Constant Size Ring Signature Without Random Oracle

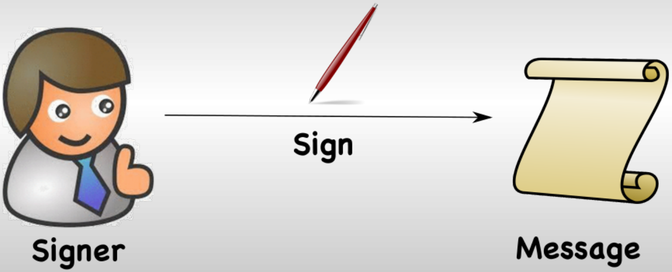
Priyanka Bose Dipanjan Das C. Pandu Rangan

Department of Computer Science and Engineering
Indian Institute of Technology, Madras

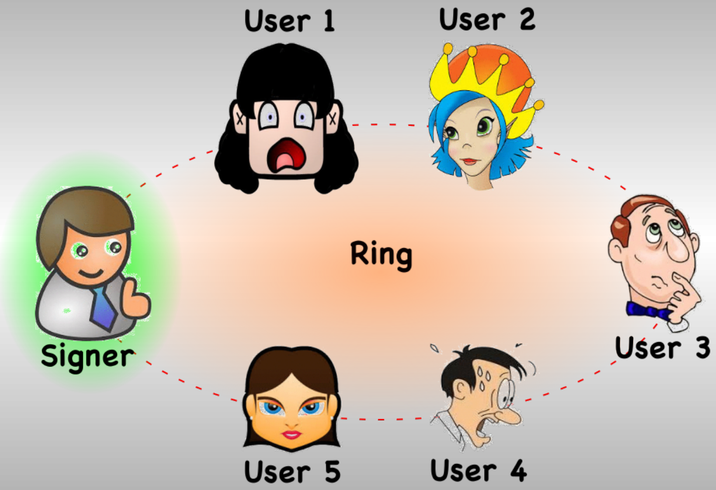
Australausian Conference in Security and Privacy
ACISP 2015

Outline

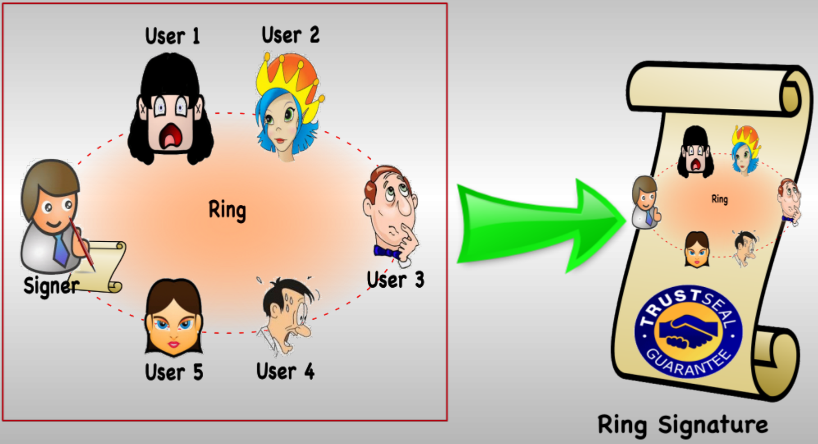
What is a Ring Signature?



What is a Ring Signature?



What is a Ring Signature?



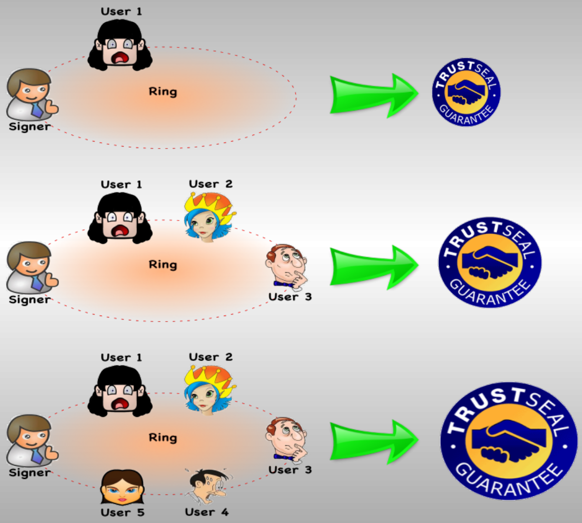
Applications of Ring Signature

- **Anonymous leaking of sensitive secrets:** A company executive wants to divulge sensitive secrets of the company to the public. But, he can't afford to be traced back, either. How can he make people **trust** the message, yet stay **behind** the scene?
- Designated verifier signatures
- **E-voting / E-cash:** A variant of *Ring Signature* known as *Blind Ring Signature* is used.

