k-times Full Traceable Ring Signature*

Xavier Bultel
Université Clermont Auvergne
LIMOS, BP 10448, 63000 Clermont-Ferrand, France
Email: xavier.bultel@udamail.fr

Pascal Lafourcade
Université Clermont Auvergne
LIMOS, BP 10448, 63000 Clermont-Ferrand, France
Email: pascal.lafourcade@udamail.fr

***Abstract*—Ring and group signatures allow their members to anonymously sign documents in the name of the group. In ring signatures, members manage the group themselves in an ad-hoc manner while in group signatures, a manager is required. Moreover, k-times traceable group and ring signatures [3] allow anyone to publicly trace two signatures from a same user if he exceeds the *a priori* authorized number of signatures. In [8], Canard *et al.* give a 1-time traceable ring signature where each member can only generate one anonymous signature. Hence, it is possible to trace any two signatures from the same user. Some other works generalize it to the k-times case, but the traceability only concerns two signatures. In this paper, we define the notion of *k-times full traceable ring signature* (k-FTRS) such that all signatures produced by the same user are traceable if and only if he produces more than k signatures. We construct a k-FTRS called Ktrace. We extend existing formal security models of k-times linkable signatures to prove the security of Ktrace in the random oracle model. Our primitive k-FTRS can be used to construct a k-times veto scheme or a proxy e-voting. The main advantage of our scheme is that it prevents denial-of-service caused by cheating users.**

I. Introduction

Organization of elections is an old concern of Humanity. Therefore many kinds of elections have been invented. With the development of Internet and the progresses in cryptography, several e-voting schemes have been developed, see [16] for a survey. In all these systems the question of proxy voting exists independently of the form of the ballots and the counting phase. The problem is how a voting system can allow someone who cannot participate to the election to delegate his vote to someone else. In some elections, with the agreement of the voter, another voter can vote twice or more. Each voter can have different numbers of proxy-votes from 0 up to a limit fixed by the election rules. In traditional paper elections, using paper ballots and a transparent box, a cheater who voted with an extra false proxy vote, is *a posteriori* detected during the tally phase, hence the election is canceled. An electronic voting system should offer a proxy mechanism that is better than in a traditional system. Once a fraud is detected then the cheater should be identified in order to blame him. Moreover, anyone should be able to find all the ballots that he has introduced in the system in order to remove them from the final count. Such a mechanism would prevent having to reorganize the election.

* This research was conducted with the support of the "Digital Trust" Chair from the University of Auvergne Foundation.

Another close problem is the elaboration of the program committee of a conference, that is often done by the members of the steering committee. Usually each member of the steering committee anonymously proposes a list of names for the program committee. Some members might want to discard some names for personal reasons, but they do not want other members of the steering committee to learn their choices. Our aim is to construct a system that allows each member to anonymously express a maximum of k vetos. If one person gives more than k vetos then we want to be able to detect it and to learn all vetos from the cheater in order to remove them. Using existing signature schemes such as [3], it is possible to design such a veto system and, in case someone uses more than his number of vetos, the cheater is identified. Unfortunately the cheater's vetos remain anonymous, hence it is not possible to discard them. While with our ring signature scheme, all cheater's vetos can be discarded without restarting the elaboration of the program committee.

For these two applications, we design a ring signature that offers the following security properties:

- A new signature cannot be forged without knowing a secret signature key.
- Each group signer i can have his own number of authorized signatures k_i which is determined when the public key is set up.
- If a signer i signs a number of messages lower or equal to his authorized number of signatures k_i, it is not possible to determine which member of the group has signed the message. Moreover, no one can determine if two signatures have been generated with the same secret signature key (*anonymity*).
- If a signer i signs more than his authorized number of signatures k_i then anybody can link all his signatures (*linkability*) and also determine the identity of the cheater (*traceability*).
- Linkability and traceability properties exist only if the signatures are generated for the same event: for instance the first and the second round of an election are two different events.

In this paper, we propose a k-times full traceable ring signature that has all these features.

a) Contributions: We define a cryptographic primitive called *k-times full traceable ring signature*. This primitive allows everyone to publicly trace all signatures of a group member who has produced more than his personal threshold number of signatures k during a specified event. Existing results in the literature only allow to link or open two of these signatures. We give a formal definition of our primitive and its security properties. More precisely, we have full anonymity of the signer if he does not over pass his threshold value. Moreover the ring signature verification key is only composed of the public key of each member, so it is easy to construct by an *ad-hoc* group of users with different thresholds for each user. Finally, the produced signatures are linear in the group size times the maxium of thresholds. We also give proofs of this scheme in the random oracle model (ROM).

Our k-times full traceable ring signature primitive can be used to design an anonymous veto mechanism for the steering committee where vetos are anonymous signatures on program committe names that are not desired. This mechanism can detect members that use more vetos than expected and remove cheater's vetos while preserving the anonymity of other users. It can also be used to organize elections with multiple proxy votes such that when a user signs more ballots that he should, everybody can open all his signatures.

b) Related Works: Group and ring signatures are two well known cryptographic primitives that first appeared respectively in [11] and [20]. Both primitives allow users to anonymously sign a digital document within a group. In group signatures, a special authority manages the group using a manager secret key, while in ring signatures, members manage the group themselves in an *ad-hoc* manner. In [5], the authors present a scheme where the size of a signature is constant, *i.e.* it does not depend to the group size. Some group/ring signature schemes deal with *linkability*. Linkable ring signatures are first defined in [17]. In this paper, the authors present a ring signature that allows users to publicly link two signatures produced by the same user within the group. The security model of this primitive are formalized in [18]. Although the signature size grows linearly with the group size in most schemes, there exists constant size signatures such as [24] and [1]. The main applications of linkable signature are e-voting and e-cash, but this property is also used in several other applications, such as the direct anonymous attestation scheme described in [7].

In [9], Canard *et al.* present list signatures that add a property to linkable group signature: the identity of the signer of two linked signatures can be publicly computed. Few years later, Canard *et al.* give a list signature construction for ad-hoc group based on ring signature [8]. In this scheme, the signature size grows linearly with the group size. In [2], the authors give a constant size identity-based ring signature scheme with similar properties (linkability and traceability).

Anonymous authentication is closely related to group/ring signatures. For instance, k-times anonymous authentications [22], [19], [23] allow a group member to anonymously authenticate himself k times, but the $(k+1)^{\text{th}}$ authentication allows the verifier to trace his identity. In [3] the authors adapt k-times anonymous authentications to linkable group signatures. This signature primitive allows everyone to trace a user that has produced more than k signatures by linking two of theses signatures. This primitive can be viewed as a generalization of list signature schemes. However, the generalization is incomplete since **only two signatures among $\mathbf{k+1}$ are linked, and the anonymity of the signer is publicly revealed only for these two signatures** but his other signatures remain anonymous. Moreover, the number k is the same for each group member. To the best of our knowledge, there exists no ring signature that ensures full anonymity for members that produce less than k signatures, and public full linkability/traceability on all signatures of a member after the $(k+1)^{\text{th}}$ signature (for a personal bound k). Solving this problem is the goal of our k-times full traceable ring signature primitive.

c) Outline: In the next section, we recall some cryptographic notions. In Section III, we present our security models. In Section IV, we give our scheme and discuss about its security. In Section V, we give applications of our scheme. Finally, we conclude in Section VI.

II. Cryptographic Tools

We present some cryptographic assumptions and some results on zero knowledge proofs. First, we recall the decisional Diffie-Hellman assumption and his bilinear variant.

Definition 1 (Decisional Diffie-Hellman [4])**.** *Let $\mathbb{G}$ be a multiplicative group of prime order p and $g \in \mathbb{G}$ be a generator. The* decisional Diffie-Hellman problem *(DDH) is to decide whether $z = ab$ given (g^a, g^b, g^z) for unknown $a, b \xleftarrow{\$} \mathbb{Z}_p^*$. The* decisional Diffie-Hellman hypothesis *states that there exists no polynomial time algorithm that solves DDH with non-negligible advantage.*

Definition 2 (Bilinear Decisional Diffie-Hellman [6])**.** *Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_t$ be three groups of prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ be a type 3 non-degenerate bilinear pairing. Let $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ be two generators. The* Bilinear Decisional Diffie-Hellman problem *(BDDH) is to decide whether $z = abc$ given $(g_1^a, g_1^b, g_2^c, e(g_1, g_2)^z)$ for unknown $a, b, c \xleftarrow{\$} \mathbb{Z}_p^*$. The* Bilinear Decisional Diffie-Hellman problem *states that there exists no polynomial time algorithm that solves BDDH with non-negligible advantage.*

In this work, we use the follwing variant of BDDH.

