Monero's Building Blocks Part 7 of 10 – *Multilayered Linkable Spontaneous Anonymous Group (MLSAG) signature scheme*

Bassam El Khoury Seguias

BTC: 3FcVvBZwTUkUrcqJd16RcjR42qT2tDWHWn

ETH: 0xb79Fb9194C8Cc6221368bb70976e18609Ab9AcA8

April 16, 2018

1 Introduction

Monero stands out from other cryptocurrencies in its ability to hide the signer, conceal the transaction amount, and protect the identity of the recepient. Parts 1 to 6 helped us build the foundation to better understand and appreciate the security properties of ring signatures (albeit in the RO model). Parts 7, 8, and 9 will focus on Monero's privacy in so far as the signer's identity and the transaction amount are concerned. Part 10 will introduce stealth addressing as a mechanism to protect the identity of the fund's recepient.

In order to describe how a Monero transaction hides both the signer's identity and the amount of the transaction, we introduce 2 additional concepts:

1. A generalization of the LSAG signature scheme (introduced in part 6) so that each member of the ring can have a key-pair vector $[(pk_1, sk_1), ..., (pk_n, sk_n)]$ instead of only one pair (pk, sk)
2. A particular map known as the *Pedersen Commitment* that will be used to hide transaction amounts while allowing the network to check that input and output amounts always balance out.

Recall that by proving that a digital signature scheme was unforgeable, one gets the assurance that only the signing algorithm Σ associated with a given ring member can produce a valid signature (i.e., verified by $\mathcal{V}$). Any other procedure that bypasses Σ will result in a failed attempt of forgery with overwhelming probability. We note the following about the verification process of $\mathcal{V}$:

- In a "non-ring" setting, the verification is done using a **particular public key** pk_π. The validation of a given signature proves that the signer of the message (in this case user π) knows the secret key sk_π associated with pk_π. Assuming that secret keys are safe-guarded and non-compromised, this actually proves that the user with key-pair (pk_π, sk_π) signed the message.

- In a ring setting, the verification is conducted using a **public key vector** $L \equiv [pk_1, ..., pk_\pi, ..., pk_n]$ known as a ring. This vector is used to conceal the identity of the signer. The validation of a given signature proves that the the signer of the message (in this case user π) knows the secret key associated with one of the public keys in L. Assuming that secret keys are safe-guarded and non-compromised, this actually proves that the user with key-pair (pk_π, sk_π) signed the message, for some index $1 \leq \pi \leq n$ that no one other then the actual signer knows.

- The ring setting can be generalized further by allowing each ring member $i, 1 \leq i \leq n$ to have a key-pair vector of length m, given by $[(pk_i^1, sk_i^1), ..., (pk_i^m, sk_i^m)]$, as opposed to a unique key pair (pk_i, sk_i). In this setting, the verification is conducted using a **public key matrix**

$$PK = \begin{bmatrix} pk_1^1 & ... & pk_\pi^1 & ... & pk_n^1 \\ ... & ... & ... & ... & ... \\ pk_1^m & ... & pk_\pi^m & ... & pk_n^m \end{bmatrix}$$

  The validation of the signature proves that the signer knows the secret key associated with each one of its public keys. In other terms, there exists a column in PK (say column $1 \leq \pi \leq n$) such that the signer knows the secret key associated with each public key appearing in that column. Assuming that secret keys are safe-guarded and non-compromised, this actually proves that the user with key-pair vector $[(pk_\pi^1, sk_\pi^1), ..., (pk_\pi^m, sk_\pi^m)]$ signed the message, for some $1 \leq \pi \leq n$ that no one other then the actual signer knows.

2 The MLSAG scheme

The MLSAG signature scheme is a generalization of the LSAG scheme encountered earlier in part 6. It was introduced by Shen Noether in his 2016 paper entitled "Ring Confidential Transactions" [3]. MLSAG security analysis closely mirrors that of the LSAG scheme. Although the security proofs are similar, we will go over them again in detail to highlight the nuances pertaining to the generalization.

Similar to the LSAG version introduced in part 6, MLSAG is built on a group E of prime order q and uses 2 statistically independent ROs:

- $\mathcal{H}_1 : \{0,1\}^* \longrightarrow \mathbb{F}_q$
- $\mathcal{H}_2 : \{0,1\}^* \longrightarrow E$

We carry forward all the notation used in the Cryptonote scheme and the LSAG scheme. In particular, we let E be a large finite group generated by the same elliptic curve introduced in part 5. The curve's equation is given by:

$$E : -x^2 + y^2 = 1 + dx^2y^2$$

For completeness purposes, we recall that the above equation is a polynomial over $\mathbb{F}_q$ where q is a very large prime and d is a pre-defined element of $\mathbb{F}_q$. We simplify the notation and refer to the group generated by this elliptic curve as $E(\mathbb{F}_q)$ (refer to the post entitled *Elliptic Curve Groups* for an introduction to this topic). We also observe the following:

