queries and

$$\mathbf{Pr}\left[\begin{array}{l}(pk, sk) \leftarrow \mathsf{Gen}(1^k); (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}_{sk}(\cdot)}(pk): \\ \sigma \notin \Sigma^* \wedge \mathsf{Ver}_{pk}(m, \sigma) = 1\end{array}\right] \geqslant \varepsilon,$$

where Σ^* is the set of signatures received from the signing oracle. A signature scheme is (t, q_s, ε)-existentially unforgeable under adaptive chosen-message attacks if no adversary (t, q_s, ε)-breaks it.

3.3. *The Waters' Signature Scheme*

We describe the Waters' signature scheme (Waters, 2005). In our description the messages will be bit strings of the form $\{0,1\}^k$ for some fixed k. However, in practice one could apply a collision-resistant hash function H: $\{0,1\}^* \to \{0,1\}^k$ to sign messages of arbitrary length.

The scheme requires, besides the random generator $g \in \mathbb{G}$, $k+1$ additional random generators $u', u_1, \ldots, u_k \in \mathbb{G}$. In the basic scheme, these can be generated at random as part of system setup and shared by all users.

The Waters' signature scheme is a tri-tuple of algorithms $\mathcal{W}$ = (Key-Gen, Sig, Ver) described as follows.

$\mathcal{W}$.Key-Gen. Pick random $x \xleftarrow{R} \mathbb{Z}_\mathrm{p}$ and set $y \leftarrow e(g,g)^x$. The public key pk is $y \in \mathbb{G}_\mathrm{T}$. The private key sk is x.

$\mathcal{W}$.Sig(sk, M). Parse the user's private key sk as $x \in \mathbb{Z}_\mathrm{p}$ and the message M as a bit string $(m_1, \ldots, m_k) \in \{0,1\}^k$. Pick a random $r \xleftarrow{R} \mathbb{Z}_\mathrm{p}$ and compute

$$\sigma_1 \leftarrow g^x \cdot \left(u' \prod_{i=1}^{k} u_i^{m_i}\right)^r, \qquad \sigma_2 \leftarrow g^r. \tag{1}$$

The signature is $\sigma = (\sigma_1, \sigma_2) \in \mathbb{G}^2$.

$\mathcal{W}$.Ver(pk, M, σ). Parse the user's public key pk as $y \in \mathbb{G}_\mathrm{T}$, the message M as a bit string $(m_1, \ldots, m_k) \in \{0,1\}^k$, and the signature σ as $(\sigma_1, \sigma_2) \in \mathbb{G}^2$. Verify that

$$e(\sigma_1, g) \cdot e\left(\sigma_2, u' \prod_{i=1}^{k} u_i^{m_i}\right)^{-1} \overset{?}{=} y \tag{2}$$

holds; if so, output 1 (valid); if not, output 0 (invalid).

This signature scheme is existentially unforgeable under adaptive chosen-message attacks, if CDH problem is hard. Readers can refer to Corollary 1 of Lu *et al.* (2006) for a roundabout proof of this conclusion.

4. Adaptively Secure Distributed Key Generation Protocol

4.1. *The Proposed DKG Protocol*

Now we present an adaptively secure distributed key generation protocol for our threshold signature scheme. The protocol is made up of three steps in logic as follows: Generating a commitment key h; Generating a random secret key $x \in \mathbb{Z}_\mathrm{p}$; Extracting the corresponding public key $y = e(g, g)^x \in \mathbb{G}_\mathrm{T}$. The detailed protocol is presented as follows, which mainly utilizes the ideas and techniques of the adaptively secure Feldman VSS scheme in Abe and Fehr (2004), the Pedersen's VSS scheme in Pedersen (1991), the additive secret sharing in Jarecki and Lysyanskaya (2000) and the DKG protocol in Gennaro *et al.* (1999b).

Adaptively Secure DKG Protocol Thresh-Key-Gen

Input: Parameters $(\mathbb{G}, \mathbb{G}_\mathrm{T}, g, p, e)$.
Steps:
K-0. Every P_j generates a commitment-key $h_j \in \mathbb{G}$ and broadcasts it.
K-1. All players jointly generate a commitment key $h \in \mathbb{G}_\mathrm{T}$. Each P_i chooses $\eta_i \in \mathbb{Z}_\mathrm{p}$ at random and shares it as follows.

K-1.1 P_i first chooses $r_{ij} \in \mathbb{Z}_\mathrm{p}$ at random and computes commitments

$$I_{ij} = e(g, g)^j \cdot e(g, h_j)^{r_{ij}}, \quad j = 1, \ldots, n.$$

Next, he chooses $\alpha_{i1}, \ldots, \alpha_{in}$ as a random permutation of $1, \ldots, n$. Then, he selects a random polynomial

$$d_i(z) = \eta_i + d_{i1} z + \cdots + d_{it} z^t$$

over $\mathbb{Z}_\mathrm{p}[z]$ and computes

$$\eta_{ij} = d_i(\alpha_{ij}), \quad j = 1, \ldots, n$$

as well as

$$D_{i0} = e(g, g)^{\eta_i}, \quad D_{ik} = e(g, g)^{d_{ik}}, \ k = 1, \ldots, t,$$

and sets

$$H_i = D_{i0} = e(g, g)^{\eta_i}.$$

Finally, P_i broadcasts $I_{i1}, \ldots, I_{in}$ ordered in such a way that I_{ij} appears in α_{ij}-th position. Additionally, he broadcasts H_i, D_{ik} for $k = 1, \ldots, t$, and privately sends $(\alpha_{ij}, r_{ij}, \eta_{ij})$ to P_j for $j = 1, \ldots, n$.

K-1.2 Each P_j identifies I_{ij} in α_{ij}-th position and accepts the assignment if

$$I_{ij} = e(g,g)^j \cdot e(g,h_j)^{r_{ij}}, \tag{3}$$

and he accepts his share η_{ij} if it satisfies the following verification equation

$$e(g,g)^{\eta_{ij}} = \prod_{k=0}^{t} D_{ik}^{\alpha_{ij}^k}. \tag{4}$$

Otherwise, he broadcasts a *complaint* against P_i. If more than t players complain then P_i is clearly faulty and is disqualified. Otherwise, P_i reveals $(\alpha_{ij}, r_{ij}, \eta_{ij})$ such that Eq. (4) holds and I_{ij} in α_{ij}-th position satisfies Eq. (3) for each complaining player P_j. If he fails, he is also disqualified. Otherwise P_j uses the new $(\alpha_{ij}, r_{ij}, \eta_{ij})$ as his assignment and share. By convention, if P_i is disqualified then $\eta_i = 0$ and each player P_j takes default values $(\alpha_{ij}, r_{ij}, \eta_{ij})$.