Definition 3 (2-Bilinear Decisional Diffie-Hellman)**.** *Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_t$ be three groups of prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ be a type 3 non-degenerate bilinear pairing. Let $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ be two generators. The* Bilinear Decisional Diffie-Hellman problem *(2BDDH) is to decide whether $z = abd$ given $(g_1^a, g_1^b, g_2^c, g_2^d, e(g_1, g_2)^{abc}, e(g_1, g_2)^z)$ for unknown $a, b, c, d \xleftarrow{\$} \mathbb{Z}_p^*$. The* 2-Bilinear Decisional Diffie-Hellman problem *states that there exists no polynomial time algorithm that solves 2BDDH with non-negligible advantage.*

This assumption is very close to BDDH. Actually, an instance of 2BDDH contains $(g_1^a, g_1^b, g_2^d, e(g_1, g_2)^z)$ that is an instance of the BDDH problem and two additional elements $(g_2^c, e(g_1, g_2)^{abc})$. We do not know any reduction from 2BDDH to BDDH (or any other standard assumption such that the decisional Diffie-Hellman assumption in $\mathbb{G}_1$ [4] or the pairing inversion problem [14]). However, this problem seems to be difficult to solve: values $(g_2^c, e(g_1, g_2)^{abc})$ are hard to exploit since $e(g_1, g_2)^{abc}$ is indistinguishable to a random group element under BDDH, and it is hard to extract g_1^{ab} from $(g_1^a, g_1^b, g_2^c, e(g_1, g_2)^{abc})$ under the pairing inversion problem.

The hardness of this assumption can be argued by the following property: $\mathsf{2BDDH}$ is hard under BDDH when e is a type 1 or type 2 pairing. Indeed, if $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ and $g_1 = g_2 = g$, it is possible to transform an instance $(g^a, g^b, g^d, e(g,g)^z)$ of BDDH into an instance of $\mathsf{2BDDH}$. Picking $c \in \mathbb{Z}_p^*$, one can compute g^c and $e(g^a, g^b)^c = e(g,g)^{abc}$ and build the instance $(g^a, g^b, g^c, g^d, e(g,g)^{abc}, e(g,g)^z)$ which is an instance of $\mathsf{2BDDH}$. Then, knowing an algorithm that solves the $\mathsf{2BDDH}$ problem, we can deduce an algorithm that solves the BDDH problem in a similar runing time. It is possible to construct a similar reduction using type 2 pairing. Unfortunately, the same reduction does not work for type 3 but the problem seems to be at least as hard as in the type 1 or type 2 cases.

Definition 4 (Zero-knowledge proofs [15])**.** *A* proof of knowledge *is a two-party protocol between two polynomial time algorithms P (the* prover*) and V (the* verifier*). It allows the prover P to convince the verifier V that he knows a solution s to the instance $\mathcal{I}$ of a problem $\mathcal{P}$. Such a protocol is said* zero-knowledge proof of knowledge *(ZKP) if it satisfies the following properties:*

Completeness: *If P knows s, then he is able to convince V (i.e. V outputs* `"accept"`*).*

Soundness: *If P does not know s, then he is not able to convince V (i.e. V outputs* `"reject"`*) except with negligible probability.*

Zero-knowledge: *V learns* nothing *about s except $\mathcal{I}$, i.e. there exists a probabilistic polynomial time algorithm* `Sim` *(called the* simulator*) such that outputs of the real protocol and outputs of $\mathtt{Sim}(\mathcal{I}, \mathsf{input}_V)$ follow the same probability distribution, where input_V denotes the input used by V for the real protocol.*

Honest-verifier ZKP *(HZKP) is a weaker notion of ZKP which is restricted to case where the verifier is* honest*, i.e. V correctly runs the protocol.*

If we only have one transaction from the prover to the verifier, we say that the ZKP is *non-interactive* (NIZKP). In the litterature, *Sigma protocols* are ZKP with three exchanges between the prover and the verifier: a commitment, a challenge, and a response (by example the Schnorr protocol [21]). If the challenge is chosen on a large set, it is possible to transform a sigma protocol into a NIZKP using the Fiat-Shamir heuristic [13] replacing the challenge by the digest of a hash function on the commitment.

Finally, our scheme uses the generic transformation of ZKP designed by [12]. The authors propose a generic transformation from the ZKP of the solution to some problem instance to a ZKP of the solution to one problem instance out of n problem instances (without revealing this problem instance). This transformation holds with any sigma protocol. The computational and space cost of the resulting ZKP is n times the cost of the primary ZKP. It is possible to use the Fiat-Shamir transformation on such a ZKP to obtain an equivalent NIZKP.

III. Model and Security

We first formally define *k-times full traceable ring signature* (k-FTRS) scheme, next we define the security models.

Let k be the maximum of authorized anonymous signatures then a k-FTRS is a ring signature scheme that has three additional functionalities depending on the parameter k:

- (i) a *link algorithm* allows users to link two signatures produced by the same member who has produced more than k signatures;
- (ii) a *match algorithm* extracts the identity of a member u and a *tracer*, denoted by $\omega_{(E,u)}$, from two linked signatures for the same event E;
- (iii) a *trace algorithm* allows users to decide whether a signature has been produced by the user u for the same event E, using the corresponding tracer $\omega_{(E,u)}$.

Each group member has his own parameter k_u that is used to generate his pair of signing/verification key. Thus, to publicly detect a cheater who has produced more than k_u signatures in a set of r signatures, it suffices to use the link algorithm on all pairs of signatures. The match algorithm allows users to identify the cheater and returns a tracer $\omega_{(E,u)}$. Using this tracer and the trace algorithm, everyone can detect all other signatures produced by the identified cheater. The number of calls to the link algorithm is quadratic in r and the the number of calls to the trace algorithm is linear in r.

As it is often required in linkable signatures, all signatures are computed for a particular event. Thus, link, match and trace algorithms can be only used for signatures coming from the same event. More formally, the signature algorithm requires a bit string E, called an *event*, that corresponds to the *identification* of a given event (for example the concatenation of the date, title and the location of the election).

Definition 5 (k-times full traceable ring signature (k-FTRS))**.** *A k-FTRS is defined by the followings algorithms:*

$\mathsf{Init}(1^t)$**:** *This algorithm outputs an init value from security parameter t.*

$\mathsf{Gen}(\mathsf{init}, k)$**:** *This algorithm outputs a signing key pair $(\mathsf{ssk}, \mathsf{svk})$ from init and a* threshold value *k denoting the maximum number of anonymous signatures authorized for the key ssk.*

$\mathsf{Sig}_E(\mathsf{ssk}, m, L, j)$**:** *This algorithm outputs a signature σ on the message m using the event E, the signing key ssk, the set of public keys of all members of the group L and the witness $j \in \{1, \ldots, k\}$, where k is the value used to generate ssk.*

$\mathsf{Ver}_E(L, \sigma, m)$**:** *This algorithm checks that σ is a valid signature of m for the event E and the set L.*

$\mathsf{Link}_E(L_1, L_2, \sigma_1, \sigma_2, m_1, m_2)$**:** *This algorithm checks that the two signatures σ_1 and σ_2, for the messages m_1 and*

m_2 and the sets of public keys L_1 and L_2, come from the same signing key ssk for the event E. In this case, it returns 1, else it returns 0. This algorithm links only two signatures of the same member out of $k+1$, for k the value used to generate ssk.

$\mathsf{Match}_E(L_1, L_2, \sigma_1, \sigma_2, m_1, m_2)$**:** *This algorithm outputs $\perp$ if $\mathsf{Link}_E(L_1, L_2, \sigma_1, \sigma_2, m_1, m_2) \neq 1$. Else, let svk_u be the key of the signer u who has produced σ_1 and σ_2. If $\mathsf{svk}_u \in L_1 \cup L_2$ then this algorithm outputs svk_u and a tracer $\omega_{(E,u)}$, else $\perp$.*

$\mathsf{Trace}_E(L, \sigma, m, \omega_{(E,u)})$**:** *This algorithm checks that the signature σ of m for the set of public keys L has been produced in the event E by the user u using the tracer $\omega_{(E,u)}$. In this case it returns 1, else 0.*

The following definition presents the correctness of such a scheme. Loosely speacking, a k-FTRS is correct when the algorithms Ver, Link, Match and Trace *correctly work* using signatures coming from the algorithm Sig.

Definition 6 (Correctness)**.** *We say that a k-FTRS is correct if for any key pair $(\mathsf{ssk}_i, \mathsf{svk}_i)$ generated by $\mathsf{Gen}(\mathsf{init}, k)$ and any signatures $\sigma = \mathsf{Sig}_E(\mathsf{ssk}_i, m, L, l)$, $\sigma_1 = \mathsf{Sig}_E(\mathsf{ssk}_i, m_1, L_1, j)$ and $\sigma_2 = \mathsf{Sig}_E(\mathsf{ssk}_i, m_2, L_2, j)$ (where $l, j \in \{1, \ldots, k\}$), the following conditions hold:*

1) *$\mathsf{Ver}_E(L, \sigma, m)$ outputs 1.*
2) *$\mathsf{Link}_E(L_1, L_2, \sigma_1, \sigma_2, m_1, m_2)$ outputs 1.*
3) *If the algorithm $\mathsf{Match}_E(L_1, L_2, \sigma_1, \sigma_2, m_1, m_2)$ outputs (svk, ω) then $\mathsf{svk} = \mathsf{svk}_i$.*
4) *If the algorithm $\mathsf{Match}_E(L_1, L_2, \sigma_1, \sigma_2, m_1, m_2)$ outputs (svk, ω) then $\mathsf{Trace}_E(=, \sigma, m, \omega)1$.*

The first point checks the verification algorithm works correctly for any signature generated by the algorithm Sig. The second one shows that the algorithm Link outputs 1 if the two given signatures use the same witnesss j and the same signature key ssk_i. The third point ensures that the algorithm Match outputs the verification key associated to two given linked signatures. The last point verifies that the tracer and the verification signature key svk_i outputed by the algorithm Match allow the algorithm Trace to trace all the signatures produced by ssk_i.