Algorithm of Ring Signature

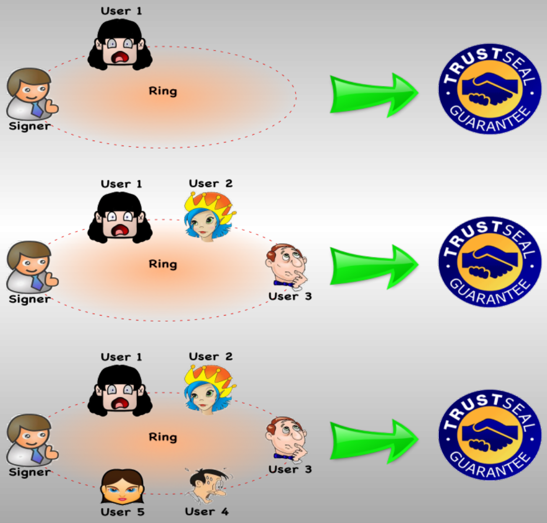
- **Setup:** Run by a `Trusted Third Party` (TTP)
  $rParam \leftarrow \text{RSetup}(1^\kappa)$
- **Key Generation:** Run locally by each of the users
  $(SK, PK) \leftarrow \text{RKeyGen}(rParam)$
- **Signing:** Run by the signer who happens to be one of *those* users
  $\Sigma \leftarrow \text{RSign}(m, SK_s, \mathcal{R})$
- **Verification:** Run by the verifier, can be anybody in practice
  $1/0 \leftarrow \text{RVerify}(m, \Sigma, \mathcal{R})$.

Why Constant size?



Signature size increases proportionally with the size of the ring, typically O(N) or O($\sqrt{N}$)

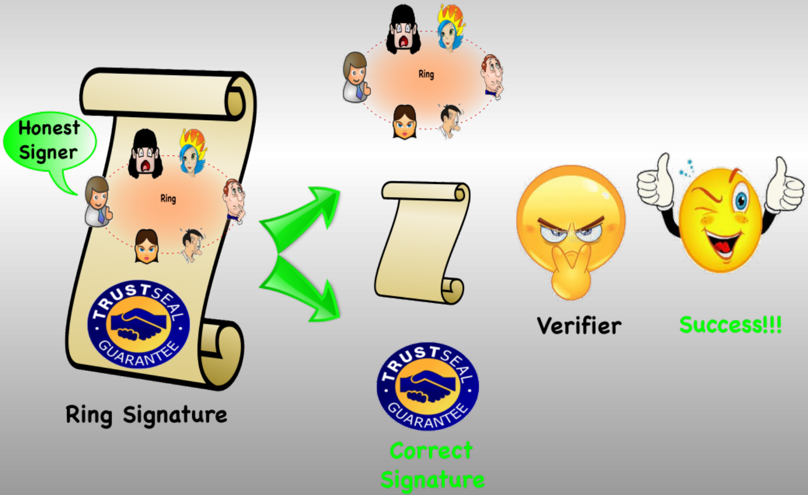
Why Constant size?



Signature size remains constant irrespective of the size of the ring, O(1)

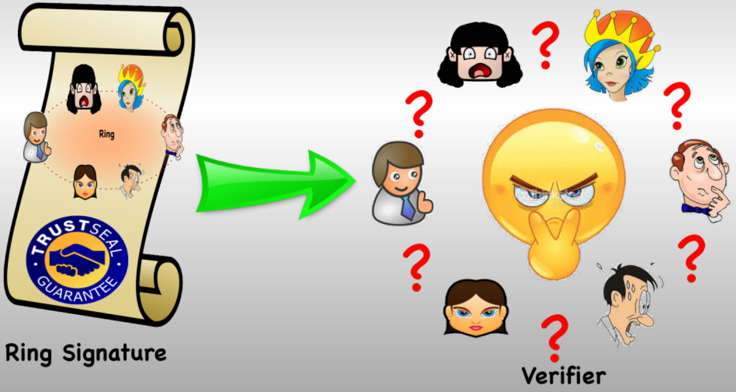
Security of Ring Signatures

- **Correctness**: All honestly generated signatures will be accepted by `RVerify` algorithm.



