Rerandomizable Threshold Blind Signatures

Veronika Kuchta(✉) and Mark Manulis

Department of Computing, University of Surrey, Guildford, UK
v.kuchta@surrey.ac.uk, mark@manulis.eu

Abstract. This paper formalizes the concept of *threshold blind signatures* (TBS) that bridges together properties of the two well-known signature flavors, blind signatures and threshold signatures. Using TBS users can obtain signatures through interaction with t-out-of-n signers without disclosing the corresponding message to any of them. Our construction is the first TBS scheme that achieves security in the standard model and enjoys the property of being rerandomizable. The security of our construction holds according to most recent security definitions for blind signatures by Schröder and Unruh (PKC 2012) that are extended in this work to the threshold setting.

Rerandomizable TBS schemes enable constructions of distributed e-voting and e-cash systems. We highlight how TBS can be used to construct the first e-voting scheme that simultaneously achieves privacy, soundness, public verifiability in the presence of *distributed* registration authorities, following the general approach by Koenig, Dubuis, and Haenni (Electronic Voting 2010), where existence of TBS schemes was assumed but no construction given. As a second application, we discuss how TBS can be used to distribute the currency issuer role amongst multiple parties in a decentralized e-cash system proposed by Miers et al.(IEEE S&P 2013).

1 Introduction

Blind Signatures. Blind signatures, introduced by Chaum [21], allow users to obtain a signature on some message through interaction with the signer in a way that doesn't expose the message. This property, which is called *blindness* is the distinctive property of blind signatures, in addition to the *unforgeability* requirement, which guarantees that no more signatures can be produced in addition to those output through the interaction with the signer. Blind signatures are considered as an important building block for a variety of applications, including e-voting [9,10,31,45] and e-cash schemes [21], anonymous credential systems [15] and oblivious transfer [19]. Security properties and constructions of blind signatures have been explored in numerous subsequent works: Pointcheval and Stern [52] defined and proved the security requirements for blind signatures in the random oracle model. Juels et al. [41] defined a blind signature scheme which is secure under general complexity assumptions. Recently, Schröder and Unruh [54] showed that security definitions from [52] have some drawbacks and came up with an improved definition of honest-user unforgeability. A lot of

M. Yung et al. (Eds.): INTRUST 2014, LNCS 9473, pp. 70–89, 2015.
DOI: 10.1007/978-3-319-27998-5_5

work has been done on the constructions of blind signature schemes, both in the random oracle model, e.g. [1,4,7,11,53], and in the standard model, e.g. [3,9,17,32,39,42,48,51].

Threshold Signatures. Threshold signatures, introduced by Desmedt [26] distribute the ability to sign a message across t-out-of-n signers. This distribution process is typically carried out using secret sharing techniques and is therefore helpful for the distribution of trust in various cryptographic applications. In addition, threshold signatures can be used to achieve reliability and thus improve on the availability of services. Security properties and constructions of threshold signatures have been explored in [33,47,56]. Well-known constructions of threshold signatures in the random oracle model under the RSA assumptions have been proposed by Desmedt and Frankel [27] and Shoup [56]. Boldyreva [11] showed how to construct threshold signatures in the random oracle model in Gap Diffie-Hellman groups. More recently, Li et al. [46] distributed the signing process of the well-known Waters signature scheme in the standard model under the CDH assumption in bilinear groups.

Our Contribution: Threshold Blind Signature Schemes. In this work we formalize the concept of threshold blind signatures (TBS) and present an instantiation that enables the user to obtain a signature through interaction with a distributed set of n signers on some message of user's choice without revealing any information about the message. Each signer is in possession of a secret key share which is used in the signing process. The distribution of secret key shares in our scheme is performed by a trusted dealer, albeit alternative methods, e.g. [34], can also be applied. The signature generation process cannot be forged unless the adversary corrupts t signers. The blindness property ensures that even if all n potential signers are corrupted no information about the message is leaked. When defining these properties for TBS we adapt new security definitions from Schröder and Unruh [54], introduced originally for blind signatures, to the threshold setting. The requirements modeled for blind signatures in [54] are considered as being stronger than those given previously by Pointcheval and Stern [52]. In particular, they prevent an attack by which the adversary queries the signing oracle twice on the same message and then outputs a forgery on a different message.

Our TBS scheme is built based on the techniques underlying the blind signature scheme introduced by Okamoto [51] that deploys bilinear groups. Our TBS is more than an adaptation of the scheme from [51] to a threshold setting since we introduce further changes to the original construction to enhance its performance. In particular, by using non-interactive zero-knowledge (NIZK) proof techniques from [35,37] we can remove several rounds of interaction between the user and the signers, thus obtaining the same round-optimality as in case of (non-threshold) blind signatures in [29]. The NIZK proof from [35], which is based on the DLIN assumptions, gives us concurrent security for the overall TBS construction in the Common Reference String (CRS) model.

The standard assumptions and stronger definitions of security make our scheme superior to the existing TBS constructions from [43,57] that were proven

secure in the random oracle model with respect to the (weaker) definitions from [52], which in turn makes them vulnerable to attacks against blind signatures identified in [54]. Our TBS construction enjoys the re-randomization property, which makes it especially attractive for a range of applications such as distributed e-voting and e-cash. We show how our TBS scheme can be used to realize e-voting in presence of distributed registration authorities and decentralized e-cash in presence of distributed currency issuers.

Applications of TBS. The use of blind signatures in e-voting schemes goes back to Chaum [21] and various e-voting schemes utilizing blind signatures have been introduced since then, e.g. [6,8,9,22,31,50]. The blindness property in most e-voting constructions is necessary to ensure privacy of the submitted votes, while the unforgeability property is used for authentication. The corresponding signature is typically issued by the registration authority, which is supposed to check the voter is eligible to participate in the election. The use of threshold blind signatures in this context is a helpful alternative for the case where the registration authority needs to be distributed across multiple not necessarily fully trustworthy entities. Such distributed approach for voter registration has been proposed by Koenig, Dubuis, and Haenni [44] assuming existence of threshold blind signatures, yet without offering concrete constructions of this primitive. As proven in [44], existence of a public registration board is necessary in order to prevent potential abuses. Public verifiability, originally defined in [40], is a property that guarantees the validity of the election outcome, preventing voting authorities from biasing the results. We show that our re-randomizable TBS construction can be used to obtain an e-voting scheme where the registration authority can be distributed across multiple parties and where the property of public verifiability holds simultaneously. In our construction we follow the template from [44]. Our scheme also achieves public verifiability as it was required in [44] because the voters send their votes together with signatures to a public board such that each voter can complain if he does not find his vote on the board or if he is generally suspicious about the content on the board. We provide an publicly verifiable e-voting scheme, which guarantees extended security in the signing process because of the threshold setting. Since the power of one signing authority is distributed amongst a number of signers, the signature on a vote will be accepted if and only if t out of n signers provide their signatures on the blinded vote to public board.

Our TBS scheme can be used to construct distributed e-cash. The concept of e-cash was introduced by Chaum [21] and later refined in [13,14,23,30,41,52]. A threshold approach was used by Camenisch et al. [18] in the design of endorsed e-cash schemes to provide fairness for the user. By utilizing threshold setting, the user creates n endorsements for one coin, of which any t can be used to reconstruct the coin. The e-cash scheme by Zhou [59] uses threshold cryptography to enable traceability of the issued e-coins. The secret sharing of the key and probabilistic encryption algorithm enable threshold management of private key and the scheme avoids the misuse of identity tracing and currency tracing in fair e-cash scheme. Miers et al. [49] recently described the common problem of many e-cash protocols

that fundamentally rely on the issuer of e-coins being trusted and mentioned the distribution of his role amongst multiple issuers as a possible solution. We describe how our TBS scheme can offer such a standard-model solution for distributed e-cash schemes.

2 Building Blocks and Hardness Assumptions

In this section we recall several hardness assumptions and building blocks that will be used in our work.