K-1.3 Each P_i computes $h = \prod_{i=1}^{n} H_i$.

K-2. All players jointly generate a random secret key x. Each player P_i chooses $x_i \in \mathbb{Z}_p$ at random and shares it as follows.

K-2.1 Each player P_i selects random polynomials

$$f_i(z) = x_i + a_{i1}z + \cdots + a_{it}z^t$$

and

$$f_i'(z) = x_i' + b_{i1}z + \cdots + b_{it}z^t$$

over $\mathbb{Z}_p[z]$ of degree t, where $x_i' \in_R \mathbb{Z}_p$. Next, he computes

$$C_i = C_{i0} = e(g,g)^{x_i} h^{x_i'}, \quad C_{ik} = e(g,g)^{a_{ik}} h^{b_{ik}},\ k = 1, \ldots, t,$$

and

$$x_{ij} = f_i(\alpha_{ij}), \quad x_{ij}' = f_i'(\alpha_{ij}),\ j = 1, \ldots, n.$$

Then, he broadcasts C_i and C_{ik} for $k = 1, \ldots, t$, and privately sends (x_{ij}, x_{ij}') to P_j for $j = 1, \ldots, n$.

K-2.2 Each player P_j verifies every (x_{ij}, x_{ij}'), $i = 1, \ldots, n$, received from other players by checking whether the equation

$$e(g,g)^{x_{ij}} h^{x_{ij}'} = C_i \cdot \prod_{k=1}^{t} C_{ik}^{\alpha_{ij}{}^k} \tag{5}$$

holds. If it holds, he broadcasts *verified*, else he broadcasts a *complaint* against P_i. If there are more than t complaints against P_i, he is disqualified. Otherwise, P_i should reveal (x_{ij}, x'_{ij}) such that Eq. (5) holds. If he fails, he is also disqualified. Otherwise P_j uses the new (x_{ij}, x'_{ij}). By convention, if P_i is disqualified then $x_i = 0$ and each player P_j takes default values (x_{ij}, x'_{ij}).

K-3. All players jointly extract the corresponding public key $y = e(g,g)^x$. Each player P_i broadcasts

$$y_i = e(g,g)^{x_i}, \quad A_{ik} = e(g,g)^{a_{ik}},\ k = 1,\dots,t.$$

Each player P_j verifies the values broadcasted by the other players P_i by checking whether the equation

$$e(g,g)^{x_{ij}} = y_i \cdot \prod_{k=1}^{t} A_{ik}^{{\alpha_{ij}}^k} \tag{6}$$

holds. If the check fails, P_j complains against P_i by broadcasting the values $(\alpha_{ij}, x_{ij}, x'_{ij})$ that satisfy Eq. (5) but do not satisfy Eq. (6). For player P_i who receives at least one valid complaint, i.e., values which satisfy Eq. (5) but not Eq. (6), the other players broadcast their values $(\alpha_{ij}, x_{ij}, x'_{ij})$ received from P_i. The bad shares can be checked out through Eq. (5) and Eq. (6), and the polynomial $f_i(z)$ can be reconstructed by $t+1$ correct shares through Lagrange interpolation. Thus x_i, y_i and $A_{ik}, k = 1,\dots,t$, can be computed publicly. Finally, each player P_i computes

$$y = \prod_{i=1}^{n} y_i = e(g,g)^x$$

as the output of the key generation protocol. Each player P_i also keeps all the values he received during the above steps.

4.2. *Security Proof of the DKG Protocol*

Next, we prove that the above DKG protocol Thresh-Key-Gen is adaptively secure. We adopt the standard simulation paradigm for the security proof of protocols. Thus we first construct a simulator SIM for the Thresh-Key-Gen protocol.

This simulation is the crux of the proof of secrecy in the protocol, namely, that nothing is revealed by the protocol beyond the value $y = e(g,g)^x$. To show this, we provide the value of y as input to the simulator and require it to simulate a run of the Thresh-Key-Gen protocol that ends with y as its public output. We denote by $\mathcal{G}$ (resp. $\mathcal{B}$) the set of *currently* good (resp. bad) players. The simulator executes the protocol for all the players in $\mathcal{G}$ except one. The state of the special player P (selected at random) is used by the simulator to "fix" the output of the simulation to y, the required public key. Since the simulator does not know the discrete logarithm of y to $e(g,g)$, it does not know the x_P^* that this player contributes to the secret key corresponding to the public key y. However, by predetermining shares for a random subset of size t of the X-coordinates $\{1,\dots,n\}$, together with the implicit point $(0, y_P^*)$, SIM can compute the desired public values which are indistinguishable from the real execution of the protocol through "interpolation in exponent" or solving the system of equations. SIM can also simulate the secret values that P generates and privately sends to the other players. (The detailed method is presented in the algorithm of SIM and the proof of Theorem 1.) But, if the adversary corrupts P during the simulation (which happens with probability $<1/2$) the simulator will not be able to provide the internal state of this player. Thus, the simulator will need to rewind the adversary and select another player P'. The simulation will conclude in expected polynomial time. The above proof techniques are called rewinding the adversary and the single-inconsistent-player (SIP) in Canetti *et al.* (1999), and SIP is also used in Abe and Fehr (2004), Jarecki and Lysyanskaya (2000).

Simulator SIM for Thresh-Key-Gen Protocol

We denote by $\mathcal{B}$ the set of players currently controlled by the adversary, and by $\mathcal{G}$ the set of currently honest parties (run by the simulator). Without loss of generality, assume $\mathcal{B} = \{P_{j_1},\dots,P_{j_{t'}}\}$ and $\mathcal{G} = \{P_{j_{t'+1}},\dots,P_{j_n}\}$, $t' \leqslant t$.

Input: Parameters $(\mathbb{G}, \mathbb{G}_\mathrm{T}, g, p, e)$ and the public key $y \in \mathbb{G}_\mathrm{T}$.
Steps:

SK-0. SIM chooses $\tau_j \in \mathbb{Z}_\mathrm{p}$ at random, computes $h_j = g^{\tau_j}$ and broadcast it for each $P_j \in \mathcal{G}$.

SK-1. SIM performs Step K-1 on behalf of the uncorrupted players in $\mathcal{G}$ exactly as in protocol Thresh-Key-Gen.

SK-2. SIM performs Step K-2 on behalf of the uncorrupted players in $\mathcal{G}$ exactly as in protocol Thresh-Key-Gen.