We formalize the following properties of k-FTRS:

Unforgeability: It is computationally infeasible to forge a valid signature without the secret key of a group member.

Traceability: More then k signatures coming from the same user in the same event are always traceable.

Anonymity: It is computationally infeasible to determine the identity of an honest user from less than $(k+1)$ of his signatures for each event.

Our security models are based on [18] that formalizes the security of linkable ring signatures.

Unforgeability: A k-FTRS is unforgeable when there exists no polynomial adversary able to create a new valid signature for the group. The adversary has access to a signature oracle that computes signatures for given messages and events using the secret key of chosen user. To win the adversary must generate a valid signature that does not come from the oracle. In the following, we denote by $\mathtt{out}_{\mathcal{O}}$ the set of all the values outputed by the oracle $\mathcal{O}$ during an experiment. Unforgeability is defined as follows.

Definition 7 (EUF-CMA security)**.** *Let P be a k-FTRS of security parameter t and let $\mathcal{A}$ be a polynomial time adversary. We define the (n,k)-existential unforgeability against chosen message attack experiment for $\mathcal{A}$ against P as follows:*

$\mathbf{Exp}_{P,\mathcal{A}}^{(n,k)\text{-euf-cma}}(t)$:
$\mathsf{init} \leftarrow \mathsf{Init}(1^t)$
$\forall\ i \in \{0, \ldots, n\}, (\mathsf{ssk}_i, \mathsf{svk}_i) \leftarrow \mathsf{Gen}(\mathsf{init}, k)$
$U \leftarrow \{\mathsf{svk}_i\}_{0 \leq i \leq n}$
$(L_*, \sigma_*, m_*, E_*) \leftarrow \mathcal{A}^{\mathcal{SO}_1(\cdot)}(t, U)$
if $\mathsf{Ver}_{E_*}(L_*, \sigma_*, m_*) = 1$ *and* $(L_*, \sigma_*, m_*, E_*) \notin \mathtt{out}_{\mathcal{SO}_1}$
then output 1 *else output* 0.

Where $\mathcal{SO}_1(\cdot)$ is a signing oracle that takes $(\mathsf{svk}_i, L, E, m, j)$ as input. If $\mathsf{svk}_i \notin U$ then it returns $\perp$, else it computes $\sigma = \mathsf{Sig}_E(\mathsf{ssk}_i, m, L, j)$ and returns (L, σ, m, E). The advantage of the adversary $\mathcal{A}$ against (n,k)-EUF-CMA is $\mathsf{Adv}_{P,\mathcal{A}}^{(n,k)\text{-euf-cma}}(t) = \Pr[\mathsf{Exp}_{P,\mathcal{A}}^{(n,k)\text{-euf-cma}}(t) = 1]$. We define the advantage on (n,k)-EUF-CMA experiment by $\mathsf{Adv}_{P}^{(n,k)\text{-euf-cma}}(t) = \max_{\mathcal{A} \in \text{POLY}(t)}\{\mathsf{Adv}_{P,\mathcal{A}}^{(n,k)\text{-euf-cma}}(t)\}$. We say that a k-FTRS scheme P is (n,k)-EUF-CMA secure when the advantage $\mathsf{Adv}_{P}^{(n,k)\text{-euf-cma}}(t)$ is negligible.

Traceability: A k-FTRS is traceable when there exists no polynomial adversary who is able to generate at least k valid signatures for the same event knowing only one secret key from the group such that the match and trace algorithms fail to trace the corresponding public key on each signatures. To help him, the adversary has access to a signature oracle that computes signatures for given messages and given events using secret keys of chosen users. This property is formally given in the following definition.

Definition 8 (Traceability)**.** *Let P be a k-FTRS of security parameter t and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of algorithm in $\text{POLY}(t)$. We define the (n,k)-traceability experiment for adversary $\mathcal{A}$ against P as follows:*

$\mathbf{Exp}_{P,\mathcal{A}}^{(n,k)\text{-trace}}(t)$:
$\mathsf{init} \leftarrow \mathsf{Init}(1^t)$
$\forall\ i \in \{0, \ldots, n\}, (\mathsf{ssk}_i, \mathsf{svk}_i) \leftarrow \mathsf{Gen}(\mathsf{init}, k)$
$U \leftarrow \{\mathsf{svk}_i\}_{0 \leq i \leq n}$
$\pi \leftarrow \mathcal{A}_1^{\mathcal{SO}_1(\cdot)}(t, U)$
$(z, \{(L_i, \sigma_i, m_i)\}_{1 \leq i \leq z}, E) \leftarrow \mathcal{A}_2^{\mathcal{SO}_1(\cdot)}(t, \mathsf{ssk}_\pi, U)$
if $(z > k)$
and $(\forall i \in \{1, \ldots, z\}, \mathsf{Ver}_E(L_i, \sigma_i, m_i) = 1$
and $(L_i, \sigma_i, m_i, E) \notin \mathtt{out}_{\mathcal{SO}_1})$
and $((\forall\ a, b, \mathsf{Link}_E(L_a, L_b, \sigma_a, \sigma_b, m_a, m_b) \neq 1)$
or $(($
$\exists\ a, b, i, \mathsf{Match}_E(L_a, L_b, \sigma_a, \sigma_b, m_a, m_b) = (\mathsf{svk}_*, \omega_\pi)$
$\Rightarrow (\mathsf{svk}_* \neq \mathsf{svk}_\pi$ *or* $\mathsf{Trace}_E(L_i, \sigma_i, m_i, \omega_\pi) \neq 1)$
$)))$
then output 1 *else output* 0.

Where $\mathcal{SO}_1(\cdot)$ is defined as in Definition 7. The advantage of the adversary $\mathcal{A}$ against (n,k)-traceability is de-

fined by $\mathsf{Adv}_{P,\mathcal{A}}^{(n,k)\text{-}trace}(t) = Pr[Exp_{P,\mathcal{A}}^{(n,k)\text{-}trace}(t) = 1]$. *We define the advantage on* (n,k)*-traceability experiment by* $\mathsf{Adv}_{P}^{(n,k)\text{-}trace}(t) = \max_{\mathcal{A}\in\mathrm{POLY}(t)}\{\mathsf{Adv}_{P,\mathcal{A}}^{(n,k)\text{-}trace}(t)\}$. *We say that a* k*-FTRS scheme* P *is* (n,k)*-traceable when the advantage* $\mathsf{Adv}_{P}^{(n,k)\text{-}trace}(t)$ *is negligible.*

Anonymity: A k-FTRS is anonymous when there exists no polynomial adversary able to distinguish the signer of a given message between two given honest group members. The adversary choses two honest user's public keys, a group of users, a message and an event and sends it to the challenger. The challenger signs the message using one of the two corresponding secret keys. Then the adversary must guess who the signer is. To help him, the adversary has access to a signature oracle that computes signatures for given messages and given events using the secret key of chosen users. However, the oracle refuses to produce more that $k-1$ signatures for the two chosen public keys and the corresponding event used to produce the challenge since signatures would be traceable in this case. This notion is formally introduced in the following definition.

Definition 9 (Anonymity)**.** *Let* P *be a* k*-FTRS of security parameter* t *and let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be polynomial time adversary. We define the* (n,k)*-anonymity experiment for adversary* $\mathcal{A}$ *against* P *as follows:*

$\mathbf{Exp}_{P,\mathcal{A}}^{(n,k)\text{-anon}}(t)$:
$b \leftarrow \{0,1\}$
$init \leftarrow Init(1^t)$
$\forall\ i \in \{0,\ldots,n\}, (ssk_i, svk_i) \leftarrow Gen(init, k)$
$U \leftarrow \{svk_i\}_{0\le i\le n}$
$(\pi_0, \pi_1, L_*, E_*, m_*, j_*) \leftarrow \mathcal{A}_1^{\mathcal{SO}_2(\cdot)}(t, U)$
$\sigma_0 \leftarrow \mathcal{SO}_2(svk_{\pi_0}, L_*, E_*, m_*, j_*)$
$\sigma_1 \leftarrow \mathcal{SO}_2(svk_{\pi_1}, L_*, E_*, m_*, j_*)$
$b' \leftarrow \mathcal{A}_2^{\mathcal{SO}_2(\cdot)}(t, \sigma_b, U)$
output $b = b'$.