Definition 1 (Bilinear Groups). *Let $\mathcal{G}(1^\lambda)$, $\lambda \in \mathbb{N}$ be an algorithm that on input a security parameter 1^λ outputs the description of two cyclic groups $\mathbb{G}_1 = \langle g_1 \rangle$ and $\mathbb{G}_2 = \langle g_2 \rangle$ of prime order q with $|q| = 1^\lambda$, where possibly $\mathbb{G}_1 = \mathbb{G}_2$, and an efficiently computable $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with $\mathbb{G}_T$ being another cyclic group of order q. The group pair $(\mathbb{G}_1, \mathbb{G}_2)$ is called* bilinear *if $e(g_1, g_2) \neq 1$ and $\forall u \in \mathbb{G}_1,\ v \in \mathbb{G}_2,\ \forall a, b \in \mathbb{Z} :\ e(u^a, v^b) = e(u, v)^{ab}$.*

Definition 2 (DLin-Assumption). *Let $\mathbb{G}$ be a cyclic group of order q. The DLin assumption states that given a tuple $\left(g, g^x, g^y, g^{xa}, g^{yb}, g^c\right)$ for random $a, b, x, y, c \in \mathbb{Z}_q^*$, it is hard to decide whether $c = a + b$. When $(g, u = g^x, v = g^y)$ is fixed, a tuple $\left(u^a, v^b, g^{a+b}\right)$ is called a linear tuple, whereas a tuple $\left(u^a, v^b, g^c\right)$ for a random and independent c is called a random tuple. Adversaries advantage in solving the assumption is negligible.*

Definition 3 (CDH-Assumption). *Let $\mathbb{G}, \mathbb{G}_T$ be two groups of prime order q. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map and let $\langle g \rangle = \mathbb{G}$ be the generator of $\mathbb{G}$. Let $\mathcal{A}_{CDH}$ be an adversary taking as input the security parameter λ. Suppose that $a, b \leftarrow \mathbb{Z}_q^*$ are randomly chosen. $\mathcal{A}_{CDH}$ is to solve the following problem: Given g, g^a, g^b compute the g^{ab}. Let ϵ be the advantage of algorithm $\mathcal{A}$ in solving the CDH assumption if*

$$\left| Pr[\mathcal{A}(g, g^a, g^b) = g^{ab}] \right| \geq \epsilon(\lambda).$$

Non-Interactive Zero-Knowledge Proof [36]**.** A non-interactive proof system $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ for a relation R with setup consists of four PPT algorithms: a setup algorithm $\mathcal{G}$, a common reference string (CRS) generation algorithm $\mathcal{K}$, a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. The setup algorithm outputs public parameters I and a commitment key ck. The CRS generation algorithm takes I as input and outputs a CRS ρ. The prover $\mathcal{P}$ takes as input (I, ρ, x, ω), where x is the statement and ω is the witness, and outputs a proof π. The verifier $\mathcal{V}$ takes as input (I, ρ, x, π) and outputs 1 if the proof is acceptable and 0 otherwise. $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is non-interactive proof system for R if it has the following properties:

Completeness. A non-interactive proof is complete if an honest prover can convince an honest verifier whenever the statement belongs to the language and the prover holds a witness testifying to this fact. For all adversaries $\mathcal{A}$ we have: $Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \rho \leftarrow \mathcal{K}(I, ck); (x, \omega) \leftarrow \mathcal{A}(I, \rho); \pi \leftarrow \mathcal{P}(I, \rho, x, \omega) : \mathcal{V}(I, \rho, x, \pi) = 1 \text{ if } (I, x, \omega) \in R] = 1$.

Soundness. A non-interactive proof is sound if it is impossible to prove a false statement. We say $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is perfectly sound if for all adversaries $\mathcal{A}$ we have: $Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \rho \leftarrow \mathcal{K}(I, ck); (x, \pi) \leftarrow \mathcal{A}(I, \rho); \pi \leftarrow \mathcal{P}(I, \rho, x, \omega) : \mathcal{V}(I, \rho, x, \pi) = 0 \text{ if } x \notin L] = 1$.

Knowledge Extraction. We say that $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is a proof of knowledge for R if there exists a knowledge extractor $\mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$ with the following properties: For all PPT adversaries $\mathcal{A}$ we have $Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \rho \leftarrow \mathcal{K}(I, ck) : \mathcal{A}(I, \rho) = 1] = Pr[I \leftarrow \mathcal{G}(1^\lambda); (\rho, \xi) \leftarrow \mathcal{E}_1(I) : \mathcal{A}(I, \rho) = 1]$. For all adversaries $\mathcal{A}$ holds $Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); (\rho, \xi) \leftarrow \mathcal{E}_1(I, ck); (x, \pi) \leftarrow \mathcal{A}(I, \rho); \omega \leftarrow \mathcal{E}_2(\rho, \xi, x, \pi) : \mathcal{V}(I, \rho, x, \pi) = 0 \text{ or } (x, \omega) \in R] = 1$.

Zero-Knowledge. We say that $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ is a NIZK proof if there exists a PPT simulator $(\mathcal{S}_1, \mathcal{S}_2)$ such that for all PPT adversaries $\mathcal{A}$ we have $Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); \rho \leftarrow \mathcal{K}(I, ck) : \mathcal{A}(I, \rho) = 1] \approx Pr[I \leftarrow \mathcal{G}(1^\lambda); (\rho, \tau) \leftarrow \mathcal{S}_1(I) : \mathcal{A}(I, \rho) = 1]$, and for all adversaries $\mathcal{A}$: $Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); (\rho, \tau) \leftarrow \mathcal{S}_1(I, ck); (x, \omega) \leftarrow \mathcal{A}(I, \rho, \tau); \pi \leftarrow \mathcal{P}(I, \rho, x, \omega) : \mathcal{A}(\pi) = 1] = Pr[(I, ck) \leftarrow \mathcal{G}(1^\lambda); (\rho, \tau) \leftarrow \mathcal{S}_2(I, ck); (x, \omega) \leftarrow \mathcal{A}(I, \rho, \tau); \pi \leftarrow \mathcal{P}(I, \rho, x, \omega) : \mathcal{A}(\pi) = 1]$, where $\mathcal{A}$ outputs $(I, x, \omega) \in R$.

3 Threshold Blind Signatures

A threshold blind signature scheme gives the user the ability to get a signature on a message without revealing its content and it distributes the secret key among a certain number of signers. We observe a $t-$out-of-n threshold blind signature scheme. It means that it is not possible to construct a valid blind signature on a message by contacting less than t-out-of-n servers. The threshold blind signature scheme is applicable to many constructions of cryptographic schemes because of its role in the decentralization the power of the signer.

Definition 4 (Threshold Blind Signature). *A t-out-of-n threshold blind signature scheme TBS in a Common Reference String model consists of the following four algorithms:*

`TBParGen`(1^λ): *A PPT algorithm takes as input the security parameter 1^λ and outputs public parameters I (possibly containing a common reference string crs_{TBS}).*

`KGen`(I): *On input public parameters I this algorithm outputs a secret share sk_i for each signer S_i, $i \in \{1, \ldots, n\}$ and a public key pk.*

`TBSign`$(\cdot)$: *This is a protocol between a user $\mathcal{U}$ and the signers S_i, $i \in \{1, \ldots, n\}$. The input of $\mathcal{U}$ is pk and a message m. The input of each server S_i is the secret share sk_i. The protocol results in a signature σ output by $\mathcal{U}$.*

`TBVerify`(pk, m, σ): *A deterministic algorithm which on input a public key pk, message m, a signature σ outputs 1 if the signature is valid and 0 otherwise.*

TBS Unforgeability. We recall the unforgeability definition for blind signatures by Schröder and Unruh [54] and adopt it to the threshold setting. This definition requires that $(m_i^*, \sigma_i^*) \neq (m_j, \sigma_j)$ for all i, j and $(m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$

for i, j with $i \neq j$, which in comparison to the earlier definition by Pointcheval and Stern [52] allows to tell which message is being signed in a given interaction. It is assumed that the adversary randomly chooses up to $(t-1)$ out of n servers. When an adversary corrupts a server, it is given the entire computation history of that server, and it gets control of the server for the running time of the system. An adversary against unforgeability of TBS has the target to generate $q_S + 1$ valid message/signature pairs after it has interacted at most q_S times with the honest signer.

Definition 5 (Unforgeability). *A threshold blind signature scheme TBS* $= (\texttt{TBParGen}, \texttt{KGen}, \texttt{TBSign}, \texttt{TBVerify})$ *is* unforgeable *if for all PPT adversaries* $\mathcal{A}$ *the probability that the following experiment* $\texttt{Unforge}_{\mathcal{A}}^{TBS}(\lambda)$ *evaluates to 1 is negligible in the security parameter* λ.

1. $I \leftarrow \texttt{TBParGen}(1^\lambda)$
2. $(sk_i, pk) \leftarrow \texttt{KGen}(I)$ *for all* $i \in \{1, \ldots, n\}$
3. $\{i_1, \ldots, i_{n-t+1}\} \leftarrow \mathcal{A}(pk)$
4. $\left(\sigma_1^*, m_1^*, \ldots, \left(\sigma_{q_S+1}^*, m_{q_S+1}^*\right)\right) \leftarrow \mathcal{A}^{\mathcal{O}\texttt{TBSign}(\cdot)}(sk_1, \ldots, sk_{t-1})$.
5. *If* $\texttt{TBVerify}(pk, m_i^*, \sigma_i^*) = 1$ *for all* $i \in [1, q_S + 1]$ *and* $(m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$ *for all* $j \in [1, q_{S+1}]$, $j \neq i$ *then return 1, otherwise return 0.*

$\mathcal{O}\texttt{TBSign}(\cdot)$ *is an oracle that executes the* $\texttt{TBSign}(sk_i, m)$ *protocol on behalf of all uncorrupted servers* $i_1, \ldots, i_{n-t+1}$. *The total number of invoked* TBSign *protocol sessions is denoted by* q_S.