SK-3'. SIM performs the following pre-computation:

- chooses at random one uncorrupted player $P \in \mathcal{G}$;
- computes $y_i = e(g,g)^{x_i}$ and $A_{ik} = e(g,g)^{a_{ik}}, k = 1,\dots,t$, for each player P_i except P;
- sets $y_P^* = y \cdot \prod_{P_i \neq P} (y_i)^{-1}$;
- sets $x^*_{Pj_k} = x_{Pj_k} = f_P(\alpha_{Pj_k})$, $k = 1,\dots,t'$. Randomly chooses $t - t'$ uncorrupted players $P_{j_{t'+1}},\dots,P_{j_t}$ from $\mathcal{G}$ and sets $x^*_{Pj_k} = x_{Pj_k} = f_P(\alpha_{Pj_k})$, $k = t'+1,\dots,t$. Stores the set

$$S = \left\{ \begin{array}{l} (\alpha_{Pj_1}, x_{Pj_1}, x'_{Pj_1}),\dots,(\alpha_{Pj_{t'}}, x_{Pj_{t'}}, x'_{Pj_{t'}}), \\ (\alpha_{Pj_{t'+1}}, x_{Pj_{t'+1}}, x'_{Pj_{t'+1}}),\dots,(\alpha_{Pj_t}, x_{Pj_t}, x'_{Pj_t}) \end{array} \right\};$$

– computes $A^*_{Pk}, k = 1, \dots, t$, by solving the following system of equations:

$$\begin{cases} e(g,g)^{x^*_{Pj_1}} = y^*_P \cdot \prod_{k=1}^t (A^*_{Pk})^{\alpha^k_{Pj_1}}, \\ \cdots \\ e(g,g)^{x^*_{Pj_t}} = y^*_P \cdot \prod_{k=1}^t (A^*_{Pk})^{\alpha^k_{Pj_t}}. \end{cases} \tag{7}$$

SK-3. SIM broadcasts y_i, A_{ik} for $P_i \in \mathcal{G}\backslash\{P\}$, and y^*_P, A^*_{Pk} for $k = 1, \dots, t$. SIM also performs the verification of Eq. (6) for each uncorrupted player on the values y_i, A_{ik} of $P_i \in \mathcal{B}$, broadcasted by the players controlled by the adversary. If the verification fails for some $P_i \in \mathcal{B}$ and $P_j \in \mathcal{G}$, SIM broadcasts a complaint $(\alpha_{ij}, x_{ij}, x'_{ij})$. (Notice that the corrupted players can publish a valid complaint only against one another.) For every P_i complained validly, SIM performs reconstruction on behalf of the uncorrupted parties to compute x_i, y_i and A_{ik} for $k = 1, \dots, t$.

REMARK 1. The system of equations (7) can be easily solved adopting the method used in the proof of Proposition 1 of Abe and Fehr (2004). The A^*_{Pk} for $k = 1, \dots, t$, can also be computed by "interpolation in the exponent" as follows. For each $k = 1, \dots, t$,

$$A^*_{Pk} = (y^*_P)^{\lambda_{k0}} \prod_{i=1}^{t} \left(e(g,g)^{\lambda_{ki} x_{Pj_i}}\right),$$

where $\lambda_{k0}, \dots, \lambda_{kt}$ are constants such that $\sum_{i=0}^t \lambda_{ki} f_P(\alpha_{Pj_i})$ equals the coefficient a_{Pk} of the random polynomial $f_P(z)$.

Theorem 1. *Simulator SIM for Thresh-Key-Gen on input* $(\mathbb{G}, \mathbb{G}_\mathrm{T}, g, p, e, y)$ *ends in expected polynomial time and computes a view for the adversary that is indistinguishable from a view of the protocol Thresh-Key-Gen on input* $(\mathbb{G}, \mathbb{G}_\mathrm{T}, g, p, e)$ *and output* y.

Proof. First we show that SIM outputs a probability distribution which is *identical* to the distribution the adversary sees in an execution of Thresh-Key-Gen that produce y as output.

1. The SK-0 step is carried out by choosing $\tau_j \in \mathbb{Z}_\mathrm{p}$ at random, computing and broadcasting $h_j = g^{\tau_j}$ for each $P_j \in \mathcal{G}$. Thus, h_j is a random element in $\mathbb{G}$ and has the required distribution.
2. The SK-1 step is carried out according to the protocol, thus values $(\alpha_{ij}, r_{ij}, \eta_{ij})$ for $P_i \in \mathcal{G}, P_j \in \mathcal{B}$, and I_{ij}, D_{ik} for $P_i \in \mathcal{G}$, $j = 1, \dots, n$, $k = 0, \dots, t$, have the required distribution.
3. The SK-2 step is carried out according to the protocol, thus values $f_i(\alpha_{ij})$, $f'_i(\alpha_{ij})$ for $P_i \in \mathcal{G}, P_j \in \mathcal{B}$, and C_{ik} for $P_i \in \mathcal{G}$, $k = 0, \dots, t$, have the required distribution.

4. The values y_i, A_{ik} for $P_i \in \mathcal{G}\backslash\{P\}$, $k = 1, \dots, t$, broadcasted in SK-3 step are distributed exactly as in the real protocol. The value

$$y_P^* = y \cdot \prod_{P_i \neq P} (y_i)^{-1}$$

is distributed uniformly in $\mathbb{G}_\mathrm{T}$ and independently from the values y_i for all i. This is because y is random and uniformly distributed and the y_i for $P_i \in \mathcal{B}$ are generated independently from the other ones. The values A_{Pk}^* for $k = 1, \dots, t$, are computed with respect to the t points $(\alpha_{Pj_1}, x_{Pj_1}), \dots, (\alpha_{Pj_t}, x_{Pj_t})$ and the implicit points $(0, x_P^*)$, which satisfy the verification Eq. (6). Thus they have the required distribution.

We have shown that the public view of the adversary during the simulation is identical to the one he would see during a real execution. Now we must proceed to show that the simulator can produce a consistent view of the internal states for the players corrupted by the adversary. For the adversary may adaptively corrupt the players, we consider all the cases as follows.

Case 1. If a player is corrupted by the adversary before K-0 Step, it is no need to simulate. (For the current internal state of the corrupted player is null and the action of the corrupted player in the real execution of the protocol is instructed by the adversary.)

Case 2. If a player P_j is corrupted by the adversary after K-0 Step, SIM reveals τ_j to the adversary, which is consistent with the public value h_j.

Case 3. If a player P_j is corrupted after Step K-1 or after Step K-2, then SIM reveals all the secret values generated on behalf of this player and secret values received from the other players to the adversary.