- Elements of $E(\mathbb{F}_q)$ are pairs $(x, y) \in \mathbb{F}_q^2$ that satisfy the above equation.
- Elliptic curve groups in general and $E(\mathbb{F}_q)$ in particular have a well defined addition operation that we denote by $\oplus$.
- $E(\mathbb{F}_q)$ contains a special element G (not necessarily unique) that we refer to as the base point. The base point has order $l < q$, where l is a very large prime. That means that adding G to itself l times yields the identity element e of $E(\mathbb{F}_q)$. In other terms, $G \oplus ... \oplus G = e$. We simply write $l \otimes G = e$ (the notation $\otimes$ serves as a reminder that this is scalar multiplication associated with $\oplus$).
- We let $\{G\}$ denote the group generated by G under the $\oplus$ operation of $E(\mathbb{F}_q)$. We also let $\{G\}^* \equiv \{G\} - e$.
- Solving the Discrete Logarithm (DL) problem on $\{G\}^*$ (and more generally on $E(\mathbb{F}_q)$) is thought to be intractable.

With a slight divergence from [3], we first introduce a hash function $\mathcal{H}_T$ before we define $\mathcal{H}_2$. The reason is the same as the one we previously articulated in parts 5 and 6 and will be highlighted in section 4 when we build the signing simulator to prove MLSAG's resilience against EFACM.

- $\mathcal{H}_1 : \{0, 1\}^* \longrightarrow \mathbb{F}_q$
- $\mathcal{H}_T : \{G\}^* \longrightarrow \mathbb{F}_l^* \times \{G\}^*$

  $\mathcal{H}_T$ takes an element $s \in \{G\}^*$ and outputs a tuple $(v_s, v_s \otimes G) \in \mathbb{F}_l^* \times \{G\}^*$. Here v_s is a random element chosen according to a uniform distribution over $\mathbb{F}_l^*$. We then let $\mathcal{H}_2(s) \equiv v_s \otimes G$. So $\mathcal{H}_2 : \{G\}^* \longrightarrow \{G\}^*$, takes an element $s \in \{G\}^*$ and returns an element $v_s \otimes G \in \{G\}^*$ where v_s is randomly chosen in $\mathbb{F}_l^*$.

  Note that [3] defines $\mathcal{H}_2$ as a map from $\{0, 1\}^*$ to $E \equiv EC(\mathbb{F}_q)$. Here we restricted the domain and the range to $\{G\}^*$ instead. This is because in our exposition, $\mathcal{H}_2$ is strictly applied to public keys as opposed to any element of $\{0, 1\}^*$. Public keys are elements of $E(\mathbb{F}_q)$ that are scalar multiples of the base point G. Moreover, the scalar is never equal to order$(G) = l$ (we impose this constraint when we introduce the key generation algorithm $\mathcal{G}$). We are then justified in restricting the domain to $\{G\}^*$. The range is arbitrarily defined to be $\{G\}^*$, which is permissible since it preserves the injective nature of the map.

The MLSAG scheme is defined by a set of 4 algorithms:

- **The key generation algorithm** $\mathcal{G}$. On input 1^k (k is the security parameter that by design we request to satisfy $k < log_2|\{G\}^*| = log_2(l - 1)$), it produces a key-pair vector $[(sk^1, pk^1), ..., (sk^m, pk^m)] \equiv [(x^1, y^1), ..., (x^m, y^m)]$ of matching secret and public keys. $\forall j \in \{1, ..., m\}, x^j$ is randomly chosen in $\mathbb{F}_l^* \equiv \{1, ..., l - 1\}$, and y^j is calculated as $x^j \otimes G$. (Note that G and y^j are both elements of $\{G\}^* \subset EC(\mathbb{F}_q)$ while x^j is an element of $\mathbb{F}_l^* \subset \mathbb{F}_q$.

In addition to the $[(x^1, y^1), ..., (x^m, y^m)]$ key-pair vector, $\mathcal{G}$ computes $I^j \equiv x^j \otimes \mathcal{H}_2(y^j)$, $\forall j \in \{1, ..., m\}$. $I \equiv [I^1, ..., I^j]$ is known as the *key image vector* (or *tag vector*). It is signer-specific since it depends only on the signer's private and public keys. It allows the ring linkability algorithm $\mathcal{L}$ to test for independence between different signatures. $\mathcal{G}$ is modeled as a PPT Turing machine.