TBS Blindness. The TBS blindness property prevents signers from linking generated signatures to corresponding sessions of the signing protocol. Therefore, it should be impossible for a malicious signer $\mathcal{A}$ to decide on the order in which two messages, m_0 and m_1, were signed in two protocol sessions with an honest user $\mathcal{U}$.

Definition 6 (Blindness). *A threshold blind signature scheme TBS* $= (\texttt{TBParGen}, \texttt{KGen}, \texttt{TBSign}, \texttt{TBVerify})$ *is called* blind *if for any PPT adversary* $\mathcal{A}$ *the probability that the following experiment* $\texttt{TBlind}_{\mathcal{A}}^{TBS}(\lambda)$ *evaluates to* 1 *exceeds* $1/2$ *by at most a negligible amount in the security parameter* λ.

1. $I \leftarrow \texttt{TBParGen}(1^\lambda)$
2. $(m_0, m_1, pk, st_{find}) \leftarrow \mathcal{A}(I, \texttt{find})$
3. *Choose* $b \xleftarrow{r} \{0, 1\}$
4. *Execute* $\sigma_b \leftarrow \texttt{TBSign}(pk, m_b)$ *and* $\sigma_{1-b} \leftarrow \texttt{TBSign}(pk, m_{1-b})$ *sessions on behalf of user* $\mathcal{U}$. *If* $\sigma_b = \perp$, *or* $\sigma_{1-b} = \perp$ *then* $(\sigma_b, \sigma_{1-b}) \leftarrow (\perp, \perp)$.
5. $b^* \leftarrow \mathcal{A}(\texttt{guess}, \sigma_0, \sigma_1)$.
6. *If* $b = b^*$, *then return* 1, *otherwise return* 0.

A Note on Key Generation. There exist several approaches for the distribution of keys amongst multiple signers. The approach by Shamir [55] applies secret sharing and distributes secret key shares to corresponding signers through a trusted dealer. The protocol by Feldman [28] minimizes this trust assumption

on the dealer by requiring the latter to broadcasts information that can then be used by the signers to individually check the validity of their shares and detect incorrect shares at reconstruction time. The key generation protocol by Gennaro et al. [34] proceeds in a pure distributed fashion, where each signer defines its own share of the secret key and participates in a protocol with all remaining signers to setup the key. During the protocol parties can determine malicious signers those contributions will be dropped. The distributed key generation protocol by Abe and Fehr [2] for discrete logarithm-based keys achieves adaptive security in the non-erasure model and avoids the use of interactive zero knowledge proofs.

4 TBS Construction in the Standard Model

4.1 Our TBS Scheme

In this section we present our TBS scheme based on the techniques underlying the Okamoto's blind signature scheme [51] and the NIZK proof from [36]. We assume existence of a trusted dealer for the distribution of secret key shares.

Parameter Generation: The algorithm $\texttt{TBParGen}(1^\lambda)$ outputs the common reference string $CRS = (\mathbb{G}, \mathbb{G}_T, q, g, e, ck)$, where $ck = (u_k^{'}, u_{k,j})$, $j = \{1, \ldots, n\}$, and $k = \{1, 2, 3\}$ is the commitment key. The perfect binding key consists of the following values $u_{1,j} = \left(u_1^{'}\right)^{\xi_{1,j}}, u_{2,j} = \left(u_2^{'}\right)^{\xi_{2,j}}, u_{3,j} = \left(u_3^{'}\right)^{\xi_{1,j}+\xi_{2,j}+\zeta}$; $\xi_{1,j}, \xi_{2,j}, \zeta \xleftarrow{r} \mathbb{Z}_q^*$ and $u_1^{'} = g^\rho, u_2^{'} = g^\tau, u_3^{'} = g$. The corresponding extraction key is given by $xk = (ck, \rho, \tau, \zeta)$. During the generation process of perfectly hiding key, the algorithm outputs the following trapdoor key $tk_j = (ck, \xi_{1,j}, \xi_{2,j})$, $j = \{1, \ldots, \ell\}$.

Key Generation: The algorithm $\texttt{KGen}(I)$ picks $x \xleftarrow{r} \mathbb{Z}_q$, computes $g_1 = g^x$, It then picks a random polynomial $f \xleftarrow{r} \mathbb{Z}_q[Z]$ of degree $t-1$, with $t \leq n$ being a threshold and $f(0) = x$. Let $f(z) = \sum_{i=1}^{t-1} a_i z_i$. The algorithm computes $x_i = f(i)$ for each server $i \in \{1, \ldots, n\}$. Let $\boldsymbol{vk} = (vk_1, \ldots, vk_n) = (g^{x_1}, \ldots, g^{x_n})$. The outputs consists of the public key $pk = (g_1, g_2, \boldsymbol{vk})$ and a separate secret share $sk_i = g_2^{x_i}$ for each S_i, $i \in \{1, \ldots, n\}$.

Signature generation: The $\texttt{TBSign}$ protocol on a ℓ-bit message $m = (\mu_1, \ldots, \mu_\ell)$ proceeds in two stages:

Stage 1: For all $i = \{1, \ldots, n\}$, user $\mathcal{U}$ chooses a random $r_i \xleftarrow{r} \mathbb{Z}_q^*$ and computes $X_i \leftarrow \left(u_1^{'} \prod_{j=1}^{\ell} u_{1,j}^{\mu_j}\right)^{r_i}$. $\mathcal{U}$ then prepares a NIZK proof for the well-formedness of X_i. This proof consists of two parts $\pi_i^{(1)}$ and $\pi_i^{(2)}$. It first part $\pi_i^{(1)}$ proves that all μ_j are bits using the NIZK proof from [37]. The user randomly selects $\alpha_{k,j} \xleftarrow{r} \mathbb{Z}_q^*$ for $k = 1, 2, 3$ and computes $A_{k,j} = \left(u_k^{'}\right)^{\alpha_{k,j}} u_{k,j}^{\mu_j}$ for $j = \{1, \ldots, \ell\}, k = \{1, 2, 3\}$. U proves to each server S_i knowledge of α_j such that $A_{k,j} = \left(u_k^{'}\right)^{\alpha_{k,j}}$ for $\mu_j = 0$ or $A_{k,j} = \left(u_k^{'}\right)^{\alpha_{k,j}} u_{k,j}$ for $\mu_j = 1$

and $j = \{1, \ldots, \ell\},\ k = \{1, 2, 3\}$. For each S_i the corresponding NIZK proof $\pi_i^{(1)} = (\bar{\pi}_1, \ldots, \bar{\pi}_\ell)$ consists of ℓ components $\bar{\pi}_j,\ j = \{1, \ldots, \ell\}$. Each of these proofs $\bar{\pi}_j = (\pi_{11}, \pi_{12}, \pi_{13}, \pi_{21}, \pi_{22}, \pi_{23})$ is computed as follows using a randomly chosen $t_j \xleftarrow{r} \mathbb{Z}_q^*$:

$$
\begin{aligned}
\pi_{11} &= \left(u_{1,j}^{2\mu_j - 1} \left(u_1^{'}\right)^{\alpha_{1,j}}\right)^{\alpha_{1,j}} \\
\pi_{12} &= u_{2,j}^{(2\mu_j - 1)\alpha_{2,j}} \left(u_2^{'}\right)^{\alpha_{1,j}\alpha_{2,j} - t_j} \\
\pi_{13} &= u_{3,j}^{(2\mu_j - 1)\alpha_{1,j}} \left(u_3^{'}\right)^{(\alpha_{1,j} + \alpha_{2,j})\alpha_{1,j} + t_j} \\
\pi_{21} &= u_{1,j}^{(2\mu_j - 1)\alpha_{2,j}} \left(u_1^{'}\right)^{\alpha_{1,j}\alpha_{2,j} + t_j} \\
\pi_{22} &= \left(u_{2,j}^{2\mu_j - 1} \left(u_2^{'}\right)^{\alpha_{2,j}}\right)^{\alpha_{2,j}} \\
\pi_{23} &= u_{3,j}^{(2\mu_j - 1)\alpha_{2,j}} \left(u_3^{'}\right)^{(\alpha_{1,j} + \alpha_{2,j})\alpha_{2,j} - t_j}
\end{aligned}
$$

U sends the proofs π_i and the commitments $\{A_{k,j}\}_{k=\{1,2,3\}, j=\{1,\ldots,\ell\}}$ to the corresponding server S_i that checks the following verification equations:

$$
\begin{aligned}
e(u_1^{'}, \pi_{11}) &= e(A_{1,j}, A_{1,j} u_{1,j}^{-1}), \\
e(u_2^{'}, \pi_{22}) &= e(A_{2,j}, A_{2,j} u_{2,j}^{-1}), \\
e(u_3^{'}, \pi_{33}) &= e(A_{3,j}, A_{3,j} u_{3,j}^{-1}), \\
e(u_1^{'}, \pi_{12}) e(u_2^{'}, \pi_{21}) &= e(A_{1,j}, A_{2,j} u_{2,j}^{-1}) e(A_{2,j}, A_{1,j} u_{1,j}^{-1}), \\
e(u_1^{'}, \pi_{13}) e(u_3^{'}, \pi_{31}) &= e(A_{1,j}, A_{3,j} u_{3,j}^{-1}) e(A_{3,j}, A_{1,j} u_{1,j}^{-1}), \\
e(u_3^{'}, \pi_{23}) e(u_3^{'}, \pi_{32}) &= e(A_{2,j}, A_{3,j} u_{3,j}^{-1}) e(A_{3,j}, A_{2,j} u_{2,j}^{-1}),
\end{aligned}
$$

for each $j = \{1, \ldots, \ell\}$ and $\pi_{33} = \pi_{1t}\pi_{2t},\ t = \{1, 2, 3\}$. The server accepts $\pi_i^{(1)}$ if all verification equations hold.