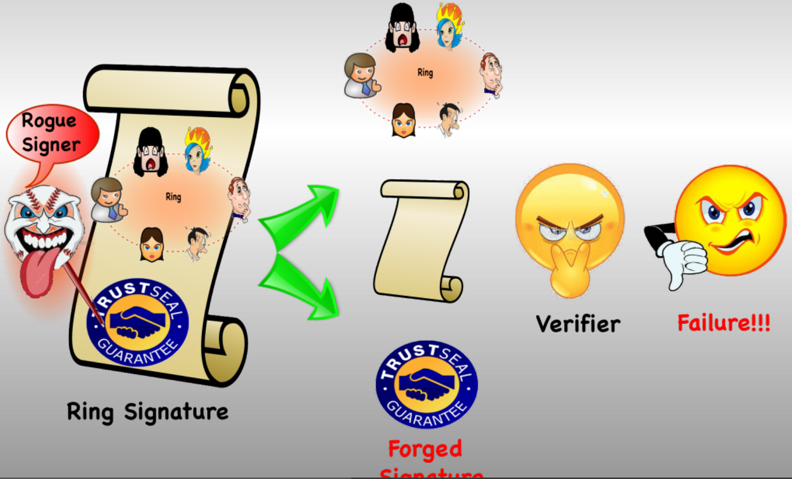
Security of Ring Signatures

- **Anonymity**: An adversary should *not* be able identify which ring member actually signed the document.



Security of Ring Signatures

- **Unforgeability**: An adversary can not output a valid signature Σ^* on message m^* with respect to a ring $\mathcal{R}^*$, unless the adversary obtained it by querying sign oracle on $(m^*, \mathcal{R}^*)$



Related Work

Signature size := $O(n)$

- Rivest *et al.* [RST06]
- Abe *et al.* [AOS02]
- Boneh *et al.* [BGLS03]
- Herranz *et al.* [HS03]
- Bender *et al.* [BKM06]
- Chow *et al.* [CWLY06]
- Shacham *et al.* [SW07]
- Boyen [Boy07]
- Schage *et al.* [SS10]
- Brakerski *et al.* [BK10]

Signature size := $O(\sqrt{n})$

- Chandran *et al.* [CGS07]
- Yuen *et al.* [YHJASZ12]
- Ghadafi [Gha13]

Signature size := $O(1)$

- Dodis *et al.* [DKNS04]

Our Contribution

Our construction achieves the following desirable properties:

- Constant size membership proof technique
- Constant size ring signature
- Provably secure without using random oracle
- Anonymity under full key exposure
- Unforgeability against insider-corruption

Bilinear Pairing

Bilinear pairing $\mathcal{G} = (n, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ where $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$ and $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ having the following properties:

- ***Bilinearity***: For $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_n$ the following holds true: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
- ***Non-degeneracy***: For any $\mathcal{X} \in \mathbb{G}_1$ and $\mathcal{Y} \in \mathbb{G}_2$, if $e(\mathcal{X}, \mathcal{Y}) = 1_T$, the identity element of $\mathbb{G}_T$, then either $\mathcal{X}$ is the identity of $\mathbb{G}_1$ or $\mathcal{Y}$ is the identity of $\mathbb{G}_2$, but not both.
- ***Efficiently Computable***: The map e should be efficiently computable.

Type-3: [GPS08] $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable isomorphism are known to exist between $\mathbb{G}_1$ and $\mathbb{G}_2$.

We will use asymmetric pairing over groups of composite order which can be shown to be generated efficiently in [MS13].