Suppose we have a ring of n members, each with m keys as described above. The objective of the MLSAG scheme is two-fold:

1. To demonstrate that one of the signers knows all the secret keys associated with her key-pair vector.
2. To ensure that if the signer uses at least one of their m secret keys in a different signature, then the 2 signatures will be flagged as linked and proper measures taken.

- **The ring signing algorithm** Σ. Suppose a user A_π decides to sign a message m on behalf of the ring $\{A_1, ., A_n\} \ni A_\pi$. A_π has key-pair vector $[(x_\pi^1, y_\pi^1), ., (x_\pi^m, y_\pi^m)]$ and key-image vector $I_\pi \equiv [(x_\pi^1 \otimes \mathcal{H}_2(y_\pi^1)), ..., (x_\pi^m \otimes \mathcal{H}_2(y_\pi^m))]$. Moreover, the public key matrix associated with the ring of users is given by:

$$PK = \begin{bmatrix} pk_1^1 & ... & pk_\pi^1 & ... & pk_n^1 \\ ... & ... & ... & ... & ... \\ pk_1^m & ... & pk_\pi^m & ... & pk_n^m \end{bmatrix}$$

Σ does the following:

1. $\forall j \in \{1, ..., m\}$, choose random $q_\pi^j \in \{1, ..., l\} \equiv \mathbb{F}_l$. Assign:

   $\{\ L_\pi^j \equiv (q_\pi^j \otimes G)$

   $\{\ R_\pi^j \equiv (q_\pi^j \otimes \mathcal{H}_2(y_\pi^j))$

2. $c_{\pi+1} \equiv \mathcal{H}_1(m, L_\pi^1, R_\pi^1, ..., L_\pi^m, R_\pi^m) \pmod{l}$

3. $\forall i \in \{\pi + 1, .., n, 1, .., \pi - 1\}$:

   $\{\ \forall j \in \{1, ..., m\}$, choose random $r_i^j \in \{1, ...l\} \equiv \mathbb{F}_l$. Assign:

   $\{\ L_i^j \equiv (r_i^j \otimes G)\ \oplus\ (c_i \otimes y_i^j)$

   $\{\ R_i^j \equiv (r_i^j \otimes \mathcal{H}_2(y_i^j))\ \oplus\ (c_i \otimes I_\pi^j)$

   $\{\ c_{i+1} \equiv \mathcal{H}_1(m, L_i^1, R_i^1, ..., L_i^m, R_i^m) \pmod{l}$, where $c_1 \equiv c_{n+1}$

4. $\forall i \in \{\pi + 1, .., n, 1, .., \pi - 1\}$, set $r_\pi^j \equiv q_\pi^j - c_\pi x_\pi^j \pmod{l}$. Here $c_\pi x_\pi^j$ denotes regular scalar multiplication in modulo l arithmetic.

Σ outputs a signature $\sigma_\pi(m, PK) \equiv (I_\pi^1, ..., I_\pi^m, c_1, r_1^1, ..., r_1^m, ..., r_n^1, ..., r_n^m)$. Σ is a PPT algorithm. We note that one drawback of the MLSAG scheme is the size of the signature. An increase of the ring size by one unit corresponds to an increase of m units in the size of the signature. This constraint limits the usage of large ring sizes in practice.

- **The ring verification algorithm $\mathcal{V}$.** Given a message m, a pubic key matrix of the ring members given by:

$$PK = \begin{bmatrix} pk_1^1 & ... & pk_\pi^1 & ... & pk_n^1 \\ ... & ... & ... & ... & ... \\ pk_1^m & ... & pk_\pi^m & ... & pk_n^m \end{bmatrix}$$

and a signature $\sigma(m, PK) \equiv (I^1, ..., I^m, c_1, r_1^1, ..., r_1^m, ..., r_n^1, ..., r_n^m)$:

```
{ Let $c_1' = c_1$
{ $\forall i \in \{1, .., n\}$:
    { $\forall j \in \{1, ..., m\}$, compute:
        { $(L_i^j)' \equiv (r_i^j \otimes G) \ \oplus \ (c_i \otimes y_i^j)$
        { $(R_i^j)' \equiv (r_i^j \otimes \mathcal{H}_2(y_i^j)) \ \oplus \ (c_i \otimes I^j)$
    { $c_{i+1}' \equiv \mathcal{H}_1(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)') \pmod{l}$
{ $\mathcal{V}$ checks if $c_1 = c_{n+1}'$ ($c_{n+1}' \equiv \mathcal{H}_1(m, (L_n^1)', (R_n^1)', ., (L_n^m)', (R_n^m)') \pmod{l}$)
```

If equality holds, the signature is valid and $\mathcal{V}$ outputs *True.* Otherwise, it outputs *False.* $\mathcal{V}$ is a deterministic algorithm.