In the second part $\pi_i^{(2)}$ user $\mathcal{U}$ proves to each server S_i the knowledge of $\{r_i, \beta_{k,i}, \delta_{i,j}\}_{k \in [3], j \in [\ell]}$ using the NIZK techniques from [35] and values $A_{k,j}$, $\alpha_{k,j}\ k \in \{1, 2, 3\}; j \in \{1, \ldots, \ell\}$ that were used to compute $\pi_i^{(1)}$ by proving that $X_i = \left(\prod_{j=1}^{\ell} A_{1,j}\right)^{r_i} (u_1^{'})^{\beta_{1,i}}$ and $X_i = (u_1^{'})^{r_i} \prod_{j=1}^{\ell} u_{1,j}^{\delta_{i,j}}$, where $\beta_{1,i} = r_i - r_i \sum_{j=1}^{\ell} \alpha_{1,j}$ and $\delta_{i,j} = r_i \mu_j,\ j \in [\ell], i \in [n]$. This proof involves building commitments $B_{k,i} = \left(\prod_{j=1}^{\ell} A_{k,j}\right)^{r_i} (u_k^{'})^{\beta_{k,i}}$ and $\hat{B}_{k,i} = (u_k^{'})^{r_i} \prod_{j=1}^{\ell} u_{k,j}^{\delta_{i,j}}$, $k = \{1, 2, 3\}, i = \{1, \ldots, n\}, j = \{1, \ldots, \ell\}$. Note that $B_{1,i} = \hat{B}_{1,i} = X_i$. This effectively binds both parts of the proof to X_i. $\mathcal{U}$ splits $B_{k,i}$ and $\hat{B}_{k,i}$ into ℓ commitments such that $B_{k,i,j} = A_{k,j}^{r_i} \left(u_k^{'}\right)^{\beta_{k,i}}$ and $\hat{B}_{k,i,j} = \left(u_{k,j}^{\delta_{i,j}} (u_k^{'})^{r_i}\right)$.

The user makes then a NIZK proof for the Pedersen commitment for each of these components. We refer to Sect. 4.5 [35] for further details on the construction of $\pi_i^{(2)}$ proof that is used in this second part. Each $\pi_i^{(2)}$ consists of $6(\ell-1)+2$ components. Each server $S_i, i = \{1,\ldots,n\}$ verifies $\pi_i^{(2)}$ and proceeds if the proof is valid.

Stage 2: If S_i accepts the NIZK proof in Stage 1, it randomly chooses $d_i \xleftarrow{r} \mathbb{Z}_q^*$ and uses its secret key share $sk_i = g_2^{x_i}$ to compute $Y_{i1} \leftarrow sk_i X_i^{d_i}$ and $Y_{i2} \leftarrow g^{d_i}$. Finally, S_i sends its signature share $\sigma_i = (Y_{i1}, Y_{i2})$ to $\mathcal{U}$. For each received $\sigma_i = (\sigma_{i1}, \sigma_{i2})$, $\mathcal{U}$ checks the equation $e(Y_{i1}, g) = e(g_2, vk_i) \cdot e(X_i, Y_{i2})$ using the corresponding verification key $vk_i \in \boldsymbol{vk}$ and if successful chooses a random $s_i \xleftarrow{r} \mathbb{Z}_q^*$, and computes

$$\sigma_{i1} \leftarrow Y_{i1}\left(u' \prod_{j=1}^{\ell} u_j^{\mu_j}\right)^{s_i} \quad \text{and} \quad \sigma_{i2} \leftarrow Y_{i2}^{r_i} g^{s_i}.$$

Assume that $\mathcal{U}$ collected t shares σ_i from corresponding servers S_i, $i = 1,\ldots,t$. $\mathcal{U}$ first computes the Lagrange coefficients $\lambda_1,\ldots,\lambda_t \in \mathbb{Z}_q$ such that $x = f(0) = \sum_{i=1}^{t} \lambda_i f(i)$ and then $\sigma_1 = \prod_{i=1}^{t}(\sigma_{i1})^{\lambda_i}$ and $\sigma_2 = \prod_{i=1}^{t}(\sigma_{i2})^{\lambda_i}$. Finally, $\mathcal{U}$ outputs $\sigma = (\sigma_1, \sigma_2)$ as the resulting signature. (Note that σ has the same form as in the Okamoto's blind signature scheme from [51]).

Verification: The algorithm $\texttt{TBVerify}(pk, m, \sigma)$ first parses pk as $(g_1, g_2, u', (u_1,\ldots,u_\ell))$, m as $(\mu_1,\ldots,\mu_\ell)$, and σ as (σ_1, σ_2) and outputs 1 if and only if $e(\sigma_1, g) = e(g_2, g_1) \cdot e\left(u' \prod_{j=1}^{\ell} u_j^{\mu_j}, \sigma_2\right)$.

4.2 Security Analysis

The unforgeability of our TBS scheme is proven in Theorem 1 through a direct reduction to the CDH assumption. Note that the blind signature scheme by Okamoto [51] those techniques we partially apply in TBS was proven to be unforgeable using a reduction to the original Waters signature scheme [58] that in turn holds under the CDH assumption.

Theorem 1 (Unforgeability). *Our TBS scheme is unforgeable in the common reference string model assuming the hardness of the CDH assumption from Definition 3 and the soundness property of the NIZK proof from [36].*

Proof To prove the above theorem we construct a simulator $\mathcal{C}$ which is given the CDH challenge (g, g^a, g^b) from Definition 3 and is internally using the unforgeability adversary $\mathcal{A}$ to compute g^{ab}. By ϵ we denote the success probability of $\mathcal{A}$ in forging the threshold blind signature. The interaction of $\mathcal{C}$ with $\mathcal{A}$ proceeds according to the following description.

Setup: To generate the public parameters the challenger $\mathcal{C}$ sets $l = 4q_S$ and chooses a random vector of length ℓ: $\boldsymbol{a} = (a_1,\ldots,a_\ell)$, where each is chosen

uniformly and random in the interval between 0 and $l-1$ and ℓ denotes the number of bits of a message m. Then it chooses a random $b' \xleftarrow{r} \mathbb{Z}_q$, and the vector $\boldsymbol{b} = (b_1, \ldots, b_\ell) \xleftarrow{r} \mathbb{Z}_q$. Next the challenger $\mathcal{C}$ sets the following public parameters $u' = g_1^{q-tl+a'} g^{b'}$ and $u_j = g_1^{a_j} g^{b_j}$, where t is the threshold number of the scheme. The public parameters $(g, g_1, g_2, u', \boldsymbol{u})$ are sent to the adversary $\mathcal{A}$. We assume for our scheme that the adversary corrupts $t-1$ servers $S_{i_1}, \ldots, S_{i_{t-1}}$. Let $\hat{S}$ be the set of indexes i_k of corrupted servers.

The challenger $\mathcal{C}$ generates secret shares sk_i of the private key sk for the corrupted servers in the following way: It picks $t-1$ random integers $x_1, \ldots, x_{t-1} \in \mathbb{Z}_q$. Let $f \in \mathbb{Z}_q[Z]$ be the $t-1$ polynomial, which satisfies $f(0) = x_i$ for $i = 1, \ldots, t-1$. The challenger $\mathcal{C}$ gives the secret key shares $sk_i = g_2^{x_i}$ to the adversary $\mathcal{A}$.