Case 4. If a player $P_j \neq P$ is corrupted after Step K-3, SIM reveals all the secret values generated on behalf of this player, and all the secret values received from the other players except P. SIM simulates the values $(\alpha_{Pj}, r_{Pj}, x_{Pj}, x'_{Pj})$ received from P as follows: Let $t' \leftarrow t' + 1$, pick $(\alpha_{Pj_{t'}}, x_{Pj_{t'}}, x'_{Pj_{t'}})$ from set S; Assume the commitment in $\alpha_{Pj_{t'}}$-th is I_{Pi}, then compute

$$r_{Pj_{t'}} = (i + r_{Pi}\tau_i - j)/\tau_j.$$

Reveal $(\alpha_{Pj_{t'}}, r_{Pj_{t'}}, x_{Pj_{t'}}, x'_{Pj_{t'}})$ to the adversary as the values received from P. It is obvious that the simulated values $(\alpha_{Pj_{t'}}, x_{Pj_{t'}}, x'_{Pj_{t'}})$ satisfy the verification Eqs. (5), (6). On the other hand, due to

$$\begin{aligned} I'_{Pj} &= e(g,g)^j \cdot e(g,h_j)^{r_{Pj_{t'}}} = e(g,g)^j \cdot e(g,g)^{\tau_j \cdot ((i + r_{Pi}\tau_i - j)/\tau_j)} \\ &= e(g,g)^i \cdot e(g,g)^{r_{Pi}\tau_i} = e(g,g)^i \cdot e(g,h_i)^{r_{Pi}} = I_{Pi}, \end{aligned}$$

thus $(\alpha_{Pj_{t'}}, r_{Pj_{t'}}, x_{Pj_{t'}}, x'_{Pj_{t'}})$ may be seen as the correct assignment and shares for P_j and they are indistinguishable from the values seen by the adversary in the real execution.

Case 5. If the player P is corrupted after Step K-3, the simulator rewinds the adversary to the beginning of Step K-3 and selects at random a different special honest player. Then, the simulator continues performing the pre-computation (Step SK-3') and Step SK-3.

For the probability that P is corrupted after Step K-3 is $<1/2$, thus in order to produce the desired view, the simulator needs to rewind the adversary only once in expectation. In summary, the simulator SIM for Thresh-Key-Gen on input $(\mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, p, e, y)$ ends in expected polynomial time and computes a view for the adversary that is indistinguishable from a view of the protocol Thresh-Key-Gen on input $(\mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, p, e)$ and output y. Thus we complete the proof.

5. Threshold Waters' Signature Scheme

5.1. *The Proposed Threshold Signature Scheme*

We present the threshold Waters' signature protocol as follows. This signature protocol generates a signature by combining the partial signatures. These partial signatures are produced by each player signing the message with his additive secret-key share. If one player cannot provide the correct partial signature, then the other players reconstruct the partial signature through Lagrange interpolation. The system parameters are identical with the Waters' signature scheme, that is $(\mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, p, e)$ and $u', u_1, \dots, u_k \in \mathbb{G}$. Like Waters' signature scheme, here we assume the message M is a bit string of the form $\{0,1\}^k$ for some fixed k. However, in practice one could apply a collision-resistant hash function H: $\{0,1\}^* \to \{0,1\}^k$ to sign messages of arbitrary length.

Threshold Signature Protocol Thresh-Sig

Inputs: Message $M = (m_1, \dots, m_k) \in \{0,1\}^k$ to be signed and the shares of x generated by the initial Thresh-Key-Gen protocol.
Outputs: A Waters' signature $\sigma = (\sigma_1, \sigma_2)$ on M.
Steps:

1. Each player P_i generates his partial signature $\sigma_i = (\sigma_{i1}, \sigma_{i2})$ by choosing $r_i \in \mathbb{Z}_{\mathrm{p}}$ at random and computing

$$\sigma_{i1} = g^{x_i} \cdot \left(u' \prod_{i=1}^{k} u_i^{m_i}\right)^{r_i}, \qquad \sigma_{i2} = g^{r_i}. \tag{8}$$

Then he broadcasts his partial signature σ_i.

2. Each partial signature is verified by checking if

$$e(\sigma_{i1}, g) \cdot e\left(\sigma_{i2}, u' \prod_{i=1}^{k} u_i^{m_i}\right)^{-1} = y_i$$

holds. If P_i's partial signature is verified invalid, then it is reconstructed by all players through Lagrange interpolation following the sub-protocol below.

- Each player P_j chooses $r_{ij} \in \mathbb{Z}_{\mathrm{p}}$ at random and computes $\sigma_{ij} = (\sigma_{ij1}, \sigma_{ij2})$, where $\sigma_{ij1} = g^{x_{ij}}(u' \prod_{l=1}^{k} u_l^{m_l})^{r_{ij}}$, $\sigma_{ij2} = g^{r_{ij}}$. Then he broadcasts $(\sigma_{ij}, \alpha_{ij})$, i.e., the share of a partial signature of P_i on message M and the assignment of P_j given by P_i.
- Each player verifies σ_{ij} by checking whether

$$e(\sigma_{ij1}, g) \cdot e\left(\sigma_{ij2}, u' \prod_{l=1}^{k} u_l^{m_l}\right)^{-1} = y_i \cdot \prod_{k=1}^{t} (A_{ik})^{\alpha_{ij}{}^{k}} \tag{9}$$

  holds. If so, it is valid. Finally, a set R is formed which contains $t+1$ valid partial signature shares and assignments.
- The partial signature of P_i is reconstructed by Lagrange interpolation:

$$\sigma_{i1} = \prod_{j \in R} (\sigma_{ij1})^{L_{ij}}, \qquad \sigma_{i2} = \prod_{j \in R} (\sigma_{ij2})^{L_{ij}},$$

  where $L_{ij} = \prod_{\alpha_{il} \neq \alpha_{ij}}^{\alpha_{il} \in R} \frac{\alpha_{il}}{\alpha_{il} - \alpha_{ij}}$ are the Lagrange interpolation coefficients.

3. The protocol outputs signature $\sigma = (\sigma_1, \sigma_2)$, where

$$\sigma_1 = \prod_{i=1}^{n} \sigma_{i1}, \qquad \sigma_2 = \prod_{i=1}^{n} \sigma_{i2}.$$

Signature Verification Algorithm Ver

The verification algorithm is identical with Waters' signature scheme. That is, for signature $\sigma = (\sigma_1, \sigma_2)$ on message $M = (m_1, \dots, m_k)$, verifying whether the equation

$$e(\sigma_1, g) \cdot e\left(\sigma_2, u' \prod_{i=1}^{k} u_i^{m_i}\right)^{-1} = y \tag{10}$$

holds. If it holds, then the signature is valid, otherwise it is invalid.

5.2. *Security Proof of the Proposed Scheme*

We firstly prove the correctness of the scheme, then prove the security of it, that is the scheme is robust and unforgeable.

Theorem 2. *The signature $\sigma = (\sigma_1, \sigma_2)$ on message $M = (m_1, \dots, m_k)$ generated by protocol Thresh-Sig is correct, that is it can be verified valid by Eq.* (10).