- **The ring linkability algorithm $\mathcal{L}$.** It takes a $\mathcal{V}$-verified valid signature $\sigma(m, PK)$. It checks if any component of the key-image vector I was used in the past by comparing it to previous key-image vector components stored in a set $\mathcal{I}$. If a match is found, then with overwhelming probability the 2 signatures were produced by the same key-pair vector (as will be justified when we prove the *exculpability* of MLSAG in section 5 below), and $\mathcal{L}$ outputs *Linked.* Otherwise, its key-image vector is added to $\mathcal{I}$ and $\mathcal{L}$ outputs *Independent.*

3 Security analysis - Correctness

Let $\sigma_\pi(m, PK) \equiv (I_\pi^1, ..., I_\pi^m, c_1, r_1^1, ..r_1^m, ...r_n^1, ..., r_n^m)$ be a Σ-generated signature. Without loss of generality, assume $1 < \pi \leq n$. Then $\forall i, 1 \leq i < \pi$, we have the following implication:

If $\{(c_i' = c_i) \ \cap \ ((L_i^j)' = L_i^j, \ \forall j \in \{1, ..., m\}) \ \cap \ ((R_i^j)' = R_i^j, \ \forall j \in \{1, ..., m\})\}$, then:

{ $c_{i+1}' = \mathcal{H}_1(m, (L_i^1)', (R_i^1)', ., (L_i^m)', (R_i^m)') \pmod{l} = \mathcal{H}_1(m, L_i^1, R_i^1, ., L_i^m, R_i^m) \pmod{l} = c_{i+1}$

and $\forall j \in \{1,.,m\}$:

$\{\ (L^j_{i+1})' \equiv (r^j_{i+1} \otimes G) \oplus (c'_{i+1} \otimes y^j_{i+1}) = (r^j_{i+1} \otimes G) \oplus (c_{i+1} \otimes y^j_{i+1}) = L^j_{i+1}$

$\{\ (R^j_{i+1})' \equiv (r^j_{i+1} \otimes \mathcal{H}_2(y^j_{i+1})) \oplus (c'_{i+1} \otimes I^j_\pi) = (r^j_{i+1} \otimes \mathcal{H}_2(y^j_{i+1})) \oplus (c_{i+1} \otimes I^j_\pi) = R^j_{i+1}$

Recall that $c'_1 = c_1$ (by design of $\mathcal{V}$) and so $\forall j \in \{1,...,m\}, (L^j_1)' = L^j_1$ and $(R^j_1)' = R^j_1$. We therefore conclude by induction on c'_i that $\forall i\ 1 \leq i \leq \pi$, $c'_i = c_i$. In particular, $c'_\pi = c_\pi$. This implies that $\forall j \in \{1,.,m\}$:

$\{\ (L^j_\pi)' = (r^j_\pi \otimes G) \oplus (c'_\pi \otimes y^j_\pi) = ((q^j_\pi - c_\pi x^j_\pi) \otimes G) \oplus (c_\pi \otimes y^j_\pi) = q^j_\pi \otimes G = L^j_\pi$

$\{\ (R^j_\pi)' = (r^j_\pi \otimes \mathcal{H}_2(y^j_\pi)) \oplus (c'_\pi \otimes I^j_\pi) = ((q^j_\pi - c_\pi x^j_\pi) \otimes \mathcal{H}_2(y^j_\pi)) \oplus (c_\pi \otimes I^j_\pi) =$
$q^j_\pi \otimes \mathcal{H}_2(y^j_\pi) = R^j_\pi$

We can then invoke a similar induction argument on c'_i as the one stated earlier, but this time for $\pi \leq i \leq n$. We therefore conclude that:

$$c_1 \equiv c_{n+1} \equiv \mathcal{H}_1(m, L^1_n, R^1_n, ..., L^m_n, R^m_n) \pmod{l}\ \textit{(by design of } \Sigma\textit{)}$$

$$= \mathcal{H}_1(m, (L^1_n)', (R^1_n)', ..., (L^m_n)', (R^m_n)') \pmod{l}\ \textit{(by induction proof showing that } \forall j \in \{1,...,m\}, (L^j_n)' = L^j_n \textit{ and } (R^j_n)' = R^j_n\textit{)}$$

$$= c'_{n+1}$$

Subsequently, any Σ-generated signature will satisfy $\mathcal{V}$'s verification test.

4 Security analysis - Unforgeability vis-a-vis EFACM

For unforgeability proofs, we follow the 5-step approach outlined earlier in part 1. (Recall that for ring signatures, we prove resilience against EFACM with respect to a fixed ring attack as described in part 3 of this series. The nuance here is that we have a fixed public key matrix PK, as opposed to a fixed public key vector L).