The challenger $\mathcal{C}$ also generates the verification keys vk_i, which are useful to prove the correctness of secret shares. It sets $vk_i = g^{f(i)}$, such that the verification keys generate a vector $\left(vk_1, \ldots, vk_n\right) = \left(g^{f(1)}, \ldots, g^{f(n)}\right)$ for the above defined polynomial f. It is easy for the challenger $\mathcal{C}$ to construct the verification keys for the corrupted servers from the set $\hat{S}$, because $f(i)$ equals to x_i, which are known to the challenger $\mathcal{C}$. Let $\tilde{S}$ denote the set of uncorrupted servers $(S_{i_t}, \ldots, S_{i_n})$. $\mathcal{C}$ has to compute the Lagrange coefficients $\lambda_{0,i}, \ldots, \lambda_{t-1,i} \in \mathbb{Z}_q$ such that $f(i) = \lambda_{0,i} f(0) + \sum_{k=1}^{t-1} \lambda_{k,i} f(k)$, where $\{i_1, \ldots, i_{t-1}\}$ are the indexes from the set $\hat{S}$ of corrupted servers and $\{i_t, \ldots, i_n\}$ are the indexes of uncorrupted servers. The Lagrange coefficients are then computed as follows:

$$\lambda_{k,i} = \prod_{k' \in \tilde{S} \setminus \{i\}} \frac{(k-k')}{(i-k')},$$

where $k \in \hat{S}$ is the index of a corrupted server and k' is the index of an uncorrupted server. It is easy to determine these Lagrange coefficients because they are independent from f. As a next step, $\mathcal{C}$ sets for $i \in \tilde{S}$ and $g_1 = g^x$:

$$\begin{aligned} vk_i &= g_1^{\lambda_{0,i}} vk_1^{\lambda_{1,i}} \cdots vk_{t-1}^{\lambda_{t-1,i}} = g_1^{\lambda_{0,i}} g^{f(1)\lambda_{1,i}} \cdots g^{f(t-1)\lambda_{t-1,i}} \\ &= g_1^{\lambda_{0,i}} g^{\sum_{k=1}^{t-1} f(k)\lambda_{k,i}} = g^{f(i)}. \end{aligned}$$

Once $\mathcal{C}$ has computed all the verification keys vk_i, it outputs them to $\mathcal{A}$.

Signature Share Query: Once the adversary $\mathcal{A}$ has the verification keys it provides up to q_S signature share generation queries to the `TBSign` oracle according to the experiment in Definition 5. The oracle queries are processed by $\mathcal{C}$ that has to output a signature share $\sigma_i = (Y_{i1}, Y_{i2})$ on input $(X_i, A_{k,j}, B_{k,i}, \hat{B}_{k,i}, \pi_i^{(1)}, \pi_i^{(2)})$, for $i = \{1, \ldots, n\}, j = \{1, \ldots, \ell\}, k = \{1, 2, 3\}$. The proofs $\pi_i^{(1)}$ and $\pi_i^{(2)}$ ensure that $X_i = \left(u_1' \prod_{j=1}^{\ell} u_{1,j}^{\mu_j}\right)^{r_i}$ due to their soundness property as proven in [35,37]. We note that the perfect binding property of the commitment scheme guarantees the soundness of the NIZK proof $(\pi_i^{(1)}, \pi_i^{(2)})$. Since our commitment scheme in Sect. 4.1 contains perfect binding keys it provides the existence of an extraction key which allows extraction of the values r_i

and $\mu_1, \ldots, \mu_\ell$. For more details on the proof we refer to [35,37]. The challenger $\mathcal{C}$ extracts $r_i, \mu_1, \ldots, \mu_\ell$ and prepares a signature on these values using the CDH challenge. $\mathcal{C}$ first defines two functions for $m = (\mu_1, \ldots, \mu_\ell)$:

$$F(m) = (q - tl) + a' + \sum_{i=1}^{\ell} a_i \mu_i \text{ and } G(m) = b' + \sum_{i=1}^{\ell} b_i \mu_i.$$

Additionally we define the following binary function:

$$K(m) = \begin{cases} 0, & \text{if } a' + \sum_{j=1}^{\ell} a_j \mu_j \equiv 0 \mod l \\ 1, & \text{otherwise.} \end{cases}$$

Upon the computation of $u' \prod_{j=1}^{\ell} u_j^{\mu_j} = g_1^{q-tl+a'} g^{b'} \prod_{j=1}^{\ell} g_1^{a_j \mu_j} g^{b_j \mu_j}$ and using $F(m)$ and $G(m)$ the challenger has to return signature shares $\sigma_i = (Y_{i1}, Y_{i2})$. These are computed by $\mathcal{C}$ using the Lagrange coefficients $\lambda_{0,i}, \lambda_{1,i}, \ldots, \lambda_{t-1,i} \in \mathbb{Z}_q$ such that $f(i) = \lambda_{0,i} f(0) + \sum_{k=1}^{t-1} \lambda_{k,i} f(k)$ using the technique from Boneh and Boyen [12]. $\mathcal{C}$ then picks $r_i' \in \mathbb{Z}_q$, and outputs the following signature tuple $\sigma_i = (Y_{i,1}, Y_{i,2})$, where $Y_{i,1} = g^{b\left(\lambda_{0,i} f(0) + \sum_{k=1}^{t-1} \lambda_{k,i} f(k)\right)} g^{aF(m) r_i'} g^{G(m) r_i'}$ and $Y_{i,2} = g^{r_i'}$, where $r_i' = r_i - \frac{b\lambda_{0,i}}{F(m)}$. Note that $f(0) = -\frac{G(m)}{F(m)}$. The signature σ_i satisfies the verification equation $e(Y_{i1}, g) = e(g_2, vk_i) e(u' \prod_{j=1}^{\ell} u_j^{\mu_j}, Y_{i2})$ since

$$\begin{aligned}
& e(g_2, vk_i) e(u' \prod_{j=1}^{\ell} u_j^{\mu_j}, Y_{i,2}) \\
&= e\left(g^b, g^{\lambda_{0,i} f(0) + \sum_{k=1}^{t-1} \lambda_{k,i} f(k)}\right) e\left(g^{aF(m)} g^{G(m)}, g^{r_i'}\right) \\
&= e\left(g, g^{b\left(\lambda_{0,i} f(0) + \sum_{k=1}^{t-1} \lambda_{k,i} f(k)\right)}\right) e\left(g^{aF(m) r_i'} g^{G(m) r_i'}, g\right) \\
&= e\left(g^{b\left(\lambda_{0,i} f(0) + \sum_{k=1}^{t-1} \lambda_{k,i} f(k)\right)} g^{aF(m)\left(r_i - \frac{b\lambda_{0,i}}{F(m)}\right)} g^{G(m)\left(r_i - \frac{b\lambda_{0,i}}{F(m)}\right)}, g\right) \\
&= e(Y_{i,1}, g).
\end{aligned}$$

In order to complete the simulation without aborting, it is required that all signature queries on m have $K(m) \neq l$. In this case, if $F(m) \neq 0$ then $\mathcal{C}$ is able to simulate the signature on the requested m; otherwise, $\mathcal{C}$ will not be able to generate such signature and the simulation aborts.

Extraction: The execution of this step corresponds to the fourth step from the experiment in Definition 5, where $\mathcal{A}$ sets $\sigma_i^* = (\sigma_{i1}^*, \sigma_{i2}^*)$ for $i \in \{1, q_S\}$ as a valid signature share for a message $m^* = (\mu_1^*, \ldots, \mu_\ell^*)$, which was not queried before. As next, we define a function $Q(m^*, \boldsymbol{q}, A')$, where $A' = (a', a_1, \ldots, a_\ell)$ are the simulated values, and $\boldsymbol{q} = (q_1, \ldots, q_S)$ as

$$Q(m^*, \boldsymbol{q}, A') = \begin{cases} 0, & \text{if } \forall_{j=1}^{s} q_j : K_{q_j}(m_j) = 1, \text{ and } a' + \sum_{j=1}^{\ell} a_j \mu_j^* \equiv 0 \mod l \\ 1, & \text{otherwise.} \end{cases}$$

The function evaluates to 0 if all signature queries will not cause an abort for a given choice of values A' and the function $a' + \sum_{j=1}^{\ell} a_j \mu_j^* \mod l$ which equals $F(m^*) \mod l$ vanishes for the values $m^* = (\mu_1^*, \ldots, \mu_\ell^*)$. That means if $F(m^*) = 0$ then $\mathcal{C}$ can extract g^{ab} by computing

$$g^{ab} = \left(\frac{\sigma_{i1}}{\sigma_{i2}^{G(m)d_i} \, (g^b)^{\sum_{k=1}^{t-1} \lambda_{k,i} f(k)}} \right)^{-\frac{1}{d_i \lambda_{0,i}}}.$$

Therefore we can consider the probability over the simulation values $(\mu_1^*, \ldots, \mu_\ell^*)$ as If $F(m^*)$ is not 0, then we have $Q(m^*, \boldsymbol{q}, A') = 1$ and the extraction aborts with probability $Pr[Q(m^*, \boldsymbol{q}, A') = 1]$. $\mathcal{C}$ repeats the above showed steps q_S times. If all q_S rounds are completed, $\mathcal{A}$ outputs at least $q_S + 1$ valid signatures with different messages, where at least one valid message-signature pair is different from the q_S valid messages-signatures given from $\mathcal{C}$ algorithm.

Analysis: The probability of success of an adversary $\mathcal{A}$ can be compared with the probability that $\mathcal{C}$ aborts in the simulation, which happens either if $F(m_i) = 0$ for a signature query on m_i or if $F(m^*) \neq 0$. The probability for $F(m^*) \neq 0$ can be bounded using following lemma.