Proof. Firstly, we show if each partial signature $\sigma_i = (\sigma_{i1}, \sigma_{i2})$, $i = 1, \dots, n$ is valid then the threshold signature $\sigma = (\sigma_1, \sigma_2)$ is valid. Due to the protocols Thresh-Sig and

Thresh-Key-Gen, we have

$$\begin{aligned}
e(\sigma_1, g) \cdot e\Big(\sigma_2, u' \prod_{i=1}^{k} u_i^{m_i}\Big)^{-1} &= e\Big(\prod_{i=1}^{n} \sigma_{i1}, g\Big) \cdot e\Big(\prod_{i=1}^{n} \sigma_{i2}, u' \prod_{i=1}^{k} u_i^{m_i}\Big)^{-1} \\
&= \prod_{i=1}^{n} e(\sigma_{i1}, g) \cdot \prod_{i=1}^{n} e\Big(\sigma_{i2}, u' \prod_{i=1}^{k} u_i^{m_i}\Big)^{-1} \\
&= \prod_{i=1}^{n} \Big(e(\sigma_{i1}, g) \cdot e\Big(\sigma_{i2}, u' \prod_{i=1}^{k} u_i^{m_i}\Big)^{-1}\Big) \\
&= \prod_{i=1}^{n} y_i = y.
\end{aligned}$$

So, the threshold signature $\sigma = (\sigma_1, \sigma_2)$ is valid.

Then, we show each partial signature $\sigma_i = (\sigma_{i1}, \sigma_{i2}), i = 1, \dots, n$ must be valid. Due to the protocols Thresh-Sig, if the partial signature $\sigma_i = (\sigma_{i1}, \sigma_{i2})$ generated and broadcasted by player P_i is invalid, it will be checked out in Step 2. Then player P_i's partial signature σ_i is reconstructed through Lagrange interpolation and the correct partial signature is generated publicly. Thus, the threshold signature generated by protocol Thresh-Sig is correct.

Theorem 3. *Under the CDH assumption, the threshold Waters' signature scheme (Thresh-Key-Gen, Thresh-Sig) is an unforgeable t-threshold signature scheme for $t < n/2$ against any adaptive adversary in the secure channel and erasure-free model.*

Proof. Assume that the proposed threshold signature scheme is not unforgeable. Then there exists a t-threshold adversary $\mathcal{A}$ through participating in the initial execution of Thresh-Key-Gen and repeatedly executing Thresh-Sig on messages $M_1, M_2, \dots,$ of his choice, with a non-negligible probability ε, $\mathcal{A}$ outputs a valid Waters (message, signature) pair (M^*, σ^*) under the public key $(\mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, p, e, y)$ produced by Thresh-Key-Gen. Furthermore, none of M_i is equal to M^*. Using such adversary $\mathcal{A}$, we show how to construct an algorithm $\mathcal{F}$ to produce g^{ab} given an instance $(g, g^a, g^b) \in \mathbb{G}^3$ of the CDH problem. Let $\mathcal{B}$ be the set of currently corrupted players controlled by the adversary, and $\mathcal{G}$ be the set of currently good players (run by the algorithm). Without loss of generality assume that $\mathcal{B} = \{P_{j_1}, \dots, P_{j_{t'}}\}$ and $\mathcal{G} = \{P_{j_{t'+1}}, \dots, P_{j_n}\}$, $t' \leqslant t$. Let q be the upper bound on the number of signature queries that the adversary makes. $\mathcal{F}$ is constructed as follows.

Setup. Let $\lambda = 2q$. $\mathcal{F}$ picks $l \xleftarrow{R} \{0, \dots, k\}$, $\alpha', \alpha_1, \dots, \alpha_k \xleftarrow{R} \mathbb{Z}_\lambda = \{0, \dots, \lambda - 1\}$ and $\beta', \beta_1, \dots, \beta_k \xleftarrow{R} \mathbb{Z}_{\mathrm{p}}$, sets $g_1 = g^a, g_2 = g^b$ and computes

$$u' \leftarrow g_2^{\alpha' - \lambda l} g^{\beta'}, \qquad u_i \leftarrow g_2^{\alpha_i} g^{\beta_i}, \quad i = 1, \dots, k.$$

$\mathcal{F}$ gives $\mathcal{A}$ the system parameters $(g, u', u_1, \ldots, u_k)$. From the perspective of the adversary the distribution of the public parameters is identical to the real construction.

Then $\mathcal{F}$ runs an interaction between SIM and $\mathcal{A}$ on inputs $(\mathbb{G}, \mathbb{G}_{\mathrm{T}}, g, p, e)$ and $y = e(g^a, g^b)$. For (g, g^a, g^b) is an instance of the CDH problem, thus y is a random element in $\mathbb{G}_{\mathrm{T}}$. By theorem 1, the simulation ends in expected polynomial time and $\mathcal{A}$ receives a view that is identical to $\mathcal{A}$'s view of a random execution of Thresh-Key-Gen that outputs y. $\mathcal{F}$ stores all the values produced by this interaction. Notice after this interaction, $\mathcal{F}$ knows all the secret shares x_i for $i = 1, \ldots, n$ except the random selected honest player P by SIM, where x_i for $P_i \in \mathcal{B}$ is reconstructed by $\mathcal{F}$ utilizing the Lagrange interpolation (as it controls a majority of players).

Signing Queries. When $\mathcal{A}$ requests a threshold signature on $M = (m_1, \ldots, m_k) \in \{0,1\}^k$ about the public key y, $\mathcal{F}$ processes as follows.

1. Define $F = -\lambda l + \alpha' + \sum_{i=1}^{k} \alpha_i m_i$ and $J = \beta' + \sum_{i=1}^{k} \beta_i m_i$. If $F \neq 0 \bmod p$, $\mathcal{F}$ picks $r \xleftarrow{R} \mathbb{Z}_{\mathrm{p}}$ and sets

$$\sigma_1 \longleftarrow g_1^{-J/F} \left(u' \prod_{i=1}^{k} u_i^{m_i} \right)^r, \qquad \sigma_2 \longleftarrow g_1^{-1/F} g^r.$$

This is a valid Waters' signature corresponding to the public key $y = e(g,g)^{ab}$ with randomness $\tilde{r} = r - a/F$: observing that $u' \prod_{i=1}^{k} u_i^{m_i} = g_2^F g^J$, we see that

$$\sigma_1 = g_1^{-J/F} \left(u' \prod_{i=1}^{k} u_i^{m_i} \right)^r = g_2^a (g_2^F g^J)^{-a/F} (g_2^F g^J)^r = g^{ab} \left(u' \prod_{i=1}^{k} u_i^{m_i} \right)^{\tilde{r}},$$

where for the second equality we have multiplied and divided by g_2^a. Additionally, we have

$$\sigma_2 = g_1^{-1/F} g^r = g^{r - a/F} = g^{\tilde{r}}.$$

Thus the signature $\sigma = (\sigma_1, \sigma_2)$ can pass the verification Eq. (2) of Waters' signature scheme.