Step 1 : To prove that this scheme is secure against EFACM in the RO model, we proceed by contradiction and assume that there exists a PPT adversary $\mathcal{A}$ such that:

$$P_{\omega,r,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\Sigma^{\mathcal{H}_1,\mathcal{H}_T}(r)}\ \textit{succeeds in EFACM}] = \epsilon(k), \text{ for } \epsilon \text{ non-negligible in } k.$$

Step 2 : Next, we build a simulator $\mathcal{S}(r')$ such that it:

- Does not have access to any component of the private key vector of any signer.
- Has the same range as the original signing algorithm Σ (i.e., they output signatures taken from the same pool of potential signatures over all possible choices of RO functions and random tapes r' and r).

- Has indistinguishable probability distribution from that of Σ over this range.

Original Signer $\Sigma(r)$

$\boxed{m}$ $\boxed{\vec{x_\pi}}$ $\boxed{\vec{I_\pi}}$ $\boxed{\text{Fixed } PK \text{ matrix}}$

$\forall j \in \{1,.,m\}$, pick rand $q^j_\pi \in \{1,...,l\} \equiv \mathbb{F}_l$

Set $L^j_\pi \equiv (q^j_\pi \otimes G)$

Calculate $\mathcal{H}_T(y^j_\pi) = (v^j_\pi, \mathcal{H}_2(y^j_\pi))$
(rand v^j_π in $\mathbb{F}^*_l, \mathcal{H}_2(y^j_\pi) \equiv v^j_\pi \otimes G$)
Set $R^j_\pi \equiv (q^j_\pi \otimes \mathcal{H}_2(y^j_\pi))$

Set $c_{\pi+1} \equiv$
$\mathcal{H}_1(m, L^1_\pi, R^1_\pi, ., L^m_\pi, R^m_\pi) \pmod l$

$\forall i \in \{\pi+1, ..., n, 1, ..., \pi-1\}$
$\forall j \in \{1, ..., m\}$
Choose rand $r^j_i \in \{1, ...l\} \equiv \mathbb{F}_l$

Set $L^j_i \equiv (r^j_i \otimes G) \oplus (c_i \otimes y^j_i)$

Calculate $\mathcal{H}_T(y^j_i) = (v^j_i, \mathcal{H}_2(y^j_i))$
(rand v^j_i in $\mathbb{F}^*_l, \mathcal{H}_2(y^j_i) \equiv v^j_i \otimes G$)
Set
$R^j_i \equiv (r^j_i \otimes \mathcal{H}_2(y^j_i)) \oplus (c_i \otimes I^j_\pi)$

Set $c_{i+1} \equiv$
$\mathcal{H}_1(m, L^1_i, R^1_i, ., L^m_i, R^m_i) \pmod l$
(where $c_1 \equiv c_{n+1}$)

$\boxed{\mathcal{H}_1}$ $\boxed{\mathcal{H}_T}$

$\forall i \in \{\pi+1, .., n, 1, .., \pi-1\}$
Set $r^j_\pi \equiv q^j_\pi - c_\pi x^j_\pi \pmod l$

output $(I^1_\pi, ., I^m_\pi, c_1, r^1_1, ., r^m_1, ., r^1_n, ., r^m_n)$

Verifier:
1) Let $c'_1 \equiv c_1$

2) $\forall i \in \{1, ., n\}$, $\forall j \in \{1, ., m\}$:
$(L^j_i)' \equiv (r^j_i \otimes G) \oplus (c'_i \otimes y^j_i)$
$(R^j_i)' \equiv (r^j_i \otimes \mathcal{H}_2(y^j_i)) \oplus (c'_i \otimes I^j_\pi)$

Calculate $c'_{i+1} \equiv$
$\mathcal{H}_1(m, (L^1_i)', (R^1_i)', ., (L^m_i)', (R^m_i)')$
$\pmod l$

3) Check if $c_1 = c'_{n+1}$

Simulator $\mathcal{S}(r')$ *(bypasses RO $\mathcal{H}_1$)*

$\boxed{m}$ $\boxed{\text{Fixed } PK \text{ matrix}}$

Choose random $\pi \in \{1, .., n\}$

$\forall j \in \{1, ..., m\}$
Calculate $\mathcal{H}_T(y^j_\pi) = (v^j_\pi, \mathcal{H}_2(y^j_\pi))$
(rand v^j_π in $\mathbb{F}^*_l, \mathcal{H}_2(y^j_\pi) \equiv v^j_\pi \otimes G$)

Set $(I^j_\pi)^\sim \equiv v^j_\pi \otimes y^j_\pi$
(this implies
$(I^j_\pi)^\sim = x^j_\pi \otimes \mathcal{H}_2(y^j_\pi)$*)*