Lemma 1 ([38]). *Let $X, Y_1, \ldots, Y_q \subseteq [l]$ such that holds $|X|, |Y_i| \geq d$ and $|(X \setminus Y_i) \cup (Y_i \setminus X)| \geq d$ for some $d \geq 1$ and all i. Then, we have*

$$Pr\left[a(X) = 0 \wedge \forall i \in [q] : a(Y_i) \neq 0\right] \geq \left(1 - C \cdot q \cdot \frac{\sqrt{\ell}}{d \cdot \sqrt{w}}\right) \cdot \frac{D}{\sqrt{d \cdot w}}$$

for $a(X) = \sum_{i \in X} x_i$ and for fixed constants C, D that do not depend on values ℓ, w, d, q, X and the Y_i

To apply this lemma to our analysis we set $X := m^*$ and $Y_i = m_i$ for $i \in [1, \ldots, q_S]$, $a(X) = F(m^*)$ and $a(Y_i) = K_{q_j}(m_j)$. ℓ denotes the bit-length of the message and w equals in our scheme to the length of the vector $\mathbf{a}$, such that $w = \ell$. It also holds that $|\,(X \setminus Y_i) \cup (Y_i \setminus X)\,| \geq$. We consider that *abort* denotes the event that the simulation fails. This happens either because $F(m_i) = 0$ for a signature query on m_i, or because $F(m^*) \neq 0$. The lemma above provides an upper bound of $1 - \Theta(1/q)$. We conclude that $Pr\left[a(X) = 0 \wedge \forall i \in [q] : a(Y_i) \neq 0\right]$ corresponds to $Pr[\overline{abort}]$. The proof in [38] for Lemma 1 showed that the upper bound can be estimated by $\frac{D\sqrt{\chi}}{4C} \frac{1}{q_S}$, where $\chi = d/\ell$. Since χ is a constant, then the probability $P[\overline{abort}]$ has a lower bound of $\Theta(1/q)$. That means that the lower bound of the probability $Pr[\overline{abort}]$ is $\epsilon(1/q)$. This completes the proof of unforgeability of our TBS scheme. □

Theorem 2 (Blindness). *Our TBS scheme is blind in the common reference string model assuming the hardness of the DLin assumption from Definition 2, the perfect hiding property and the zero-knowledge property of the NIZK proof from [35, 37].*

Proof. The full proof of blindness is given in Appendix B.

5 Applications

Having introduced a new TBS construction, we highlight now its application to distributed e-voting and to distributed e-cash systems.

Distributed Verifiable E-voting. We recall first the general concept of an e-voting scheme and highlight functionalities of its algorithms based on [20]: a *voter* V is a party that is authorized by a *voting authority* to submit votes. The *tallying authority* collects individual votes and tally the results of the election to obtain the outcome. A *public board* which can be considered as a broadcast channel makes its content public to all parties and each party can add information to the board but not remove or modify any of the published contents. This board is typically used for the purpose of universal verifiability [25] of the e-voting process.

Koenig et al. [44] presented a generic template for e-voting protocols with distributed voting authorities assuming existence of threshold blind signatures, yet without offering concrete constructions of the latter schemes. Their generic e-voting protocol was shown to satisfy the security properties from [45]. By using our rerandomizable threshold blind signature scheme we therefore enable the actual construction of such distributed e-voting scheme.

The resulting scheme proceeds as follows. Each voting authority A_i, $i = 1, \ldots, n$ is in possession of the secret share sk_i from our TBS construction, and the corresponding public key pk is assumed to be published on the public board. Each voter encrypts its vote using the public key of the tallying authority and executes the `TBSign` protocol with each of n voting authorities to obtain a threshold blind signature σ on the encrypted vote if at least t out of n authorities provide their shares. It combines all signature share to a common threshold blind signature σ. This signature and the encrypted vote are sent to public board. The tallying authority decrypts each vote and publishes its content on the board together with the corresponding proof of decryption. The votes can then be counted and verified publicly.

In general, an e-voting scheme is required to provide the following properties that we recall informally here. More formal definitions can be found in [8,31, 45] The first important property is *privacy*, which means that individual votes remain hidden. The *soundness* property prevents dishonest voters from biasing the voting process. Finally, the *public verifiability* property ensures that anyone can check that the votes has been counted to prevent potential falsifications of counting process.

We briefly discuss why our construction offers privacy, soundness and public verifiability. Privacy follows from the fact, that the e-voting scheme is based on our TBS scheme. This guarantees in case that all authorities collude against a voter, the voter's privacy remains preserved due to the blindness property of the TBS scheme. Soundness is satisfied because of the TBS unforgeability. Public verifiability is satisfied because the decrypted votes, corresponding ciphertext as well as proofs for correct decryption of votes are published on the public board.

Distributed E-cash. We recall the basic functionality of an e-cash scheme [23]. The parties involved are the banks, users and merchants. Any user can *withdraw* an e-coin from her account at the bank and then *spend* it at some merchant. The merchant can then *deposit* the received coin on its account in the bank. In a distributed e-cash the role of the bank is split amongst n currency issuers, who are involved in the process of coin generation. This distributed approach helps to mitigate the threat of dishonest banks as suggested by Miers et al. [49].

Our TBS scheme can be used as a building block for a distributed e-cash scheme as discussed in the following. Upon withdrawal the user requests a coin by choosing a random unique coin identifier r and by executing the `TBSign` protocol over a secure channel with n issuers, each in possession of a secret key share sk_i. After obtaining at least t valid signature shares on r the user can compute the blind signature σ which resembles the coin. The coin spending protocol is performed over a secure channel through which the user sends its coin (σ, r) to the merchant, who in turn can check the validity of the coin by executing the `TBVerify` algorithm. If the coin is valid, the merchant establishes a secure connection with the bank aiming to deposit it on its account. In order to avoid double-spending the bank must check that no coins with identifier r were previously spent using the coin database that is maintained by the bank. If the coin passes this check then the bank deposits it on the merchant's account.

A (distributed) e-cash scheme is supposed to fulfill the following three common properties [16,18]. The *anonymity* property means that even if $t-1$ dishonest issuers conspire with malicious merchants, the coin withdrawal and spending phases performed by the user should remain unlinkable. The *balance* property prevents coalitions of malicious users and merchants from depositing more coins than were originally withdrawn. The *(ex)culpability* property implies that any dishonest user who is willing to spend one coin twice is caught and that no coalition of at most $t-1$ malicious issuers with merchants is able to accuse an honest user of double-spending.

We discuss briefly the security of the above approach. The anonymity property follows from the blindness of TBS signatures, which guarantees that spending of a coin (r, σ) cannot be linked to the corresponding withdrawal phase. The balance property follows from unforgeability of TBS signatures and the requirement on the bank to check that coin identifiers r do not repeat. If r does not repeat and more coins were deposited than issued then at least of those coins would resemble a TBS forgery. The (ex)culpability property does not rely on the security of the TBS scheme and follows from the authentication property of secure channels between the user and the issuers upon withdrawal and between the user and the merchant upon spending. More precisely, from the authentication requirement on such channels. In order to accuse an honest user of double spending malicious issuers and merchants would need to come up with a transcript of the spending protocol authenticated by the user that shows the attempt to spend the same coin (r, σ) twice. Similarly, in order to catch a dishonest user who double-spends a coin honest issuers and merchants would be able to present two transcripts of the spending protocol authenticated by this user.

6 Conclusion

We proposed the first standard-model construction of (re-randomizable) threshold blind signatures (TBS), where signatures can be obtained in a blind way through interaction with n signers of which t are required to provide their signature shares. The stronger security notions for TBS schemes formalized in our work extend the definitions from [54] to the threshold setting. We further showed how our TBS construction can be used to realize a distributed e-voting protocol following the template from [44] that guarantees privacy, soundness and public verifiability in presence of distributed voting authorities. As a second application we discussed construction of a distributed e-cash scheme, which achieves the desirable properties of anonymity, balance, and (ex)culpability, and where n issuers are involved in the generation of coins, a measure suggested in [49] to address the trust problem in non-distributed e-cash scenarios.

A Blind Signature Scheme by Okamoto [51]

Our construction is influenced by the techniques underlying the following blind signature scheme from [51].