Hardness Assumptions

Definition: Decisional Diffie-Hellman Assumption (DDH) [NR97]

Given a cyclic group $\mathbb{G} = \langle g \rangle$, a tuple $\langle g, g^a, g^b, g^{ab}, g^c \rangle$ where $a, b, c \in_R \mathbb{Z}_n$ and for all PPT adversaries $\mathcal{A}_{DDH}$, the probability

$$|Pr[\mathcal{A}_{DDH}(g, g^a, g^b, g^{ab}) = 1] - Pr[\mathcal{A}_{DDH}(g, g^a, g^b, g^c) = 1]| < \nu(\kappa)$$

Definition: Symmetric External Diffie-Hellman Assumption (SXDH) [NR97]

DDH holds in both groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

Hardness Assumptions

Definition: Decisional Linear Assumption (DLIN) [BBS04]

For Type-1 bilinear groups where $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G} = \langle g \rangle$, given $\langle g^a, g^b, g^{ra}, g^{sb}, g^t \rangle$ and $a, b, s, r, t \in \mathbb{Z}_p$ being unknown, it is hard to tell whether $t = r + s$ or t is random.

Definition: q-Strong Diffie-Hellman Assumption (q-SDH) [BB08]

Let $\alpha \in_R \mathbb{Z}_p$. Given a $(q+1)$-tuple $\langle g, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^q} \rangle \in \mathbb{G}^{q+1}$ as input, for every adversary $\mathcal{A}_{q\text{-}SDH}$, the probability

$$Pr[\mathcal{A}_{q\text{-}SDH}(g, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^q}) = \langle c, g^{\frac{1}{\alpha+c}} \rangle] < \nu(\kappa)$$

for any value of $c \in \mathbb{Z}_p \backslash \{-\alpha\}$. Though naturally q-type assumptions are defined on prime order groups, it has been shown in [CM14] that all q-type assumptions can also be proven to be secure in composite order groups provided subgroup hiding assumption (SGH) holds.

Definition: Subgroup Hiding Assumption (SGH) [BGN05]

Given a generation algorithm $\mathcal{G}$, which takes security parameter κ as input and gives output a tuple $\langle \mathbb{G}, \mathbb{G}_T, e, sk \rangle$, where $sk = (p, q)$ such that $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ and $\mathbb{G}$ and $\mathbb{G}_T$ are both groups of order $n = pq$, it is computationally in feasible to distinguish between an element of $\mathbb{G}$ and an element of $\mathbb{G}_p$. Formally, for all PPT adversaries $\mathcal{A}_{SGH}$, the probability

$$|Pr[(sk, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\kappa); n = pq; sk = (p, q); x \leftarrow \mathbb{G} : \mathcal{A}_{SGH}(n, \mathbb{G}, \mathbb{G}_T, e, x) = 0] \; - \; Pr[(sk, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\kappa); n = pq; sk = (p, q); x \leftarrow \mathbb{G} : \mathcal{A}_{SGH}(n, \mathbb{G}, \mathbb{G}_T, e, x^q) = 0]| < \nu(\kappa)$$

where $\mathcal{A}_{SGH}$ outputs 1 if it believes $x \in \mathbb{G}_p$ and 0 otherwise. **SGH** being hard in asymmetric pairing over composite order groups means, it is hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

Hardness Assumptions

Definition: Square Root Modulo Composite (SQROOT) [MVSOP96]

Given a composite integer n and $a \in Q_n$(the set of quadratic residues modulo n), it is computationally hard to find a square root of $a \mod n$; that is an integer x such that $x^2 \equiv a \pmod{n}$, where $n = pq$, product of two safe primes.

Groth-Sahai Proofs

Q: What is this all about?

A Non-Interactive (NI), Zero-Knowledge (ZK) proof system for equations over bilinear groups.

Q: What all are the parties involved?

A `Trusted Third Party` (TTP), a prover and a verifier.

Q: What role does the TTP play?

Generates `Common Reference String` (CRS) to be shared between prover and verifier.

Q: What is to be proven and verified by the prover and the verifier?

The knowledge of *some* solution of a set of equations without revealing(prover) / knowing(verifier) the solution (also called *witness*) itself.

Groth-Sahai Proofs

$\mathcal{X}_1, \ldots, \mathcal{X}_m \in \mathbb{G}_1, \mathcal{Y}_1, \ldots, \mathcal{Y}_n \in \mathbb{G}_2, x_1, \ldots, x_{m'} \in \mathbb{Z}_n$ and $y_1, \ldots, y_{n'} \in \mathbb{Z}_n$ are variables.