$\forall i \in \{1, .., n\},\ j \in \{1, ..., m\}$
Pick rand $c^\sim_i, (r^j_i)^\sim \in \mathbb{F}_l$
Set $(L^j_i)^\sim \equiv$
$((r^j_i)^\sim \otimes G) \oplus (c^\sim_i \otimes y^j_i)$

$\boxed{\mathcal{H}_T}$

$\forall i \in \{1, .., n\}, i \neq \pi,\ j \in \{1, .., m\}$
calculate $\mathcal{H}_T(y^j_i) = (v^j_i, \mathcal{H}_2(y^j_i))$
(rand v^j_i in $\mathbb{F}^*_l, \mathcal{H}_2(y^j_i) \equiv v^j_i \otimes G$)

$\forall i \in \{1, .., n\}$, $\forall j \in \{1, ..., m\}$
Set $(R^j_i)^\sim \equiv$
$((r^j_i)^\sim \otimes \mathcal{H}_2(y^j_i)) \oplus (c^\sim_i \otimes I^j_\pi)$

Let $c^\sim_{n+1} \equiv c^\sim_1$, and $\forall i \in \{1, ., n\}$, Set
$\mathcal{H}_1(m, (L^1_i)^\sim, (R^1_i)^\sim, ., (L^m_i)^\sim, (R^m_i)^\sim)$
$\equiv c^\sim_{i+1}$
(bypass RO $\mathcal{H}_1$)

output $((I^1_\pi)^\sim, ., (I^m_\pi)^\sim, c^\sim_1,$
$(r^1_1)^\sim, ., (r^m_1)^\sim, ., (r^1_n)^\sim, ., (r^m_n)^\sim)$

Verifier:
1) Let $(c^\sim_1)' \equiv c^\sim_1$

2) $\forall i \in \{1, ., n\}, \forall j \in \{1, ., m\}$:
$((L^j_i)^\sim)' \equiv ((r^j_i)^\sim \otimes G) \oplus ((c^\sim_i)' \otimes y^j_i)$

$((R^j_i)^\sim)' \equiv ((r^j_i)^\sim \otimes \mathcal{H}_2(y^j_i))$
$\oplus((c^\sim_i)' \otimes I_{\pi^j})$

$(c^\sim_{i+1})' \equiv$
$\mathcal{H}_1(m, (L^1_i)^\sim, (R^1_i)^\sim, ., (L^m_i)^\sim, (R^m_i)^\sim)$
$\pmod l$

3) Check if $c^\sim_1 = (c^\sim_{n+1})'$

The reason we introduced $\mathcal{H}_T$ as opposed to introducing only $\mathcal{H}_2$ is that the simulator makes use of the random elements $v_\pi^j,\ j \in \{1, ..., m\}$ in order to set $(I_\pi^j)^\sim$ to the desired value. In other words, in the absence of knowledge about x_π^j, the simulator needs to have access to the random element $v_\pi^j \in \mathbb{F}_l^*$ that is used in the calculation of $\mathcal{H}_2(y_\pi^j)$ in order to ensure that $(I_\pi^j)^\sim$ equates to $x_\pi^j \otimes \mathcal{H}_2(y_\pi^j)$

By construction, the output of $\mathcal{S}$ will satisfy the verification equation. Moreover, it does its own random assignments to what otherwise would be calls to RO $\mathcal{H}_1$ (i.e., S bypasses RO $\mathcal{H}_1$). Next, note the following:

1. $\mathcal{S}$ does not use any private key.

2. Σ and $\mathcal{S}$ both have a range $R \equiv (\gamma^1, ...\gamma^m, \epsilon_1, \beta_1^1, ..., \beta_1^m, ...\beta_n^1, ..., \beta_n^m) \in (\{G\}^*)^m \times (\mathbb{F}_l)^{mn+1}$ such that $\epsilon_1 = \mathcal{H}_1(m, (L_n^1)', (R_n^1)', ..., (L_n^m)', (R_n^m)') \pmod{l}$ and where $(L_n^j)'$ and $(R_n^j)'$ are calculated as follows:

   - Let $c_1' \equiv \epsilon_1$

   - $\forall i \in \{1, .., n\}$:

     { $\forall j \in \{1, .., m\}$ compute:

     { $(L_i^j)' = (\beta_i^j \otimes G) \oplus (c_i' \otimes y_i^j)$

     { $(R_i^j)' = (\beta_i^j \otimes \mathcal{H}_2(y_i^j)) \oplus (c_i' \otimes \gamma^j)$

     { $c_{i+1}' = \mathcal{H}_1(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$

3. Σ and $\mathcal{S}$ have the same probability distribution over R. Indeed, $\forall(\gamma^1, ...\gamma^m, \epsilon_1, \beta_1^1, ..., \beta_1^m, ...\beta_n^1, ..., \beta_n^m) \in R$, we have:

   - For Σ :

   $$P[(I_\pi^1, ., I_\pi^m, c_1, r_1^1, ., r_1^m, ., r_n^1, ., r_n^m) = (\gamma^1, ., \gamma^m, \epsilon_1, \beta_1^1, ., \beta_1^m, ., \beta_n^1, ., \beta_n^m)] =$$

   $$P_{I_\pi^j \in \{G\}^*,\ c_1 \in \mathbb{F}_l,\ r_i^j \in \mathbb{F}_l}[(I_\pi^j = \gamma^j,\ \forall j \in \{1, ..., m\}) \ \cap\ (c_1 = \epsilon_1) \ \cap\ (r_i^j = \beta_i^j,\ \forall i \in \{1, ..., n\},\ j \in \{1, ..., m\})]$$

   $$= \left(\frac{1}{|\{G\}^*|}\right)^m \times \left(\frac{1}{l}\right)^{mn+1} = \frac{1}{(l-1)^m \times l^{mn+1}}$$

   The first factor is the probability of choosing the exact I_π^j value in the set $\{G\}^*$ that is equal to γ^j. The second factor is the probability of choosing the exact $mn+1$ values given by ϵ_1 and the β_i^j's $\in \mathbb{F}_l$.

   - For $\mathcal{S}$:

   $$P[((I_\pi^1)^\sim, ., (I_\pi^m)^\sim, c_1^\sim, (r_1^1)^\sim, ., (r_1^m)^\sim, ., (r_n^1)^\sim, ., (r_n^m)^\sim) = (\gamma^1, ., \gamma^m, \epsilon_1, \beta_1^1, ., \beta_1^m, ., \beta_n^1, ., \beta_n^m)] =$$

$$P_{(I_\pi^j)^\sim \in \{G\}^*,\ c_1^\sim \in \mathbb{F}_l,\ (r_i^j)^\sim \in \mathbb{F}_l}[((I_\pi^j)^\sim = \gamma^j,\ \forall j \in \{1, ..., m\}) \cap (c_1^\sim = \epsilon_1) \cap ((r_i^j)^\sim = \beta_i^j,\ \forall i \in \{1, ..., n\},\ j \in \{1, ..., m\})]$$

$$= (\tfrac{1}{|\{G\}^*|})^m \times (\tfrac{1}{l})^{mn+1} = \tfrac{1}{(l-1)^m \times l^{mn+1}}$$

Note that the range of $(I_\pi^j)^\sim$ is equal to $\{G\}^*$ by construction of S. And so the first factor is the probability of choosing the exact $(I_\pi^j)^\sim$ value in the set $\{G\}^*$ that is equal to γ^j. The second factor is the probability of choosing the exact $mn+1$ values given by ϵ_1 and the β_i^j's $\in \mathbb{F}_l$.

With $\mathcal{S}$ adequately built, we conclude that (as justified in section 6 of part 1):

$$P_{\omega,r,'\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\mathcal{S}^{\mathcal{H}_T}(r')}\ \textit{succeeds in EFACM}] = \epsilon(k), \text{ for } \epsilon \text{ non-negligible in } k.$$

Step 3 : We now show that the probability of faulty collisions is negligible (refer to section 6 of part 1 for an overview). The 2 tyes of collisions are:

- $Col_{Type\ 1}$: $\exists\ i \in \{1, .., n\}$ such that a tuple $(m, L_i^1, R_i^1, ..., L_i^m, R_i^m)$ that $\mathcal{S}$ encounters – recall that $\mathcal{S}$ makes its own random assignment to $\mathcal{H}_1(m, L_i^1, R_i^1, ..., L_i^m, R_i^m)$ and bypasses RO $\mathcal{H}_1$ – also appears in the list of queries that $\mathcal{A}(\omega)$ sends to RO $\mathcal{H}_1$. A conflict in the 2 values will happen with overwhelming probability and the execution will halt.
- $Col_{Type\ 2}$: $\exists\ i, j \in \{1, .., n\}$ such that a tuple $(m, L_i^1, R_i^1, ., L_i^m, R_i^m)$ that $\mathcal{S}$ encounters – recall that $\mathcal{S}$ makes its own random assignment to $\mathcal{H}_1(m, L_i^1, R_i^1, ., L_i^m, R_i^m)$ – is the same as another tuple $(m', (L_k^1)', (R_k^1)', ., (L_k^m)', (R_k^m)')$ that $\mathcal{S}$ encountered earlier – here too, $\mathcal{S}$ would have made its random assignment to $\mathcal{H}_1(m', (L_k^1)', (R_k^1)', ., (L_k^m)', (R_k^m)')$. Since $(m, L_i^1, R_i^1, ., L_i^m, R_i^m) = (m', (L_k^1)', (R_k^1)', ., (L_k^m)', (R_k^m)')$, the assignments must also match (i.e., $\mathcal{H}_1(m, L_i^1, R_i^1, ., L_i^m, R_i^m) = \mathcal{H}_1(m', (L_k^1)', (R_k^1)', ., (L_k^m)', (R_k^m)')$). However, the likelihood that the 2 are equal is negligible. Hence they will be different with overwhelming probability and the execution will halt.