Where $\mathcal{SO}_2(\cdot)$ *is a signing oracle that takes* (svk_i, L, E, m, j) *as input. If* $j > k$ *of* $svk_i \notin U$ *then it returns* $\perp$ *and aborts. If* svk_i *and* E *are asked together for the first time, then this oracle instantiates* $wit(svk_i, E) \leftarrow \emptyset$. *During the second phase, if* $E = E_*$ *and* $(i = \pi_0 \vee i = \pi_1)$ *and* $j \in wit(svk_i, E)$ *then it returns* $\perp$ *and aborts. Finally, it computes* $wit(svk_i, E) \leftarrow wit(svk_i, E) \cup \{j\}$ *and returns* $Sig_E(ssk_i, m, L, j)$*. The advantage of the adversary* $\mathcal{A}$ *against* (n,k)*-anonymity is* $\mathsf{Adv}_{P,\mathcal{A}}^{(n,k)\text{-}anon}(t) = \left|Pr[Exp_{P,\mathcal{A}}^{(n,k)\text{-}anon}(t) = 1] - \frac{1}{2}\right|$. *We define the advantage on* (n,k)*-anonymity experiment by* $\mathsf{Adv}_{P}^{(n,k)\text{-}anon}(t) = \max_{\mathcal{A}\in\mathrm{POLY}(t)}\{\mathsf{Adv}_{P,\mathcal{A}}^{(n,k)\text{-}anon}(t)\}$. *We say that a* k*-FTRS scheme* P *is* (n,k)*-anonymous when the advantage* $\mathsf{Adv}_{P}^{(n,k)\text{-}anon}(t)$ *is negligible.*

IV. Our Construction: Ktrace

We present the scheme Ktrace, a secure construction of our primitive. It is based on the list signature scheme for small groups of Canard *et al.* [8]. We first recall this scheme.

A. *Canard* et al. *List Signature Scheme*

Each member generates an ElGamal secret/public key pair $(\mathsf{sk} = x, \mathsf{pk} = g^x)$ for g a generator of a prime order multiplicative group. Let I be the set of user identities. The group key $\mathsf{GPK} = \{\mathsf{pk}_i\}_{i\in I}$ is the set of all member's public keys. To sign a message m for the event E, we use two hash functions H_0 and H_1 to compute the hashed values $A = \mathsf{H}_1(E,0), B = \mathsf{H}_1(E,1)$ and $u = H_0(m,E,1)$. Finally, we compute $T_1 = A^x$ and $T_2 = B^x \cdot (g^u)^x$, and we compute T_3 an *ad-hoc* NIZKP of the knowledge of the two values x and i such that $x = \log_g(\mathsf{pk}_i) = \log_A(T_1) = \log_{B\cdot g^u}(T_2)$. The signature of m is the triplet (T_1, T_2, T_3). The verification algorithm consists in checking the NIZKP T_3. Furthermore, to link two signatures $\sigma_1 = (T_{1,1}, T_{1,2}, T_{1,3})$ and $\sigma_2 = (T_{2,1}, T_{2,2}, T_{2,3})$, it suffices to check that $T_{1,1} = T_{2,1}$. Finally, to match the identity of a signer using two signatures, we compute $(T_{1,2}/T_{2,2})^{1/u_1-u_2} = (B^x\cdot(g^{u_1})^x/B^x\cdot(g^{u_2})^x)^{1/u_1-u_2}$ which is equal to the public key $\mathsf{pk} = g^x$.

This scheme allows users to produce only one anonymous signature per key. As a consequence, it suffices to generate k keys per user to allow them to produce k anonymous signatures. Moreover, any extra signature can be linked to one of the first k signatures because a user must use the same key twice to produce more than k signatures. However this solution does not allow users to recover all the messages of a cheater.

B. *An Helpful ZKP*

In order to have all the necessary tools to construct our scheme, we build the following ZKP, denoted Π.

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_t$ be three groups of prime order p, g_1 and g_2 be two respective generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_t$ be a non-degenerate bilinear pairing and n be an integer. Let A, B, C, W, T_1, T_2, T_3, be seven elements of $\mathbb{G}_1$, T_4 be an element of $\mathbb{G}_2$, T_5 be an element of $\mathbb{G}_t$ and u and v be two elements of $\mathbb{Z}_p^*$. Finally, for all $i \in \{1,\cdots,n\}$, let (h_i, l_i) be some couples of $\mathbb{G}_1^2$. Using

$$Q = (g_1, g_2, A, B, C, W, u, v, T_1, T_2, T_3, T_4, T_5)$$
$$S = \{(Q, h_i, l_i)\}_{i\in\{1,\ldots,n\}}$$

we build Π, a NIZKP of knowledge of $(x,y,z) \in (\mathbb{Z}_p^*)^3$ such that $T_1 = A^x$; $T_2 = B^x \cdot g_1^{u\cdot y}$; $T_3 = C^x \cdot W^{v\cdot y}$; $T_4 = g_2^z$; $T_5 = e(W^y, T_4)$; $h = g_1^x$ and $l = g_1^y$ for one $(Q,h,l) \in S$.

We first describe the interactive case Π_1 where $n = 1$, hence there is only one couple (h,l). The description of Π_1 is given in Figure 1. It is based on the classical methodology of ZKP of discrete logarithm knowledge [21] and equality of two discrete logarithms [10]. This proof is by construction a sigma-protocol.

Lemma 10. *The ZKP* Π_1 *is complete, sound, and honest-verifier zero-knowledge.*

See Appendix A, for the proof of Lemma 10. As Π_1 is honest-verifier zero knowledge and a sigma protocol, we can use the generic transformation of [12] to obtain the interactive

Prover P		**Verifier** V
Secret: (x, y, z)		Public: $(g_1, g_2, A, B, C, W, u, v, T_1, T_2, T_3, T_4, T_5, h, l)$
$r, s, q \xleftarrow{\$} \mathbb{Z}_p^*$ $R_0 = g_1^r$; $R_1 = A^r$; $R_2 = B^r$; $R_3 = C^r$ $S_0 = g_1^s$; $S_1 = W^s$; $S_2 = e(W, T_4)^s$ $Q_0 = g_2^q$		
	$\xrightarrow{(R_0,R_1,R_2,R_3,S_0,S_1,S_2,Q_0)}$	$\epsilon \xleftarrow{\$} \mathbb{Z}_p^*$
$\bar{x} = r + x \cdot \epsilon$; $\bar{y} = s + y \cdot \epsilon$; $\bar{z} = q + z \cdot \epsilon$	$\xleftarrow{\epsilon}$	
	$\xrightarrow{(\bar{x},\bar{y},\bar{z})}$	Check that: $A^{\bar{x}} = R_1 \cdot {T_1}^{\epsilon}$; $B^{\bar{x}} \cdot g_1^{u \cdot \bar{y}} = R_2 \cdot S_0^u \cdot T_2^{\epsilon}$ $C^{\bar{x}} \cdot W^{v \cdot \bar{y}} = R_3 \cdot S_1^v \cdot T_3^{\epsilon}$; $g_2^{\bar{z}} = Q_0 \cdot T_4^{\epsilon}$ $e(W, T_4)^{\bar{y}} = S_2 \cdot T_5^{\epsilon}$; $g_1^{\bar{x}} = R_0 \cdot h^{\epsilon}$; $g_1^{\bar{y}} = S_0 \cdot l^{\epsilon}$

Fig. 1. Interactive zero knowledge proof Π_1.

version of our proof for any $n \geq 0$. Finally, using this transformation and the Fiat-Shamir heuristic on Π_1, we build the non-interactive proof Π in the random oracle model.

Theorem 11. *The NIZKP Π is complete, sound, and zero-knowledge in the random oracle model.*

It is a direct implication of [12] and Lemma 10.

C. Our Construction: Ktrace

Our construction, given in Scheme 1, requires three groups $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_t$ of prime order p and a bilinear pairing e. Let g_1 and g_2 be two respective generators of $\mathbb{G}_1$ and $\mathbb{G}_2$. Each user generates k ElGamal signing key pairs $\{(x_{i,j}, g_1^{x_{i,j}})\}_{1 \leq j \leq k}$ and an extra key pair $(x_i, g_1^{x_i})$ used to identify him with the algorithms match and trace.

To sign a message, a user picks $r \xleftarrow{\$} \mathbb{Z}_p^*$ and chooses one of these ElGamal secret keys and computes $T_1 = A^{x_{i,j}}$ and $T_2 = B^{x_{i,j}} \cdot (g_1^u)^{x_i}$ as in the scheme [8] except that $u = \mathsf{H}_1(E, m, 0, g_2^r)$. Thus, using these two values, two signatures can be linked when the same secret key is used, and it is possible to deduce the public key $g_1^{x_i}$ from theses two signatures. Then the signer computes additional values T_3, T_4, T_5 and T_6 as follows: he sets $C = \mathsf{H}_0(E, 2), W = \mathsf{H}_0(E, 3)$ and $v = H_1(E, m, 1, g_2^r)$ and sets $T_3 = C^{x_{i,j}} \cdot W^{v \cdot x_i}, T_4 = g_2^r$ and $T_5 = e(W, T_4)^{x_i}$. As in list signatures, the signer computes in T_6 a non-interactive proof that all other parts of the signatures T_1, T_2, T_3, T_4 and T_5 are *"correctly formed"* according to one of the verification keys of the group using the proof Π given in Section IV-B. The complete signature is the tuple $(T_1, T_2, T_3, T_4, T_5, T_6)$ and is verified by checking the validity of the proof T_6.