Pairing product equation:

$$\prod_{i=1}^{n} e(\mathcal{A}_i, \underline{\mathcal{Y}}_i) \cdot \prod_{i=1}^{m} e(\underline{\mathcal{X}}_i, \mathcal{B}_i) \cdot \prod_{i=1}^{m} \prod_{j=1}^{n} e(\underline{\mathcal{X}}_i, \underline{\mathcal{Y}}_j)^{\gamma_{ij}} = t_T$$

For constants $\mathcal{A}_i \in \mathbb{G}_1, \mathcal{B}_i \in \mathbb{G}_2, t_T \in \mathbb{G}_T, \gamma_{ij} \in \mathbb{Z}_n$

Multi-scalar multiplication equation in $\mathbb{G}_1$:

$$\sum_{i=1}^{n'} \underline{y}_i \mathcal{A}_i + \sum_{i=1}^{m} b_i \underline{\mathcal{X}}_i + \sum_{i=1}^{m} \sum_{j=1}^{n'} \gamma_{ij} \underline{y}_j \underline{\mathcal{X}}_i = T_1$$

For constants $\mathcal{A}_i, T_1 \in \mathbb{G}_1$ and $b_i, \gamma_{ij} \in \mathbb{Z}_n$

Groth-Sahai Proofs

Multi-scalar multiplication equation in $\mathbb{G}_2$:

$$\sum_{i=1}^{n} a_i \underline{\mathcal{Y}_i} + \sum_{i=1}^{m'} \underline{x_i} \mathcal{B}_i + \sum_{i=1}^{m'} \sum_{j=1}^{n} \gamma_{ij} \underline{x_i} \underline{\mathcal{Y}_j} = T_2$$

For constants $\mathcal{B}_i, T_2 \in \mathbb{G}_2$ and $a_i, \gamma_{ij} \in \mathbb{Z}_n$

Quadratic equation in $\mathbb{Z}_n$:

$$\sum_{i=1}^{n'} a_i \underline{y_i} + \sum_{i=1}^{m'} \underline{x_i} b_i + \sum_{i=1}^{m'} \sum_{j=1}^{n'} \gamma_{ij} \underline{x_i} \underline{y_j} = t$$

For constants $a_i, b_i, \gamma_{ij}, t \in \mathbb{Z}_n$ For clarity we will underline the elements of the witness in the description of equations.

Constant Size Set Membership Proof

Q: What is this all about?

A Non-Interactive (NI), Zero-Knowledge (ZK) proof technique.

Q: What all are the parties involved?

A `Trusted Third Party` (TTP), a prover and a verifier.

Q: What role does the TTP play?

Generates CRS to initialize GS protocol and a q-SDH instance to be shared between prover and verifier.

Q: What is to be proven and verified by the prover and the verifier?

The containment of an integer α_δ in a *public* set $S = \{\alpha_1, \alpha_2, ..., \alpha_\delta, ...\alpha_N\} \in \mathbb{Z}_n^N$ without revealing the integer itself.

Constant Size Set Membership Proof - Idea

Outline of the idea

- **MemSetup:** Establishes a trusted set-up.
- **MemWitness:** Prover forms a polynomial $F(x)$ having the set elements $\alpha_\delta \in S$ as its roots. Further, $\psi(x)$ is computed based on `Little-Bezout` theorem. Witness is $w = g_1^{\psi(\beta)}$.
- **MemProve:** Prover produces the GS proof ϕ_{mem} of the verification equation as the proof of set membership.
- **MemVerify:** Verifier convinces himself by running GS verification algorithm on the equation above and GS proof ϕ_{mem}.

Constant Size Set Membership Proof - Algorithm

- **MemSetup**$(1^\kappa, q)$**:** Run by a `Trusted Third Party` (TTP)
  - Generate the definition of a bilinear group $\mathcal{G}$ parameterized by security parameter κ
  - Generate a CRS *crs* to initialize GS protocol
  - Choose a secret key $\beta \in_R \mathbb{Z}_n^*$
  - Generate a q-SDH instance $qSDH = \langle g_1, g_1^{\beta}, g_1^{\beta^2}, ..., g_1^{\beta^q} \rangle \in \mathbb{G}_1^{q+1}$ to inject the hard problem
  - Publish public parameters $mParam = \langle \mathcal{G}, crs, qSDH, g_2^{\beta} \rangle$
- **MemWitness**$(mParam, \alpha_\delta, S)$**:** Run by the prover
  - Compute the polynomial $F(x) = \prod_{i=1}^{|S|}(x - \alpha_i)$
  - Compute the polynomial $\psi(x) = \frac{F(x)}{(x - \alpha_\delta)}$
  - Compute $w = g_1^{\psi(\beta)}$
  - Compute $D = g_2^{\alpha_\delta}$
  - Output the tuple $W = \langle \alpha_\delta, w, D \rangle$ as witness.