2. $\mathcal{F}$ randomly chooses $r_i \in \mathbb{Z}_{\mathrm{p}}$ for $i = 1, \ldots, n$ except P, computes $\sigma_i = (\sigma_{i1}, \sigma_{i2})$ using Eq. (8). Then $\mathcal{F}$ computes $\sigma_P = (\sigma_{P1}, \sigma_{P2})$ as follows:

$$\sigma_{P1} = \sigma_1 \cdot \left(\prod_{\substack{i=1 \\ i \neq P}}^{n} \sigma_{i1} \right)^{-1}, \qquad \sigma_{P2} = \sigma_2 \cdot \left(\prod_{\substack{i=1 \\ i \neq P}}^{n} \sigma_{i2} \right)^{-1}.$$

Notice σ_P is a valid Waters' signature corresponding to the partial public key y_P^* of the player P generated in SIM. Because

$$\begin{aligned}
&e(\sigma_{P1}, g) \cdot e\Big(\sigma_{P2}, u' \prod_{j=1}^{k} u_j^{m_j}\Big)^{-1} \\
&= e\Big(\sigma_1 \cdot \Big(\prod_{\substack{i=1\\ i\neq P}}^{n} \sigma_{i1}\Big)^{-1}, g\Big) \cdot e\Big(\sigma_2 \cdot \Big(\prod_{\substack{i=1\\ i\neq P}}^{n} \sigma_{i2}\Big)^{-1}, u' \prod_{j=1}^{k} u_j^{m_j}\Big)^{-1} \\
&= e(\sigma_1, g) \cdot e\Big(\sigma_2, u' \prod_{j=1}^{k} u_j^{m_j}\Big)^{-1} \cdot \prod_{\substack{i=1\\ i\neq P}}^{n} \Big(e(\sigma_{i1}, g) \cdot e\Big(\sigma_{i2}, u' \prod_{j=1}^{k} u_j^{m_j}\Big)^{-1}\Big)^{-1} \\
&= y \cdot \prod_{\substack{i=1\\ i\neq P}}^{n} y_i^{-1} = y_P^*.
\end{aligned}$$

3. $\mathcal{F}$ runs the Thresh-Sig protocol on behalf of the good players with $\mathcal{A}$. In Step 1 of the protocol, $\mathcal{F}$ broadcasts σ_i computed above for $P_i \in \mathcal{G}\backslash\{P\}$ as well as σ_P for the good player P. In Step 2 of the protocol, $\mathcal{F}$ checks the partial signatures broadcasted by $\mathcal{A}$. If an invalid one is checked out, $\mathcal{F}$ generates $t+1$ shares of the partial signature and broadcasts, then the partial signature is reconstructed by Lagrange interpolation. In Step 3 of the protocol, $\mathcal{F}$ computes σ. This execution of the Thresh-Sig protocol is identical to the real execution by all players and the obtained signature $\sigma' = (\sigma_1', \sigma_2')$ is also a valid signature corresponding to the public key y.

If $F \equiv 0 \bmod p$ in Step 1, $\mathcal{F}$ declares failure and halts.

Due to the construction of $\mathcal{F}$ and theorem 1, we can see the public view of the adversary during the simulation of the Thresh-Key-Gen protocol and the simulation of the Thresh-Sig protocol in Step 3 of Signing Queries is identical to the one he would see during a real execution of the protocols. Due to theorem 1, we can also see that the simulator of the Thresh-Key-Gen protocol can produce a consistent view of the internal states for the players corrupted by the adversary before the Thresh-Sig protocol executed. If a player $P_j \neq P$ is corrupted after Step 1 of the Thresh-Sig protocol, by using the same process as Case 4 in proof of theorem 1 and revealing the random r_j used in generating the partial signature, $\mathcal{F}$ can produce a consistent view of the internal state for P_j. If the player P is corrupted after Step 1 of the Thresh-Sig protocol, it is processed just as Case 5 in proof of theorem 1.

Output. Finally, since its view of the simulations is indistinguishable from that of the real ones, $\mathcal{A}$ outputs a signature $\sigma^* = (\sigma_1^*, \sigma_2^*)$ on a message $M^* = (m_1^*, \ldots, m_k^*)$; it must not have queried $\mathcal{F}$ on M^*. Define $F^* = -\lambda l + \alpha' + \sum_{i=1}^{k} \alpha_i m_i^*$ and

$J^* = \beta' + \sum_{i=1}^{k} \beta_i m_i^*$. If $F^* \neq 0 \bmod p$, $\mathcal{F}$ declares failure and exits. Otherwise, we have $u' \prod_{i=1}^{k} u_i^{m_i^*} = g^{J^*}$, so that

$$y = e(\sigma_1^*, g) \cdot e\Bigl(\sigma_2^*, u' \prod_{i=1}^{k} u_i^{m_i^*}\Bigr)^{-1} = e(\sigma_1^*, g) \cdot e(\sigma_2^*, g^{J^*})^{-1} = e\bigl(\sigma_1^* (\sigma_2^*)^{-J^*}, g\bigr).$$

Since $y = e(g^a, g^b) = e(g^{ab}, g)$, thus $\sigma_1^* (\sigma_2^*)^{-J^*}$ equals g^{ab}, which is the solution of the instance (g, g^a, g^b) of the CDH problem. $\mathcal{F}$ outputs it and halts.

The probability that $\mathcal{F}$ does not abort in any signing query is at least $1 - 1/\lambda$; since there are at most $q = \lambda/2$ such queries, $\mathcal{F}$ manages to answer all queries without aborting with probability at least $1/2$. Having done so, $\mathcal{F}$ then receives a forgery such that $F^* = 0 \bmod p$ with probability at least $1/(\lambda l) \geqslant 1/(2kq)$. The probability that $\mathcal{A}$ outputs a valid Waters (message, signature) pair is ε. Thus $\mathcal{F}$ succeeds with probability at least $\varepsilon/(4kq)$. $\mathcal{F}$'s run-time overhead is $\mathrm{O}(1)$ to answer each of $\mathcal{A}$'s queries and to compute the final output. Thus we have the conclusion: *the threshold Waters signature scheme is (t, q, ε)-unforgeable if Computational Diffie–Hellman is $(t + \mathrm{O}(q), 4kq\varepsilon)$-hard on $\mathbb{G}$, here q is the number of signing queries.*

It is obviously that the threshold Waters' signature scheme (Thresh-Key-Gen, Thresh-Sig) is robust against t-limited adaptive adversary for $t < n/2$. In the Thresh-Key-Gen protocol, after Step K-2, each player's contribution to secret key x is fixed. If some player P_i broadcasts invalid y_i and $A_{ik}, k = 1, \ldots, t$, then it will be checked out by Eq. (6) and the values will be reconstructed by Lagrange interpolation. In the Thresh-Sig protocol, if some player P_i broadcasts invalid partial signature, it will be checked out and reconstructed in Step 2. Combining this point and theorem 3, we have the following conclusion.