$\texttt{BParGen}(1^\lambda)$: Generate the public bilinear group parameters $I = (\mathbb{G}, \mathbb{G}_T, q, g, e)$.
$\texttt{KGen}(I)$: Pick $x \xleftarrow{r} \mathbb{Z}_q^*$ and generators $g_2, u', u_1, \ldots, u_n \xleftarrow{r} \mathbb{G}$ and set $g_1 \leftarrow g^x$. Output $pk = (g, g_1, g_2, u', u_1, \ldots, u_n)$ and $sk = g_2^x$.
$\texttt{BSign}(\cdot)$: Let $m \in \{0,1\}^n$ be a message and μ_i the i-th bit of m. User U selects $r \xleftarrow{r} \mathbb{Z}_p^*$ and computes $X \leftarrow \left(u' \prod_{i=1}^{n} u_i^{\mu_i}\right)^r$ and sends X to the signer S. U additionally provides to S that it knows $(r, \mu_1, \ldots, \mu_n)$ with $\mu_i \in \{0,1\}$ for X using the following witness indistinguishable Σ protocol:

- U selects $\delta_1, \ldots, \delta_n \xleftarrow{r} \mathbb{Z}_p^*$, computes $M_i = u_i^{\mu_i}(u')^{\delta_i}$, $(i = 1, \ldots, n)$ and sends $(M_1, \ldots, M_n)$ to S.
- U proves to S that U knows δ_i such that $M_i = (u')^{\delta_i}$ for $\mu_i = 0$ or $M_i = u_i(u')^{\delta_i}$ for $\mu_i = 1$, where $i \in [1, n]$. This proof can be realized by a Σ protocol which was described in [5].
- U proves to S that U knows $(t, \beta, \gamma_1, \ldots, \gamma_n)$ such that $X = \left(\prod_{i=1}^{n} M_i\right)^t \cdot (u')^\beta$, and $X = (u')^t \prod_{i=1}^{n} u_i^{\gamma_i}$, where $\beta \leftarrow t - t(\sum_{i=1}^{n} \delta_i) \mod p$ and $\gamma_i \leftarrow t\mu_i$.

If S accepts in the above protocol then it selects $d \xleftarrow{r} \mathbb{Z}_p^*$, computes $Y_1 \leftarrow g_2^x X^d$, $Y_2 \leftarrow g^d$, and sends (Y_1, Y_2) to U. U eventually selects $s \xleftarrow{r} \mathbb{Z}_p^*$ and computes a blind signature $\sigma = (\sigma_1, \sigma_2)$, where

$$\sigma_1 \leftarrow Y_1 \left(u' \prod_{i=1}^{n} u_i^{\mu_i}\right)^s \quad \text{and} \quad \sigma_2 \leftarrow Y_2^r g^s.$$

$\texttt{BVerify}(pk, m, \sigma)$: Parse pk as $(g, g_1, g_2, u', u_1, \ldots, u_n)$ and σ as (σ_1, σ_2). If $e(\sigma_1, g) = e(g_1, g_2)e\left(\sigma_2, u' \prod_{i=1}^{n} u_i^{\mu_i}\right)^{-1}$ output 1; otherwise output 0.

The unforgeability and blindness of the scheme were proven in [51] based on the unforgeability of the Waters scheme [58] and the security of "OR" proofs [24].

B Proof of Theorem 2 (Blindness)

Proof. We assume that the proposed signature scheme is not blind. That means the existence of a dishonest signer $\mathcal{S}^*$, which can guess b correctly with a non-negligible advantage $1/2 + \epsilon$. We construct an algorithm $\mathcal{C}$ which can break the security of the DLIN assumption as follows. Given the public parameters $pp = (\mathbb{G}, \mathbb{G}_T, q, e, g)$, the DLIN problem instance $(g^a, g^b, g^c) = (u_1', u_2', u_3')$ the challenger $\mathcal{C}$ computes $(u_{1,j}, u_{2,j}, u_{3,j}) = \left(\left(u_1'\right)^{\xi_{1,j}}, \left(u_2'\right)^{\xi_{2,j}}, \left(u_3'\right)^{\xi_{3,j}}\right)$, with $\xi_{1,j}, \xi_{2,j}, \xi_{3,j} \in \mathbb{Z}_q^*$, and $\xi_{3,j} = \xi_{1,j} + \xi_{2,j}$, $j \in \{1, \ldots, \ell\}$. $\mathcal{C}$ gives $(pp, pk, u_1', u_2', u_3', u_{1,j}, u_{2,j}, u_{3,j})$ to $\mathcal{S}^*$ as CRS. $\mathcal{S}^*$ gives $\mathcal{C}$ a public key $pk = (g_1, g_2, \mathbf{vk})$ and two messages $m_0, m_1 \in \mathbb{Z}_q^*$. The challenger $\mathcal{C}$ checks if $pk \in \mathbb{G}$ and $m_0, m_1 \in \mathbb{Z}_q^*$. If it holds $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$. $\mathcal{C}$ chooses $r_i \in \mathbb{Z}_q^*$ and computes $X_{i,0} = (u_1' \prod_{j=1}^{\ell} u_{1,j}^{\mu_{j,0}})^{r_i}$ and $X_{i,1} = (u_1' \prod_{j=1}^{\ell} u_{1,j}^{\mu_{j,1}})^{r_i}$ for $m_b = (\mu_{1,b}, \ldots, \mu_{\ell,b}), b \in \{0, 1\}$. $\mathcal{C}$ executes the both NIZK protocols from Sect. 4.1 to prove $\mathcal{S}^*$ that $\mathcal{C}$ knows $(r_i, \mu_{1,b}, \ldots, \mu_{\ell,b})$ for both messages $m_b = \{m_0, m_1\}$. From the proofs in [35,37] follows that for $u_{3,j} = \left(u_3'\right)^{\xi_{1,j}+\xi_{2,j}}$ the commitments are perfect hiding and the two parameter initializations are indistinguishable under the DLIN assumption. Therefore the commitments on the messages m_b and m_{1-b} leak no information about the message. The perfect hiding property of commitments allows to simulate NIZK proofs $(\pi_{i,0}^{(1)}, \pi_{i,0}^{(2)})$ and $(\pi_{i,1}^{(1)}, \pi_{i,1}^{(2)})$, that remain indistinguishable from real proofs as shown in Sect. 4.4, [35]. $\mathcal{C}$ outputs $X_{i,b}X_{i,1-b}$ and the simulated NIZK proofs $(\pi_{i,b}^{(1)}, \pi_{i,b}^{(2)})$ and $(\pi_{i,1-b}^{(1)}, \pi_{i,1-b}^{(2)})$, where $\pi_{i,b}^{(1)}$ is the first part of NIZK proof, which is built to the commitment $X_{i,b}$ and $\pi_{i,b}^{(2)}$ is the corresponding second part of NIZK proof to the commitment $X_{i,b}$. Analogously are defined the proofs $(\pi_{i,1-b}^{(1)}, \pi_{i,1-b}^{(2)})$. After completing the NIZK protocol the challenger $\mathcal{C}$ acts as a honest user and proceeds in the same manner as the real one. $\mathcal{C}$ sends his outputs to the dishonest signer $\mathcal{S}^*$. The challenger $\mathcal{C}$ executes the signing process first on behalf of $\mathcal{U}_b$ on input $(pk, X_{i,b}, \pi_{i,b}^{(1)}, \pi_{i,b}^{(2)})$ and then on behalf of $\mathcal{U}_{1-b}$ on input $pk, X_{i,1-b}, \pi_{i,1-b}^{(1)}, \pi_{i,1-b}^{(2)})$. Since the commitments and the proofs do not leak any information about the message, the output $\sigma_{i,b}$ of the signing protocol on behalf of $\mathcal{U}_b$ is indistinguishable from the output $\sigma_{i,1-b}$ of the protocol on behalf of $\mathcal{U}_{1-b}$. If $\mathcal{S}^*$ rejects to sign one of the inputs $(X_{i,b}, \pi_{i,b}^{(2)})$ or $(X_{i,1-b}, \pi_{i,1-b}^{(2)})$, then for the corresponding output holds $\sigma_b = \perp$ or $\sigma_{1-b} = \perp$. This means that the both resulting signatures are set to $\perp$, and $\mathcal{S}^*$,

does not gain any advantage if he would try to hinder the game execution. Otherwise, after finishing the signing phase of the blind signature for $\mathcal{U}_b$ and $\mathcal{U}_{1-b}$, $\mathcal{C}$ checks the validity of the obtained signatures for $\mathcal{U}_0$, $\mathcal{U}_1$ by computing the follows $e(Y_{i,b,1}, g) = e(g_2, vk_i)e(X_{i,b}, Y_{i,b,2})$. If both of the signatures $\sigma_{i,b}, \sigma_{i,1-b}$ are valid, $\mathcal{C}$ gives them to $\mathcal{S}^*$. If only one of them is valid, $\mathcal{C}$ outputs $\perp$. $\mathcal{C}$ obtains then the output b' of $\mathcal{S}^*$. If $b = b'$, $\mathcal{C}$ outputs $\beta \leftarrow 0$, otherwise it outputs $\beta \leftarrow 1$.