Constant Size Set Membership Proof - Algorithm

- **MemProve**$(mParam, S, W)$: Run by the prover
  - Compute $C = g_1^{F(\beta)} = \prod_{i=0}^{|S|}(g_1^{\beta^i})^{F_i}$
  - Compute $t = e(C, g_2)$
  - Compute the membership proof $\phi_{mem} = \langle\{\Upsilon_w, \Upsilon_{\alpha_\delta}, \Upsilon_D\}, \vec{\Gamma}_{mem}\rangle$

    $$\phi_{mem} \leftarrow \texttt{GSProve}\{\mathcal{G}, crs, \{e(\underline{w}, g_2^{\beta}/\underline{D}) = t \wedge \underline{D} = g_2^{\underline{\alpha_\delta}}\}, (\alpha_\delta, w, D)\}$$

  - Send the proof ϕ_{mem} to the verifier.
- **MemVerify**$(mParam, S, \phi_{mem})$: Run by the verifier
  - Compute $F(x)$, C and t
  - $c \leftarrow \texttt{GSVerify}\{\mathcal{G}, crs, \{e(\underline{w}, g_2^{\beta}/\underline{D}) = t \wedge \underline{D} = g_2^{\underline{\alpha_\delta}}\}, \phi_{mem}\}$
  - Announce 'Success' if $c = 1$, 'Failure' otherwise

Constant Size Set Membership Proof - Security

The set membership proof technique is:

- **Correct:** If GS proof is *complete*.
- **Perfectly-Sound:** If GS proof is *perfectly sound* and q-SDH assumption holds in bilinear group $\mathbb{G}_1 \in \mathcal{G}$.
- **Zero-Knowledge:** If GS proof ϕ_{mem} is *zero-knowledge*.

Constant Size Ring Signature

- The root of inefficiency of all earlier ring signature construction lies in the fact that the size of the proof of *ring containment* of signer's public key were either linear or sub-linear.
- Our construction of *Constant Size Ring Signature* can be viewed as an application of our *Constant Size Set Membership Proof* technique.
- Our paper provides a generic technique to construct a ring signature scheme on top of any *compatible* signature scheme and a concrete instantiation of ring signature scheme based on `Full Boneh-Boyen` (FBB) signature scheme.
- Generic technique is fairly involved to present in due time. Hence, we will *only* talk about the later one in subsequent slides.

Constant Size Ring Signature - Idea

Outline of the idea

- **RSetup:** Establishes a trusted set-up.
- **RKeyGen:** Each user of the system locally runs the key generation algorithm `SKeyGen` of the underlying signature scheme. For each component $sk_i \in \mathbb{Z}_n$ of the secret key, we augment the public with components $q_i = sk_i^2 (mod\ n)$
- **RSign:** Signer signs on a combined hash of message and ring, produces GS proofs of signature verification equation, ring containment and proofs of correlation.
- **RVerify:** Verifier convinces himself by running GS verification algorithm on the equations above and corresponding GS proofs.

Constant Size Ring Signature - Algorithm

- **RSetup**$(1^\kappa, q)$**:** Run by a `Trusted Third Party` (TTP)
  - $mParam \leftarrow \texttt{MemSetup}(1^\kappa, q)$. $\langle \mathcal{G}, crs, qSDH, g_2^\beta \rangle \leftarrow mParam$. $\langle n, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2 \rangle \leftarrow \mathcal{G}$. Groups of of composite order $n = p.q$ (large primes). q-SDH assumption holds in $\mathbb{G}_1$.
  - $\mathcal{H} : \{0,1\}^* \rightarrow \mathcal{M}$
  - Publish public parameters $rParam = \langle mParam, \mathcal{H} \rangle$
- **RKeyGen**$(rParam)$**:** Run locally by each user
  - FBB secret key $SK_i = \langle a_i, b_i \rangle \in_R \mathbb{Z}_n^2, i \in \mathcal{R}$
  - FBB public key $PK_i = \langle A_i, B_i \rangle = \langle g_2^{a_i}, g_2^{b_i} \rangle$
  - $q_{ia} = a_i^2 \pmod n$ and $q_{ib} = b_i^2 \pmod n$
  - Extended public key $PK_i' = \langle PK_i, q_{ia}, q_{ib} \rangle$
  - Publish extended public keys $\{PK_i'\}$ to the world.