Theorem 4. *Under the CDH assumption, the threshold Waters' signature scheme (Thresh-Key-Gen, Thresh-Sig) is an secure (unforgeable and robust) t-threshold signature scheme for $t < n/2$ against adaptive adversary in the secure channel and erasure-free model.*

REMARK 2. Based on Waters' scheme, Xu proposed a provably secure threshold signature schemes (Xu, 2006). The design and security proof of her scheme mainly follows the notion of *simulatable* threshold signature scheme defined in Gennaro *et al.* (2001) and adopts the same method used in Gennaro *et al.* (2001). We found her scheme could tolerate $t < n/2$ malicious parties also for the validity of every partial signature σ_i could be checked, while she understated that it could tolerate $t < n/4$ ones. But her scheme is still not proved adaptively secure.

Though also based on Waters' scheme, our scheme is proved adaptively secure without random oracles, and no need to securely erase local data or to use the interactive zero-knowledge proofs. Our scheme can tolerate $t < n/2$ malicious parties, which is the optimal resilience. Further, our proof combines the simulation method (for proving the security of the DKG protocol) and the direct reduction to the hardness of the underlying mathematical problem method (for proving the security of the threshold signature protocol).

The main drawback of our scheme is the cost to achieve adaptive security and optimal robustness. Firstly, each party should keep all the values he receives in the run of Thresh-Key-Gen protocol, thus the storage is linear with n, the number of players. Second, when an invalid partial signature is checked out in threshold signature generating, it should be reconstructed through Lagrange interpolation, thus the computation is also linear with n. Currently, the efficiency of adaptively secure threshold cryptosystems is still an open problem. We will continue the related study.

6. Conclusions

We have presented a secure, i.e., robust and unforgeable, threshold Waters' signature scheme against the adaptive adversary, who chooses which players to corrupt at any time and based on any information he sees during the key generation and signature generation, in the secure channel model. The scheme does not need the players to securely erasing values generated during the run of the protocols. And it has been implemented completely avoiding the use of the interactive zero-knowledge proofs. The scheme achieved the optimal resilience $t < n/2$. We have proved its security by exhibiting a direct reduction of its security to the hardness of the Computational Diffie–Hellman problem without random oracles. Furthermore, to the best of our knowledge, the Thresh-Key-Gen protocol is the first adaptively secure distributed key generation protocol for pairing-based cryptosystems and it may be applied to other distributed pairing-based signature schemes.

Acknowledgements

The authors thank the anonymous reviewers for their valuable comments. This work has been partially supported by the National Natural Science General Foundation of China (Grant Nos. 60703004 and 60873217), the Research Fund for the Doctoral Program of Higher Education of China (Grant Nos. 20060269006 and 20070269005), and the Science Research Project of Shanghai Municipal Education Commission (Grant No. 06RZ019).

References

Abe, M., Fehr, S. (2004). Adaptively secure Feldman VSS and applications to universally-composable threshold cryptography. In: *Advances in Cryptology – Crypto'04, LNCS*, Vol. 3152. Springer-Verlag, pp. 317–334.

Bellare, M., Rogaway, P. (1993). Random oracles are practical: A paradigm for designing efficient protocols. In: *First ACM Conference on Computer and Communications Security*. ACM, pp. 62–73.

Bellare, M., Rogaway, P. (1996). The exact security of digital signatures: how to sign with RSA and Rabin. In: *Proceedings of Eurocrypt 1996, LNCS*, Vol. 1070. Springer-Verlag, pp. 399–416.

Boldyreva, A. (2003). Efficient threshold signature, multisignature and blind signature schemes based on the gap-Diffie–Hellman-group signature scheme. In: *Proceedings of PKC 2003*, *LNCS*, Vol. 2567. Springer-Verlag, pp. 31–46.

Boneh, D., Boyen, X. (2004). Short signatures without random oracles. In: *Advances in Cryptology – Eurocrypt'04*, *LNCS*, Vol. 3027. Springer-Verlag, pp. 56–73.

Boneh, D., Lynn, B., Shacham, H. (2001). Short signatures from the Weil pairing. In: *Proceedings of Asiacrypt 2001*, *LNCS*, Vol. 2248. Springer-Verlag, pp. 514–532.

Boyd, C. (1986). Digital multisignatures. In: *Cryptography and Coding*. Clarendon Press, Oxford, pp. 241–246.

Cachin, C., Poritz, J.A. (2002). Secure intrusion-tolerant replication on the internet. In: *Proceedings of International Conference on Dependable Systems and Networks* (*DNS-2002*). IEEE, Washington, pp. 167–176.

Canetti, R. (2000). Security and composition of multi-party cryptographic protocols. *Journal of Cryptology*, 13(1), 143–202.

Canetti, R. (2001). Universally composable security: A new paradigm for cryptographic protocols. In: *Proceedings of the 42nd IEEE Annual Symposium on Foundations of Computer Science* (*FOCS*). IEEE, pp. 136–145.

Canetti, R., Feige, U., Goldreich, O., Naor, M. (1996). Adaptively secure multi-party computation. In: *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (*STOC*). ACM, pp. 639–648.

Canetti, R., Goldreich, O., Halevi, S. (1998). The random oracle methodology, revisited. In: *Proceedings of the 30th Annual ACM Symposium on Theory of Computing* (*STOC*). ACM, pp. 209–218.

Canetti, R., Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T. (1999). Adaptive security for threshold cryptosystems. In: *Advances in Cryptology – Crypto'99*, *LNCS*, Vol. 1666. Springer-Verlag, pp. 98–115.

Chang, Y.F., Chang, C.C., Lin, P.Y. (2007). A concealed t-out-of-n signer ambiguous signature scheme with variety of keys. *Informatica*, 18(4), 535–546.

Chien, H.Y., Jan, J.K., Tseng, Y.M. (2003). Partially blind threshold signature based on RSA. *Informatica*, 14(2), 155–166.

Cramer, R., Shoup, V. (2000). Signature schemes based on the strong RSA assumption. *ACM TISSEC*, 3(3), 161–185.

Cramer, R., Damgard, I., Dziembowski, S., Hirt, M., Rabin, T. (1999). Efficient multiparty computations secure against an adaptive adversary. In: *Advances in Cryptology – EUROCRYPT'99*, *LNCS*, Vol. 1592. Springer-Verlag, pp. 311–326.

Croft, R.A., Harris, S.P. (1989). Public-key cryptography and re-usable shared secrets. In: *Cryptography and Coding*. Clarendon Press, Oxford, pp. 189–201.