Analysis: Observe that if $b = b'$ then $(u_{1,j}, u_{2,j}, u_{3,j})$ for $j = \{1, \ldots, \ell\}$ are DLIN tuples with $(u_{1,j}, u_{2,j}, u_{3,j}) = \left(\left(u_1^{'}\right)^{\xi_{1,j}}, \left(u_2^{'}\right)^{\xi_{2,j}}, \left(u_3^{'}\right)^{\xi_{3,j}}\right)$, with $\xi_{3,j} = \xi_{1,j} + \xi_{2,j}$ and $(u_1^{'}, u_2^{'}, u_3^{'}) = (g^a, g^b, b^c)$. In this case the challenger outputs $b_{DLIN} = 1$ and σ_b, σ_{1-b} are perfectly simulated. Therefore $Pr[b_{DLIN} = 1 | b = b'] = 1/2$ Whether the challenger $\mathcal{C}$ outputs $\perp$ or two valid signatures σ_0, σ_1 depends only the adversary's reply, i.e. whether its reply σ_i satisfies the verification process or not. Therefore it is completely independent from b, since the distribution of X_0 and X_1 are indistinguishable from each other. Hence $Pr[b_{DLIN} = 0 | b \neq b'] = 1/2 + \epsilon$. Eventually it follows that the success probability in DLIN problem is $1/2(1/2) + 1/2(1/2 + \epsilon) = 1/2 + \epsilon/2$, which contradicts the DLIN assumption, for negligible ϵ. □

References

1. Abe, M.: A secure three-move blind signature scheme for polynomially many signatures. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 136–151. Springer, Heidelberg (2001)
2. Abe, M., Fehr, S.: Adaptively secure Feldman VSS and applications to universally-composable threshold cryptography. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 317–334. Springer, Heidelberg (2004)
3. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
4. Abe, M., Ohkubo, M.: A framework for universally composable non-committing blind signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 435–450. Springer, Heidelberg (2009)
5. Abe, M., Okamoto, T.: Provably secure partially blind signatures. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 271–286. Springer, Heidelberg (2000)
6. Baudron, O., Fouque, P., Pointcheval, D., Stern, J., Poupard, G.: Practical multi-candidate election system. In: Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, PODC 2001, pp. 274–283. ACM (2001)
7. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. In: Syverson, P.F. (ed.) FC 2001. LNCS, vol. 2339, pp. 309–328. Springer, Heidelberg (2002)
8. Benaloh, J.C., Tuinstra, D.: Receipt-free secret-ballot elections (extended abstract). In: Proceedings of the 26th Annual ACM Symposium on Theory of Computing, pp. 544–553. ACM (1994)

9. Blazy, O., Fuchsbauer, G., Pointcheval, D., Vergnaud, D.: Signatures on randomizable ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 403–422. Springer, Heidelberg (2011)
10. Blazy, O., Fuchsbauer, G., Pointcheval, D., Vergnaud, D.: Short blind signatures. J. Comput. Secur. **21**(5), 627–661 (2013)
11. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2002)
12. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
13. Brands, S.: Untraceable off-line cash in wallets with observers. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 302–318. Springer, Heidelberg (1994)
14. Brands, S.A.: An efficient off-line electronic cash system based on the representation problem. Technical report, Amsterdam, The Netherlands (1993)
15. Camenisch, J., Groß, T.: Efficient attributes for anonymous credentials. In: Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, pp. 345–356. ACM (2008)
16. Camenisch, J.L., Hohenberger, S., Lysyanskaya, A.: Compact e-Cash. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 302–321. Springer, Heidelberg (2005)
17. Camenisch, J.L., Koprowski, M., Warinschi, B.: Efficient blind signatures without random oracles. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 134–148. Springer, Heidelberg (2005)
18. Camenisch, J., Lysyanskaya, A., Meyerovich, M.: Endorsed e-cash. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007), pp. 101–115. IEEE Computer Society (2007)
19. Camenisch, J.L., Neven, G., Shelat, A.: Simulatable adaptive oblivious transfer. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 573–590. Springer, Heidelberg (2007)
20. Cetinkaya, O., Cetinkaya, D.: Verification and validation issues in electronic voting. Electron. J. e-Government **5**, 117–126 (2007)
21. Chaum, D.: Blind signatures for untraceable payments. CRYPTO 1982, pp. 199–203. Springer, Heidelberg (1982)
22. Chaum, D.: Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 177–182. Springer, Heidelberg (1988)
23. Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 319–327. Springer, Heidelberg (1990)
24. Cramer, R., Damgård, I.B., Schoenmakers, B.: Proof of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
25. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multiauthority election scheme. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 103–118. Springer, Heidelberg (1997)
26. Desmedt, Y.G.: Society and group oriented cryptography: a new concept. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 120–127. Springer, Heidelberg (1988)
27. Desmedt, Y.G., Frankel, Y.: Shared generation of authenticators and signatures. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 457–469. Springer, Heidelberg (1992)

28. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: 28th Annual Symposium on Foundations of Computer Science, pp. 427–437. IEEE Computer Society (1987)
29. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 60–77. Springer, Heidelberg (2006)
30. Franklin, M., Yung, M.: Towards provably secure efficient electronic cash. Technical report TR CUSC-018-92, Columbia University, Department of Computer Science (1993). Also in: Lingas, A., Carlsson, S., Karlsson, R. (eds.): ICALP 1993. LNCS, vol. 700. Springer, Heidelberg (1993)
31. Fujioka, A., Okamoto, T., Ohta, K.: A practical secret voting scheme for large scale elections. In: Zheng, Y., Seberry, J. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 244–251. Springer, Heidelberg (1993)
32. Garg, S., Rao, V., Sahai, A., Schröder, D., Unruh, D.: Round optimal blind signatures. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 630–648. Springer, Heidelberg (2011)
33. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Robust threshold DSS signatures. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 354–371. Springer, Heidelberg (1996)
34. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure distributed key generation for discrete-log based cryptosystems. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 295–310. Springer, Heidelberg (1999)
35. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (2006)
36. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 321–340. Springer, Heidelberg (2010)
37. Groth, J., Ostrovsky, R., Sahai, A.: Non-interactive Zaps and New Techniques for NIZK. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 97–111. Springer, Heidelberg (2006)
38. Hofheinz, D., Jager, T., Knapp, E.: Waters Signatures with Optimal Security Reduction. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 66–83. Springer, Heidelberg (2012)
39. Horvitz, O., Katz, J.: Universally-composable two-party computation in two rounds. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 111–129. Springer, Heidelberg (2007)
40. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, WPES 2005, pp. 61–70. ACM (2005)
41. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures. In: Kaliski Jr, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 150–164. Springer, Heidelberg (1997)
42. Kiayias, A., Zhou, H.-S.: Equivocal blind signatures and adaptive UC- security. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 340–355. Springer, Heidelberg (2008)
43. Kim, J.-H., Kim, K., Lee, C.S.: An efficient and provably secure threshold blind signature. In: Kim, K. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 318–327. Springer, Heidelberg (2002)

44. Koenig, R.E., Dubuis, Haenni, R.: Why public registration boards are required in e-voting systems based on threshold blind signature protocols. In: Electronic Voting 2010, EVOTE 2010, 4th International Conference, Co-organized by Council of Europe, Gesellschaft für Informatik and E-Voting.CC, vol. 167 LNI, pp. 255–266. GI (2010)
45. Lee, B., Kim, K.: Receipt-free electronic voting scheme through collaborationf of voter and honest verifier. In: Proceeding of JW-ISC 2000, pp. 101–108 (2000)
46. Li, J., Yuen, T.H., Kim, K.: Practical threshold signatures without random oracles. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 198–207. Springer, Heidelberg (2007)
47. Lysyanskaya, A., Peikert, C.: Adaptive security in the threshold setting: from cryptosystems to signature schemes. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 331–350. Springer, Heidelberg (2001)
48. Meiklejohn, S., Shacham, H., Freeman, D.M.: Limitations on transformations from composite-order to prime-order groups: the case of round-optimal blind signatures. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 519–538. Springer, Heidelberg (2010)
49. Miers, I., Garman, C., Green, M., Rubin, A.D. : Zerocoin: Anonymous distributed e-cash from bitcoin. In: 2013 IEEE Symposium on Security and Privacy, SP 2013, pp. 397–411. IEEE Computer Society (2013)
50. Okamoto, T.: An electronic voting scheme. In: Terashima, N., Altman, E. (eds.) Advanced IT Tools. IFIP, pp. 21–30. Springer, Heidelberg (1996)
51. Okamoto, T.: Efficient blind and partially blind signatures without random oracles. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 80–99. Springer, Heidelberg (2006)
52. Pointcheval, D., Stern, J.: Provably secure blind signature schemes. In: Kim, K., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 252–265. Springer, Heidelberg (1996)
53. Pointcheval, D., Stern, J.: New blind signatures equivalent to factorization (extended abstract). In: Proceedings of the 4th ACM Conference on Computer and Communications Security CCS 1997, pp. 92–99. ACM (1997)
54. Schröder, D., Unruh, D.: Security of blind signatures revisited. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 662–679. Springer, Heidelberg (2012)
55. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979)
56. Shoup, V.: Practical threshold signatures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 207–220. Springer, Heidelberg (2000)
57. Vo, D.L., Zhang, F., Kim, K.: A new threshold blind signature scheme from pairings (2003)
58. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
59. Zhou, X.:Threshold cryptosystem based fair off-line e-cash. In: Proceedings on the 2nd International Symposium on Intelligent Information Technology, pp. 692–696 (2008)