Note that using two linked signatures with respective third terms $T_{1,3} = C^{x_{i,j}} \cdot W^{v_1 \cdot x_i}$ and $T_{2,3} = C^{x_{i,j}} \cdot W^{v_2 \cdot x_i}$, it is possible to compute the *tracer* $\omega_{(E,i)} = (T_{1,3}/T_{2,3})^{1/(v_1 - v_2)} = W^{x_i}$ in addition to the public key $g_1^{x_i}$ in the match algorithm, for an evant E[1]. Finally, to trace a signature T generated by the owner of the public key $g_1^{x_i}$ using the tracer ω_i, anybody can check that $e(\omega_i, T_4) = e(W^{x_i}, T_4) = e(W, T_4)^{x_i}$. Since the proof T_6 assures that members T_3, T_4 and T_5 are well formed, this equation always holds when the signer is the owner of the public key $g_1^{x_i}$.

[1] We omit the event E in the notation of the tracer. When it is clear from the context, we simply write ω_i.

Scheme 1 (Ktrace scheme)**.** *Ktrace is a k-FTRS with the following algorithms:*

$\mathsf{Init}(1^t)$**:** *This algorithm generates three groups $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_t$ of prime order p, two respective generators g_1 and g_2 of $\mathbb{G}_1$ and $\mathbb{G}_2$, a non-degenerate bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_t$ of type 3 and two hash functions $\mathsf{H}_0 : \{0,1\}^* \rightarrow \mathbb{G}_1$ and $\mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. The algorithm outputs $\mathsf{init} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, e, \mathsf{H}_0, \mathsf{H}_1)$.*

$\mathsf{Gen}(\mathsf{init}, k_i)$**:** *This algorithm randomly picks x_i in $\mathbb{Z}_p^*$. It then picks k_i other values $x_{i,j}$ in $\mathbb{Z}_p^*$ for j in $\{1, \ldots, k_i\}$. It sets $\mathsf{svk}_i = (g_1^{x_i}, \{g_1^{x_{i,j}}\}_{1 \leq j \leq k_i})$ and $\mathsf{ssk}_i = (x_i, \{x_{i,j}\}_{1 \leq j \leq k})$ and returns $(\mathsf{svk}_i, \mathsf{ssk}_i)$.*

$\mathsf{Sig}_E(\mathsf{ssk}_i, m, L, j)$**:** *Using $\mathsf{ssk}_i = (x_i, \{x_{i,j}\}_{1 \leq j \leq k})$ and the event E, this algorithm picks $r \xleftarrow{\$} \mathbb{Z}_p^*$ and computes the following values:*

$$A = \mathsf{H}_0(E, 0) \qquad B = \mathsf{H}_0(E, 1)$$
$$C = \mathsf{H}_0(E, 2) \qquad W = \mathsf{H}_0(E, 3)$$
$$u = \mathsf{H}_1(E, m, 0, g_2^r) \qquad v = H_1(E, m, 1, g_2^r)$$
$$T_1 = A^{x_{i,j}} \qquad T_2 = B^{x_{i,j}} \cdot g_1^{u \cdot x_i}$$
$$T_3 = C^{x_{i,j}} \cdot W^{v \cdot x_i} \qquad T_4 = g_2^r$$
$$T_5 = e(W, T_4)^{x_i}$$

Let I be the set of user indexes such that:

$$L = \{(g_1^{x_i}, \{g_1^{x_{i,j}}\}_{1 \leq j \leq k_i})\}_{i \in I}$$

We set:

$$Q = (g_1, g_2, A, B, C, W, u, v, T_1, T_2, T_3, T_4, T_5)$$

The algorithm computes the following set:

$$S = \left\{ \left(Q, g_1^{x_i}, g_1^{x_{i,j}} \right) \right\}_{(i,j) \in I \times \{1, \cdots, k_i\}}$$

Finally, it generates a non-interactive proof of knowledge T_6 using Π (see section IV-B) of the solution $(x_i, x_{i,j}, r)$ of one instance $\mathcal{I} = (Q, g_1^{x_i}, g_1^{x_{i,j}})$ out of the instance set S without revealing neither $(x_i, x_{i,j}, r)$ nor $\mathcal{I}$. It outputs the signature $\sigma = (T_1, T_2, T_3, T_4, T_5, T_6)$.

$\mathsf{Ver}_E(L,\sigma,m)$**:** *We set* $\sigma = (T_1,T_2,T_3,T_4,T_5,T_6)$. *Let* I *be a set of user indexes such that* $L = \{(g_1^{x_i},\{g_1^{x_{i,j}}\}_{1\le j\le k_i})\}_{\forall i\in I}$. *This algorithm computes:*

$$A = \mathsf{H}_0(E,0) \qquad B = \mathsf{H}_0(E,1)$$
$$C = \mathsf{H}_0(E,2) \qquad W = \mathsf{H}_0(E,3)$$
$$u = \mathsf{H}_1(E,m,0,T_4) \qquad v = H_1(E,m,1,T_4)$$

It also computes:

$$Q = (g_1,g_2,A,B,C,W,u,v,T_1,T_2,T_3,T_4,T_5)$$
$$S = \left\{\left(Q,g_1^{x_i},g_1^{x_{i,j}}\right)\right\}_{(i,j)\in I\times\{1,\cdots,k_i\}}$$

Finally, it returns 1 *if the proof* T_6 *is valid for* Π *(see section IV-B) according to the set of instances* S.

$\mathsf{Link}_E(L_1,L_2,\sigma_1,\sigma_2,m_1,m_2)$**:** *We set*

$$\sigma_1 = (T_{1,1},T_{1,2},T_{1,3},T_{1,4},T_{1,5},T_{1,6})$$
$$\sigma_2 = (T_{2,1},T_{2,2},T_{2,3},T_{2,4},T_{2,5},T_{2,6})$$

This algorithm returns 1 *if and only if* $\mathsf{Ver}_E(L_1,\sigma_1,m_1) = \mathsf{Ver}_E(L_2,\sigma_2,m_2) = 1$ *and* $T_{1,1} = T_{2,1}$.

$\mathsf{Match}_E(L_1,L_2,\sigma_1,\sigma_2,m_1,m_2)$**:** *We set*

$$\sigma_1 = (T_{1,1},T_{1,2},T_{1,3},T_{1,4},T_{1,5},T_{1,6})$$
$$\sigma_2 = (T_{2,1},T_{2,2},T_{2,3},T_{2,4},T_{2,5},T_{2,6})$$

This algorithm computes the following hashed values:

$$u_1 = \mathsf{H}_1(E,m_1,0,T_{1,4}) \quad v_1 = H_1(E,m_1,1,T_{1,4})$$
$$u_2 = \mathsf{H}_1(E,m_2,0,T_{2,4}) \quad v_2 = H_1(E,m_2,1,T_{2,4})$$

If $\mathsf{Link}_E(L_1,L_2,\sigma_1,\sigma_2,m_1,m_2) = 0$ *this algorithm outputs* $\perp$*, else it computes:*

$$\mathsf{id} = \left(\frac{T_{1,2}}{T_{2,2}}\right)^{\frac{1}{u_1-u_2}}; \quad \omega_{(E,i)} = \left(\frac{T_{1,3}}{T_{2,3}}\right)^{\frac{1}{(v_1-v_2)}}$$

Let $\mathsf{svk}_i = (g_1^{x_i},\{g_1^{x_{i,j}}\}_{1\le j\le k_i})$ *be the element of* $L_1\cup L_2$ *such that* $g_1^{x_i} = \mathsf{id}$. *If such an element does not exist then this algorithm outputs* $\perp$*, else this algorithm outputs* $(\mathsf{svk}_i,\omega_{(E,i)})$.

$\mathsf{Trace}_E(L,\sigma,m,\omega_{(E,i)})$**:** *This algorithm set* $\sigma = (T_1,T_2,T_3,T_4,T_5,T_6)$. *If* $\mathsf{Ver}_E(L,\sigma,m) = 1$ *and* $e(\omega_{(E,i)},T_4) = T_5$ *then it returns* 1*, else* 0.

D. Correctness

Theorem 12. *Ktrace is correct.*

Proof. Ver: The set of instances S is similarly computed in both algorithms Sig and Ver. The part T_6 of a signature is a proof Π on the set of instances S. Since the verification algorithm constists to check this proof, and since Π is complete, then algorithm Ver is correct, *i.e.* $\mathsf{Ver}_E(L,\mathsf{Sig}_E(\mathsf{ssk}_i,m,L,j),m) = 1$.

Link: We show that given two signatures $\sigma_1 = \mathsf{Sig}_E(\mathsf{ssk}_i,m_1,L_1,j)$ and $\sigma_2 = \mathsf{Sig}_E(\mathsf{ssk}_i,m_2,L_2,j)$, then $\mathsf{Link}_E(L_1,L_2,\sigma_1,\sigma_2,m_1,m_2) = 1$. Using the same event E, the same secret key $\mathsf{ssk}_i = (x_i,\{x_{i,j}\}_{1\le j\le k})$ and the same witness j, the first part of the signature is $T_1 = A^{x_{i,j}}$ where $A = \mathsf{H}_0(E,0)$. Let $T_{1,1}$ and $T_{2,1}$ be the first part of the two signatures (σ_1,σ_2), clearly $T_{1,1} = T_{2,1} = A^{x_{i,j}}$.