Desmedt, Y. (1988). Society and group oriented cryptography: A new concept. In: *Advances in Cryptology – Crypto'87*, *LNCS*, Vol. 293. Springer-Verlag, pp. 120–127.

Desmedt, Y. (1994). Threshold cryptography. *European Transactions on Telecommunications*, 5(4), 449–457.

Desmedt, Y., Frankel, Y. (1990). Threshold cryptosystems. In: *Advances in Cryptology – Crypto'89*, *LNCS*, Vol. 435. Springer-Verlag, pp. 307–315.

Feldman, P. (1987). A practical scheme for non-interactive verifiable secret sharing. In: *Proceedings of the 28th IEEE Annual Symposium on Foundations of Computer Science* (*FOCS*). IEEE, pp. 427–437.

Frankel, Y., MacKenzie, P., Yung, M. (1999a). Adaptively-secure distributed threshold public key systems. In: *European Symposium on Algorithms – ESA'99*, *LNCS*, Vol. 1643. Springer-Verlag, pp. 16–18.

Frankel, Y., MacKenzie, P., Yung, M. (1999b). Adaptively-secure optimal resilience proactive RSA. In: *Advances in Cryptology – Asiacrypt'99*, *LNCS*, Vol. 1716. Springer-Verlag, pp. 180–195.

Galbraith, S. (2005). Pairings. In: *Advances in Elliptic Curve Cryptography, London Mathematical Society Lecture Notes*, Vol. 317. Cambridge University Press, pp. 183–213.

Gennaro, R., Halevi, S., Rabin, T. (1999a). Secure hash-and-sign signatures without the random oracle. In: *Advances in Cryptology – Eurocrypt'99*, *LNCS*, Vol. 1592. Springer-Verlag, pp. 123–139.

Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T. (1999b). The (in)security of distributed key generation in dlog-based cryptosystems. In: *Advances in Cryptology – Eurocrypt'99*, *LNCS*, Vol. 1592. Springer-Verlag, pp. 295–310.

Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T. (2001). Robust threshold DSS signatures. *Information and Computation*, 164(1), 54–84.

Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T. (2003). Secure applications of Pedersen's distributed key generation protocol. In: *CT-RSA 2003*, *LNCS*, Vol. 2612. Springer-Verlag, pp. 373–390.

Goldwasser, S., Micali, S., Rivest, R. (1988). A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, 17(2), 281–308.

Huang, Z., Chen, K., Lin, X., Huang, R. (2007). Analysis and improvements of two identity-based perfect concurrent signature schemes. *Informatica*, 18(3), 375–394.

Jarecki, S., Lysyanskaya, A. (2000). Adaptively secure threshold cryptography: Introducing concurrency, removing erasures (extended abstract). In: *Advances in Cryptology – Eurocrypt 2000*, *LNCS*, Vol. 1807. Springer-Verlag, pp. 221–242.

Kancharla, P.K., Gummadidala, S., Saxena, A. (2007). Identity based strong designated verifier signature scheme. *Informatica*, 18(2), 239–252.

Long, Y., Chen, K., Liu, S. (2006). Adaptive chosen ciphertext secure threshold key escrow scheme from pairing. *Informatica*, 17(4), 519–534.

Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B. (2006). Sequential aggregate signatures and multisignatures without random oracles. In: *Advances in Cryptology – Eurocrypt 2006*, *LNCS*, Vol. 4004. Springer-Verlag, pp. 465–485.

Lysyanskaya, A., Peikert, C. (2001). Adaptive security in the threshold setting: From cryptosystems to signature schemes. In: *Advances in Cryptology – Asiacrypt 2001*, *LNCS*, Vol. 2248. Springer-Verlag. pp. 331–350.

Paterson, K. (2005). Cryptography from pairings. In: *Advances in Elliptic Curve Cryptography, London Mathematical Society Lecture Notes*, Vol. 317. Cambridge University Press, pp. 215–51.

Pedersen, T. (1991). A threshold cryptosystem without a trusted party. In: *Advances in Cryptology – Eurocrypt'91*, *LNCS*, Vol. 547. Springer-Verlag, pp. 522–526.

Qian, H., Cao, Z., Xue, Q. (2005). Efficient pairing-based threshold proxy signature scheme with known signers. *Informatica*, 16(2), 261–274.

Shoup, V. (2000). Practical threshold Signatures. In: *Proceedings of Eurocrypt 2000*, *LNCS*, Vol. 1807. Springer-Verlag, pp. 207–220.

Tsai, C.S., S.F. Tzeng, S.F., Hwang, M.S. (2003). Improved non-repudiable threshold proxy signature scheme with known signers. *Informatica*, 14(3), 393–402.

Tseng, Y.M., Wu, T.Y., Wu, J.D. (2008). A pairing-based user authentication scheme for wireless clients with smart cards. *Informatica*, 19(2), 285–302.

Waters, B. (2005). Efficient identity-based encryption without random oracles. In: *Proceedings of Eurocrypt 2005*, *LNCS*, Vol. 3494. Springer-Verlag, pp. 114–27.

Wang, H., Zhang, Y., Feng, D. (2005). Short threshold signature schemes without random oracles. In: *Indocrypt 2005*, *LNCS*, Vol. 3797. Springer-Verlag, pp. 297–310.

Xu, J. (2006). Provably secure threshold signature schemes without random oracles. *Chinese Journal of Computers*, 29(9), 1636-01640 (in Chinese).

Z. Wang received the MS degree in computer science from East China Normal University (China) in 2003. He is currently a PhD candidate in the Computer Science Department of East China Normal University. His research interests include cryptography and network security.

H. Qian received the PhD degree in computer science from Shanghai Jiao Tong University (China) in 2006. He is currently an associate professor in the Computer Science Department of East China Normal University. His research interests include cryptography, information security and network security.

Z. Li received the PhD degree in mathematics from Lanzhou University (China) in 1988. He is currently a professor in the Computer Science Department of East China Normal University. His research interests include symbolic computation, computational complexity, cryptography and information security.

Standartinio modelio saugus adaptyvusis slenkstinis parašas

Zecheng WANG, Haifeng QIAN, Zhibin LI

Pasiūlytas paskirstytasis raktų generavimo protokolas porinėms kriptosistemoms, kuris yra adaptyviai saugus ir nereikalauja interaktyvių nulinių žinių įrodymų. Pateikiama saugi slenkstinė parašo schema, pagrįsta Waters parašo schema. Įrodyta, kad ši schema yra nesuklastojama ir atspari įsilaužėlio veiksmams, kuris gali protokolo vykdymo metu pasirinkti dalyvius ir adaptyviai atakuoti pasirinktus pranešimus. Ši schema yra optimali funkcinio atsparumo prasme.