Constant Size Ring Signature - Algorithm

- **RSign**$(m, SK_s, rParam)$: Run by the signer
  - $m' \leftarrow \mathcal{H}(m||\mathcal{R}), m \in \{0,1\}^*$
  - $\Delta \leftarrow$ FBB signature
  - GS proofs (signature): $\phi_{sig} \leftarrow \texttt{GSProve}(\cdot, \cdot, \{\texttt{VE}\}, \cdot)$
  - Witnesses $W_a \leftarrow \texttt{MemWitness}(mParam, \underline{q_{sa}}, \mathcal{R}_a)$
  - GS proofs(Membership): $\phi_{mem_a} \leftarrow \texttt{MemProve}(mParam, \mathcal{R}_a, W_a)$
  - GS proofs(Correlation): $\phi_{q_a} \leftarrow \texttt{GSProve}(\cdot, \cdot, \{\underline{q_{sa}} = \underline{a}^2\}, (q_{sa}, a))$
  - GS proofs(Correlation): $\phi_{pk_A} \leftarrow \texttt{GSProve}(\cdot, \cdot, \{\underline{A} = g_2^{\underline{a}}\}, (A, a))$
  - Publish message m, ring information $\mathcal{R}$ and ring signature $\Sigma \leftarrow \langle \phi_{sig}, \phi_{mem}, \phi_q, \phi_{pk}, \Delta \backslash \Delta' \rangle$
- **RVerify**$(rParam)$: Run by the verifier
  - Verify the consistency of the ring signature by running `GSVerify()` on $\phi_{sig}, \phi_{mem}, \phi_q, \phi_{pk}$ respectively.
  - 'Success' if all of the above verification passes, 'Failure' otherwise

Constant Size Ring Signature - Highlights

- We use the same randomness throughout in forming GS proofs
- Public key $PK_s \xrightarrow{\phi_{pk}} SK_s \xrightarrow{\phi_q} q_s \xrightarrow{\phi_{mem}}$ ring $\mathcal{R}$

Constant Size Ring Signature - Security

The ring signature construction is:

- **Correct:** If GS proof system is *perfectly complete* and underlying signature scheme $\mathtt{Sig}$ is *correct*.
- **Anonymous under Full Key Exposure:** If GS proof system is *hiding* (*i.e.* witness-indistinguishable/zero-knowledge).
- **Unforgeable in the Presence of Insider Corruption:** If GS proof system is *perfectly sound*, the hash function $\mathcal{H}$ is *collision-resistant*, and the signature scheme $\mathtt{Sig}$ is *existentially unforgeable against adaptive chosen-message attack*.

Efficieny Comparison

Table: Cost of Signature Instantiations

Instantiations	Setting	Signature Size	Complexity Assumptions
Our Scheme	Type - 1	-	-
	Type - 2	-	-
	Type - 3	$\mathbb{G}_1^{50} + \mathbb{G}_2^{42} + \mathbb{Z}_p^3$	q-SDH + SXDH
Ghadafi [Gha13]	Type - 1	$\mathbb{G}^{42n+39} + \mathbb{Z}_p^4$	CDH + DLIN
	Type - 2	$\mathbb{G}_1^{20n+14} + \mathbb{G}_2^{30n+21} + \mathbb{Z}_p^4$	q-SDH + $\text{DDH}_{\mathbb{G}_1}$ + $\text{DLIN}_{\mathbb{G}_2}$
	Type - 3	$\mathbb{G}_1^{20n+14} + \mathbb{G}_2^{20n+14} + \mathbb{Z}_p^3$	q-SDH + SXDH

Open Probelms

- Our construction requires sharing of a `Common Reference String` (CRS) generated by a `Trusted Third Party` (TTP). Construction of a scheme in Standard Model, based on simpler number theoretic assumptions can be an interesting direction for further research.

The End

Questions?