Match: We show that given two signatures $\sigma_1 = \mathsf{Sig}_E(\mathsf{ssk}_i,m_1,L_1,j)$ and $\sigma_2 = \mathsf{Sig}_E(\mathsf{ssk}_i,m_2,L_2,j)$ then $\mathsf{Match}_E(L_1,L_2,\sigma_1,\sigma_2,m_1,m_2)$ outputs the public key of the signer svk_i:

$$\mathsf{id} = \left(\frac{T_{1,2}}{T_{2,2}}\right)^{\frac{1}{u_1-u_2}} = \left(\frac{B^{x_{i,j}}\cdot g_1^{u_1\cdot x_i}}{B^{x_{i,j}}\cdot g_1^{u_2\cdot x_i}}\right)^{\frac{1}{u_1-u_2}}$$
$$= g_1^{x_i\cdot\frac{u_1-u_2}{u_1-u_2}} = g_1^{x_i}$$

Then the algorithm outputs $\mathsf{svk}_i = (g^{x_i},\{g^{x_{i,j}}\}_{1\le j\le k_i})$ which is the public key corresponding to ssk_i.

Trace: Given two signatures $\sigma_1 = \mathsf{Sig}_E(\mathsf{ssk}_i,m_1,L_1,j)$ and $\sigma_2 = \mathsf{Sig}_E(\mathsf{ssk}_i,m_2,L_2,j)$ we show that $\mathsf{Match}_E(L_1,L_2,\sigma_1,\sigma_2,m_1,m_2)$ outputs a tracer $\omega_{(E,i)}$ such that:

$$\omega_{(E,i)} = \left(\frac{T_{1,3}}{T_{2,3}}\right)^{\frac{1}{(v_1-v_2)}} = \left(\frac{C^{x_{i,j}}\cdot W^{v_1\cdot x_i}}{C^{x_{i,j}}\cdot W^{v_2\cdot x_i}}\right)^{\frac{1}{(v_1-v_2)}}$$
$$= W^{x_i\cdot\frac{v_1-v_2}{v_1-v_2}} = W^{x_i}$$

Then, using any m, E, L and l, we show that $\mathsf{Trace}_E(L,\mathsf{Sig}_E(\mathsf{ssk}_i,m,L,l),m,\omega_{(E,i)}) = 1$. Indeed, using $\mathsf{Sig}_E(\mathsf{ssk}_i,m,L,l) = (T_1,T_2,T_3,T_4,T_5,T_6)$ such that $T_4 = g_2^r$ and $T_5 = e(W,T_4)^{x_i}$, we have:

$$e(\omega_{(E,i)},T_4) = e(W^{x_i},T_4) = e(W,T_4)^{x_i} = T_5$$

□

E. Security

According to the security model introduced in Section III, we have the following theorem.

Theorem 13. *Ktrace is (n,k)-EUF-CMA secure,* (n,k)*-traceable and* (n,k)*-anonymous under the decisional Diffie-Hellman assumption in* $\mathbb{G}_1$ *(Def. 1) and the 2-bilinear decisional Diffie-Hellman assumption (Def. 3) in the random oracle model for any polynomial bounded* k *and* n.

Full proof of this theorem will appear in the full version of this paper. In the following, we just give some proof intuitions for each properties.

(n,k)-EUF-CMA security: In order to build a valid signature of m using E, the adversary must to compute a valid NIZKP in T_6. This proof convinces the verifier that the signer has used a valid secret key ssk according to the set of public keys L. As a consequence, since the NIZKP is sound, the signer cannot generate a valid signature except with negligible probability.

(n,k)-traceability: In a similar way, as Π is sound, a signature that success the verification check is correctly constructed. Since the adversary is not able to forge a signature using the secret key of an honest user under the soundness of Π, the traceability of Ktrace is a direct implication of the correctness of the algorithms link, match and trace.

(n,k)-anonymity: Let $T_1 = A^{x_{i,j}}$, $T_2 = B^{x_{i,j}} \cdot g_1^{u \cdot x_i}$ and $T_3 = C^{x_{i,j}} \cdot W^{v \cdot x_i}$ be the three first parts of the challenge. To discover the user's identity using this three values, the adversary must deduce $g_1^{x_i}$ or $g_1^{x_{i,j}}$ from T_1, T_2 and T_3 knowing the public hash values A, B and C. However, since the key $x_{i,j}$ is used only once per event, $A^{x_{i,j}}$, $B^{x_{i,j}}$ and $C^{x_{i,j}}$ are computationally indistinguishable to three random elements according to the Diffie-Hellman assumption in $\mathbb{G}_1$. As a consequence, these three values hide all the information about the signer public key contained in T_1, T_2 and T_3. On the other hand, let $T_4 = g_2^r$, $T_5 = e(W^{x_i}, g_2)$ and T_6 be the three last terms of the signatures. Clearly, T_6 leaks no information about the user identity since it is zero-knowledge. Moreover, it seems to be hard to deduce the signer identity from T_5 since it is hard to guess that $T_5 = e(W^{x_i}, g_2^r)$ knowing W, $g_1^{x_i}$ and g_2^r under the BDDH assumption. Actually, we do not reduce the security of Ktrace to BDDH but to $\mathsf{2BDDH}$ because asking the signing oracle, the adversary can learn the part $T_5' = e(W^{x_i}, g_2^{r'})$ of an other signature using the same key as the challenge. Thus, the adversary wins the experiment if he guesses whether $T_5 = e(W^{x_i}, g_2^r)$ or not knowing W, $g_1^{x_i}$, g_2^r, $g_2^{r'}$ and $e(W^{x_i}, g_2^{r'})$. Finally, it is computationally infeasible to guess the identity of an honest user from one of its signatures under the decisional Diffie-Hellman assumption in $\mathbb{G}_1$ and the 2-bilinear decisional Diffie-Hellman assumption.

F. Performances

The performances of the scheme Ktrace depend of n the number of users in the group and $k = \max_{0 \leq i \leq n}(k_i)$ the maximum in the set of all user threshold values k_i. The secret key of each user contains at most $(k+1)$ elements of $\mathbb{Z}_p^*$, *i.e.* the size of the public key of each user is $(k+1) \cdot s_p$ where $s_p = \log_2(p)$ is the bit size of the elements of $\mathbb{Z}_p^*$. Thus, the group public key size is $(k+1) \cdot n \cdot s_1$ where s_1 is the bit size of an element of $\mathbb{G}_1$.

In the signature algorithm, we use the NIZKP Π which is constructed using the generic transformation [12] on Π_1. The size of this NIZKP is $6 \cdot s_1 + s_2 + s_T + 3 \cdot s_p$ where s_2 is the bit size of an element of $\mathbb{G}_2$ and s_T is the bit size of an element of $\mathbb{G}_t$. The size of a NIZKP built from [12] is x times the size of the original proof, where x is the number of instances used. In our scheme, the cardinal of the set of instances $S = \{(Q, g^{x_i}, g^{x_{i,j}})\}_{(i,j) \in I \times \{1, \cdots, k_i\}}$ is at most $n \cdot$k. Then the size of the NIZKP is bounded by $(6 \cdot s_1 + s_2 + s_T + 3 \cdot s_p) \cdot n \cdot k$. Finally, the size of a signature $\sigma = (T_1, T_2, T_3, T_4, T_5, T_6)$ is $(6 \cdot s_1 + s_2 + s_T + 3 \cdot s_p) \cdot n \cdot k + 3 \cdot s_1 + s_2 + s_T$ which is in $O(n \cdot k)$. The complexity of the signing algorithm and the verification algorithm are in $O(n \cdot k)$.

The complexity of the three algorithms link, match and trace are constant. Moreover, the tracer outputted by the match algorithm is also in constant size s_1. According to the comparative table in Figure 2, we show that the signature size of our scheme loses at least a factor k compared to other ring signature schemes, particulary comparing to the list signature scheme [8]. However, it is the only one which is k-times, ad-hoc and fully traceable together. Note that a 1-times scheme is allways full traceable by construction.

Finally, note that in both applications given in the next section, k is *a priori* much smaller than n. Moreover, our applications do not implicate that k increases when n does. Thus, depending to the context, the signature size $O(n \cdot k)$ can be often considered to be relatively closed to $O(n)$.

V. Applications

A. k-times Proxy E-Voting

Linkability in group and ring signatures is a useful property to design cryptographic e-voting schemes. As a matter of fact, one can built a simple e-voting scheme from any linkable group signature scheme as follows. To vote for the candidate c, the voter signs c in σ using a linkable group signature scheme. He then publishes (c, σ) on a public bulletin board. The bulletin board is a publicly readable storage such that everybody can write data on the board but it is not possible to discard any data from it. To compute the result of the election, it is sufficient to remove all non-valid signatures according to the group public key and all linked signatures, and to count the number of ballots for each candidate. Such a scheme presents several properties:

- Ballots are anonymous thanks to the anonymity of the group signature.
- A voter cannot vote more than his number of authorized vote thanks to the linkability of the group signature.
- The result of the election can be publicly computed by anyone.
- Using linkable ring signature, this scheme does not require any manager or trust party.
- Using event oriented signatures, voters can use the same keys for several elections. In a multi-group context, using the group identity as event allows the users to use the same signing key for different groups.
- Using traceable signatures, the identity of a cheater is public, and it is possible to revoke the cheaters.

On the other hand, k-times schemes can be used to design such an e-voting scheme with proxy mechanism. In proxy voting, a signer who receives a proxy vote from another one must be able to vote one more time. Thus, any voter can vote k times where k is the number of proxy that he has received. Of course, if the voter votes more than k times, he must be traced and revoked. In this case, we would like to discard all votes of the cheater during the counting of votes. However, such a property cannot be achieved with a basic k-times signature scheme since it is not full traceable. To solve this problem, we propose the following k-times proxy e-voting scheme based on our k-FTRS scheme.

Setup: Each voter uses Gen to generate secret/public keys depending on his number of proxy k. The voters construct the group public key GPK which is the set of all voters public keys.

Initialization of an election: Users choose the event E and the set of candidates C.

Papers	Schemes	Sig. size	Ad-hoc	Times	Pub. link.	Pub. trac.	Full trac.	Events
[20]	Ring signature	$O(n)$	Yes	∞	-	-	-	-
[5]	Short group sig.	$O(1)$	No	∞	-	-	-	-
[17]	Linkable ring sig.	$O(n)$	Yes	1	Yes	No	No	No
[8]	List sig. (Ad-hoc)	$O(n)$	Yes	1	Yes	Yes	Yes	Yes
[24]	Short link. ring sig.	$O(1)$	Yes	1	Yes	No	No	No
[2]	Id-based link. ring sig.	$O(1)$	Yes	1	Yes	Yes	Yes	Yes
[3]	k-times group sig.	$O(1)$	No	k	Yes	Yes	No	Yes
	Ktrace	$\mathbf{O(n \cdot k)}$	**Yes**	**k (fine-grained)**	**Yes**	**Yes**	**Yes**	**Yes**

Fig. 2. Comparison of the group/ring signatures schemes, where - means that the property is not relevent for the scheme.

Voting protocol: Each user chooses k candidates $(c_1, \ldots, c_k) \in C^k$ and publishes on the public bulletin board the ballots $(c_j, \mathsf{Sig}_E(\mathsf{ssk}, c_j, \mathsf{GPK}, j))$ for all $j \in \{1, \ldots, k\}$ using his secret key svk.

Counting of votes: Copy all ballots from the bulletin board. First, everyone removes each ballot which contains invalid signature using the algorithm Ver. Then anyone uses the Link algorithm on each pair of ballots, and the Match algorithm on each pair of linked ballots to obtain the cheaters identities and the corresponding tracers. For each tracer, we can use the Trace algorithm on each ballot, and remove it if Trace returns 1. Finally, everyone counts the number of ballots for each candidate and deduces the election winner.

Note that cheaters can also be revoked of the group for the future elections.

B. *k-times Anonymous Veto*

A k-times anonymous veto allows members of a group to anonymously express k vetos. For example, during the organization of a conference, the members of the steering committee constitute the list of members of the technical committee. For some reason, a member of the steering committee might want to refuse the designation of some people on this list. However, members who exclude people might want to be anonymous. On the other hand, it is desirable that the number of people excluded per member is limited to a fixed value k. Members who exceed the limit must be revoked to the steering committee. In this case, it is necessary to discard all requests produced by the revoked members. Such a veto scheme can be easily designed using a k-FTRS scheme:

Setup: Each group member uses Gen to generate secret/public keys depending to his number of veto k. The members construct the group public key GPK which is the set of all members public keys.

Initialization: Users choose the event E and the set of items C.

Veto protocol: Each member chooses l items $(c_1, \ldots, c_l) \in C^k$ such that $l \leq k$. To veto each item c_j, the member publish on the public bulletin board the witness $(c_j, \mathsf{Sig}_E(\mathsf{ssk}, c_j, \mathsf{GPK}, j))$ for all $j \in \{1, \ldots, l\}$ using his secret key svk.

Cheater detection: Anybody can use the following procedure to detect a member who places more than k vetos: we use Link algorithm on each pair of witness, and Match algorithm on each pair of linked witness to obtain the cheater's identities and the corresponding tracers. Thanks to Match, it is possible to revoke the cheaters.

Finding invalid witness: All witnesses which contain an invalid signature are invalid. If the previous procedure detects cheaters, the Trace algorithm is used on each witness for each tracer outputted by Match. If Trace outputs 1 then the corresponding witness is invalid.

Limitation of other k-times schemes: Since other schemes in the literature are not full traceable, they present some limitation for building k-times anonymous veto schemes. Without full traceability, it is not possible to discard all witnesses produced by a cheater. However, the cheater is revoked, so he is no longer in the committee, and he no longer has the right to exclude people from the list. In this case, other members of the committee have no other choice but to restart the veto procedure since other witness of the cheater remain anonymous. Our scheme allows to solve this problem.

VI. Conclusion

In this paper, we give a more general definition of k-times linkable/traceable signatures and their security. Our construction allows members to chose a different threshold k for each group member, and anybody can trace all signatures generated by the same member. Future work will investigate the design of such a scheme with smaller (or constant) size of signature. On the other hand, we aim at designing k-times full traceable schemes using the same methodology on other existing linkable/traceable ring/group signatures.

References

[1] M. H. Au, S. S. M. Chow, W. Susilo, and P. P. Tsang. Short linkable ring signatures revisited. In *EuroPKI 06*. Springer, 2006.

[2] M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen. Constant-size id-based linkable and revocable-iff-linked ring signature. In *Progress in Cryptology, 7th International Conference on Cryptology in India - INDOCRYPT*. Springer, 2006.

[3] M. H. Au, W. Susilo, and S.-M. Yiu. Event-oriented k-times revocable-iff-linked group signatures. In *Information Security and Privacy, 11th Australasian Conference - ACISP*. Springer, 2006.

[4] D. Boneh. The decision Diffie-Hellman problem. In *Third Algorithmic Number Theory Symposium (ANTS)*, volume 1423 of *LNCS*. Springer, 1998. Invited paper.

[5] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Aug. 2004.

[6] D. Boneh and M. K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.
[7] E. F. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In *ACM CCS 04*, page 132145. ACM Press, 2004.
[8] S. Canard, B. Schoenmakers, M. Stam, and J. Traoré. List signature schemes. *Discrete Appl. Math.*, 154(2):189–201, Feb. 2006.
[9] S. Canard and J. Traoré. List signature schemes and application to electronic voting. In *International Workshop on Coding and Cryptography - WCC*, 2003.
[10] D. Chaum and T. P. Pedersen. Wallet databases with observers. In E. F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 89–105. Springer, Aug. 1993.
[11] D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT91*, volume 547 of *LNCS*. Springer, 1991.
[12] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y. Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 174–187. Springer, Aug. 1994.
[13] A. Fiat and A. Shamir. *Advances in Cryptology — CRYPTO' 86: Proceedings*, chapter How To Prove Yourself: Practical Solutions to Identification and Signature Problems, pages 186–194. Springer Berlin Heidelberg, Berlin, Heidelberg, 1987.
[14] S. D. Galbraith, F. Hess, and F. Vercauteren. Aspects of pairing inversion. In *Information Theory, IEEE Transactions*, volume 54, pages 5719 – 5728. IEEE, 2008.
[15] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, pages 186–208, 1989.
[16] H. Jonker, S. Mauw, and J. Pang. Privacy and verifiability in voting systems: Methods, developments and trends. *Computer Science Review*, 10:1–30, 2013.
[17] J. K. Liu, V. K. Wei, and D. S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups. In *ACISP 04*, volume 3108 of *LNCS*. Springer, 2004.
[18] J. K. Liu and D. S. Wong. Linkable ring signatures: Security models and new schemes. In *ICCSA*, LNCS. Springer, 2005.
[19] L. Nguyen and R. Safavi-Naini. Dynamic k-times anonymous authentication. In *Applied Cryptography and Network Security, Third International Conference - ACNS*. Springer, 2005.
[20] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In C. Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer, Dec. 2001.
[21] C.-P. Schnorr. Efficient identification and signatures for smart cards. In *EUROCRYPT89*, volume 434 of *LNCS*. Springer, 1989.
[22] I. Teranishi, J. Furukawa, and K. Sako. k-times anonymous authentication (extended abstract). In *ASIACRYPT*. Springer, 2004.
[23] I. Teranishi and K. Sako. k-times anonymous authentication with a constant proving cost. In *9th International Conference on Theory and Practice in Public-Key Cryptography - PKC*. Springer, 2006.
[24] P. P. Tsang and V. K. Wei. Short linkable ring signatures for e-voting, e-cash and attestation. In *ISPEC 05*. Springer, 2005.

Appendix A
Proof of Lemma 10

Lemma 10. **Completeness:** Knowing the solution (x, y, z) of the instance $(T_1 = A^x, T_2 = B^x \cdot g_1^{u \cdot y}, T_3 = C^x \cdot W^{v \cdot y}, T_4 = g_2^z, T_5 = e(W^y, T_4), h = g_1^x, l = g_1^y)$, an honest prover is always able to compute $(\bar{x}, \bar{y}, \bar{z})$ from any commitments $R_0 = g_1^r$; $R_1 = A^r$; $R_2 = B^r$; $R_3 = C^r$; $S_0 = g_1^s$; $S_1 = W^s$; $S_2 = e(W, T_4)^s$ and $Q_0 = g_2^q$ following the protocol. Then the followings equations hold:

$$A^{\bar{x}} = A^r \cdot A^{x \cdot \epsilon} = R_1 \cdot {T_1}^{\epsilon}$$
$$B^{\bar{x}} \cdot g_1^{u \cdot \bar{y}} = B^r \cdot g_1^{s \cdot u} \cdot B^{x \cdot \epsilon} \cdot g_1^{u \cdot y \cdot \epsilon} = R_2 \cdot S_0^u \cdot T_2^{\epsilon}$$
$$C^{\bar{x}} \cdot W^{v \cdot \bar{y}} = C^r \cdot W^{s \cdot v} \cdot C^{x \cdot \epsilon} \cdot W^{v \cdot y \cdot \epsilon} = R_3 \cdot S_1^v \cdot T_3^{\epsilon}$$
$$g_2^{\bar{z}} = g_2^q \cdot g_2^{z \cdot \epsilon} = Q_0 \cdot {T_4}^{\epsilon}$$
$$e(W, T_4)^{\bar{y}} = e(W, T_4)^s \cdot e(W, T_4)^{y \cdot \epsilon} = S_2 \cdot {T_5}^{\epsilon}$$
$$g_1^{\bar{x}} = g_1^r \cdot g_1^{x \cdot \epsilon} = R_0 \cdot h^{\epsilon}; g_1^{\bar{y}} = g_1^s \cdot g_1^{y \cdot \epsilon} = S_0 \cdot l^{\epsilon}$$

Soundness: We consider a prover who is able to respond to two different challenges $\epsilon_0 \xleftarrow{\$} \mathbb{Z}_p^*$ and $\epsilon_1 \xleftarrow{\$} \mathbb{Z}_p^*$ from the same commitment $(R_0, R_1, R_2, R_3, S_0, S_1, S_2, Q_0)$. We then prove that this prover is able to compute a valid witness (x, y, z). Let $(\bar{x}_0, \bar{y}_0, \bar{z}_0)$ (*resp.* $(\bar{x}_1, \bar{y}_1, \bar{z}_1)$) be the response to the challenge ϵ_0 (*resp.* ϵ_1). By hypothesis:

$$A^{\bar{x}_0} = R_1 \cdot {T_1}^{\epsilon_0}; \qquad A^{\bar{x}_1} = R_1 \cdot {T_1}^{\epsilon_1}$$
$$B^{\bar{x}_0} \cdot g_1^{u \cdot \bar{y}_0} = R_2 \cdot S_0^u \cdot T_2^{\epsilon_0}; \qquad B^{\bar{x}_1} \cdot g_1^{u \cdot \bar{y}_1} = R_2 \cdot S_0^u \cdot T_2^{\epsilon_1}$$
$$C^{\bar{x}_0} \cdot W^{v \cdot \bar{y}_0} = R_3 \cdot S_1^v \cdot T_3^{\epsilon_0}; \qquad C^{\bar{x}_1} \cdot W^{v \cdot \bar{y}_1} = R_3 \cdot S_1^v \cdot T_3^{\epsilon_1}$$
$$g_2^{\bar{z}_0} = Q_0 \cdot T_4^{\epsilon_0}; \qquad g_2^{\bar{z}_1} = Q_0 \cdot T_4^{\epsilon_1}$$
$$e(W, T_4)^{\bar{y}_0} = S_2 \cdot T_5^{\epsilon_0}; \qquad e(W, T_4)^{\bar{y}_1} = S_2 \cdot T_5^{\epsilon_1}$$
$$g_1^{\bar{x}_0} = R_0 \cdot h^{\epsilon_0}; \qquad g_1^{\bar{x}_1} = R_0 \cdot h^{\epsilon_1}$$
$$g_1^{\bar{y}_0} = S_0 \cdot l^{\epsilon_0}; \qquad g_1^{\bar{y}_1} = S_0 \cdot l^{\epsilon_1}$$

We prove that $(x, y, z) = (\frac{\bar{x}_1 - \bar{x}_0}{\epsilon_1 - \epsilon_0}, \frac{\bar{y}_1 - \bar{y}_0}{\epsilon_1 - \epsilon_0}, \frac{\bar{z}_1 - \bar{z}_0}{\epsilon_1 - \epsilon_0})$ is a valid witness. We then check that:

$$\begin{aligned}
A^x = &\quad A^{\frac{\bar{x}_1 - \bar{x}_0}{\epsilon_1 - \epsilon_0}} = \left(\frac{R_1 \cdot {T_1}^{\epsilon_1}}{R_1 \cdot {T_1}^{\epsilon_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} = T_1 \\
B^x \cdot g_1^{u \cdot y} = &\quad B^{\frac{\bar{x}_1 - \bar{x}_0}{\epsilon_1 - \epsilon_0}} \cdot g_1^{u \cdot \frac{\bar{y}_1 - \bar{y}_0}{\epsilon_1 - \epsilon_0}} = \left(\frac{B^{\bar{x}_1} \cdot g_1^{u \cdot \bar{y}_1}}{B^{\bar{x}_0} \cdot g_1^{u \cdot \bar{y}_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} \\
= &\quad \left(\frac{R_2 \cdot S_0^u \cdot T_2^{\epsilon_1}}{R_2 \cdot S_0^u \cdot T_2^{\epsilon_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} = T_2 \\
C^x \cdot W^{v \cdot y} = &\quad C^{\frac{\bar{x}_1 - \bar{x}_0}{\epsilon_1 - \epsilon_0}} \cdot W^{v \cdot \frac{\bar{y}_1 - \bar{y}_0}{\epsilon_1 - \epsilon_0}} = \left(\frac{C^{\bar{x}_1} \cdot W^{v \cdot \bar{y}_1}}{C^{\bar{x}_0} \cdot W^{v \cdot \bar{y}_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} \\
= &\quad \left(\frac{R_3 \cdot S_1^v \cdot T_3^{\epsilon_1}}{R_3 \cdot S_1^v \cdot T_3^{\epsilon_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} = T_3 \\
g_2^z = &\quad g_2^{\frac{\bar{z}_1 - \bar{z}_0}{\epsilon_1 - \epsilon_0}} = \left(\frac{Q_0 \cdot T_4^{\epsilon_1}}{Q_0 \cdot T_4^{\epsilon_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} = T_4 \\
e(W^y, T_4) = &\quad e(W, T_4)^{\frac{\bar{y}_1 - \bar{y}_0}{\epsilon_1 - \epsilon_0}} = \left(\frac{S_2 \cdot T_5^{\epsilon_1}}{S_2 \cdot T_5^{\epsilon_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} = T_5 \\
g_1^x = &\quad g_1^{\frac{\bar{x}_1 - \bar{x}_0}{\epsilon_1 - \epsilon_0}} = \left(\frac{R_0 \cdot h^{\epsilon_1}}{R_0 \cdot h^{\epsilon_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} = h \\
g_1^y = &\quad g_1^{\frac{\bar{y}_1 - \bar{y}_0}{\epsilon_1 - \epsilon_0}} = \left(\frac{S_0 \cdot l^{\epsilon_1}}{S_0 \cdot l^{\epsilon_0}}\right)^{\frac{1}{\epsilon_1 - \epsilon_0}} = l
\end{aligned}$$

This concludes the proof.

Honest verifier zero knowledge: We show how to construct a simulator `Sim` that outputs a valid transcript for Π_1 from the same distribution as a real Π_1 transaction. `Sim` picks $\bar{x} \xleftarrow{\$} \mathbb{Z}_p^*$, $\bar{y} \xleftarrow{\$} \mathbb{Z}_p^*$, $\bar{z} \xleftarrow{\$} \mathbb{Z}_p^*$, $\epsilon \xleftarrow{\$} \mathbb{Z}_p^*$ and $S_1 \xleftarrow{\$} \mathbb{G}_1$. It then computes the following values: $S_0 = g_1^{\bar{y}}/l^{\epsilon}$; $S_2 = e(W, T_4)^{\bar{y}}/T_5^{\epsilon}$; $R_0 = g_1^{\bar{x}}/h^{\epsilon}$; $R_1 = A^{\bar{x}}/{T_1}^{\epsilon}$; $R_2 = B^{\bar{x}} \cdot g_1^{u \cdot \bar{y}}/S_0^u \cdot T_2^{\epsilon}$; $R_3 = C^{\bar{x}} \cdot W^{v \cdot \bar{y}}/S_1^v \cdot T_3^{\epsilon}$; $Q_0 = g_2^{\bar{z}}/T_4^{\epsilon}$. Then the transcript $\langle (R_0, R_1, R_2, R_3, S_0, S_1, S_2, Q_0), \epsilon, (\bar{x}, \bar{y}, \bar{z}) \rangle$ is valid for Π_1. Moreover, since $\bar{x}$, $\bar{y}$, $\bar{z}$ and ϵ come from a uniform distribution, then `Sim` outputs valid transcripts from the same distribution as real Π_1 transactions, it concludes the proof. □