The aforementioned collisions must be avoided. In order to do so, we first calculate the probability of their occurence. We assume that during an EFACM attack, $\mathcal{A}(\omega)$ can make a maximum of Q_1 queries to RO $\mathcal{H}_1$, a maximum of Q_T queries to RO $\mathcal{H}_T$, and a maximum of Q_S queries to $\mathcal{S}(r')$. Q_1, Q_T, and Q_S are all assumed to be polynomial in the security parameter k, since the adversary is modeled as a PPT Turing machine.

$$P[Col_{Type\ 1}] = P[\cup_{all\ (m,L_i^1,R_i^1,.,L_i^m,R_i^m),\ (i=1,.,n)}\{(m, L_i^1, R_i^1, ., L_i^m, R_i^m)\ \textit{appeared in at least one of the } Q_S \textit{ queries to } \mathcal{S} \textit{ and } Q_1 \textit{ queries to RO } \mathcal{H}_1\}]$$

$$\leq \sum_{i=1}^n P[\cup_{all\ L_i^1}\{L_i^1\ \textit{was part of at least one of the } Q_S \textit{ queries to } \mathcal{S} \textit{ and } Q_1 \textit{ queries to RO } \mathcal{H}_1\}]$$

$$\leq \sum_{i=1}^n \sum_{all\ L_i^1 \in \{G\}} P[\cup_{(j=1,..,Q_S),\ (k=1,..,Q_1)}\{L_i^1\ \textit{was part of at least the } j^{th} \textit{ query to } \mathcal{S}$$

$$and\ k^{th}\ queries\ to\ RO\ \mathcal{H}_1\}]$$

$$\leq \sum_{i=1}^{n} \sum_{all\ L_i^1 \in \{G\}} \sum_{j=1}^{Q_S} \sum_{k=1}^{Q_1} P[L_i^1\ was\ part\ of\ at\ least\ the\ j^{th}\ query\ to\ \mathcal{S}\ and\ k^{th}\ queries\ to\ RO\ \mathcal{H}_1]$$

$$\leq \sum_{i=1}^{n} \sum_{all\ L_i^1 \in \{G\}} \sum_{j=1}^{Q_S} \sum_{k=1}^{Q_1} \frac{1}{|\{G\}|^2} = n \times |\{G\}| \times \frac{Q_S Q_1}{|\{G\}|^2} = n \times \frac{Q_S Q_1}{|\{G\}|} < \frac{nQ_S Q_1}{2^k}.$$

$$(since\ k < log_2(|\{G\}^*|) < log_2(|\{G\}|)\ by\ design).$$

Recalling that Q_S and Q_1 are polynomial in k, we conclude that $P[Col_{Type\ 1}]$ is negligible in k.

Next, we compute $P[Col_{Type\ 2}]\ =$

$$P[\cup_{all\ (m, L_i^1, R_i^1, .., L_i^m, R_i^m),\ (i=1,..,n)}\{(m, L_i^1, R_i^1, ., L_i^m, R_i^m)\ appeared\ at\ least\ twice\ during\ queries\ to\ \mathcal{S}\}]$$

$$\leq \sum_{i=1}^{n} P[\cup_{all\ L_i^1 \in \{G\}}\{L_i^1\ was\ part\ of\ at\ least\ 2\ queries\ to\ \mathcal{S}\}]$$

$$\leq \sum_{i=1}^{n} \sum_{L_i^1 \in \{G\}} \binom{Q_S}{2} \times \frac{1}{|\{G\}|^2} \leq n \times |\{G\}| \times \binom{Q_S}{2} \times \frac{1}{|\{G\}|^2} < n \times \binom{Q_S}{2} \times \frac{1}{|\{G\}|} < \frac{nQ_S^2}{2 \times 2^k}.$$

$$(\text{since } k < log_2(|\{G\}^*|) < log_2(|\{G\}|) \text{ by design}).$$

Recalling that Q_S is polynomial in k, we conclude that $P[Col_{Type\ 2}]$ is negligible in k.

Putting it altogether, we find that the below quantity is negligible in k:

$$P[Col] = P[Col_{Type\ 1} \cup Col_{Type\ 2}] \leq \sum_{i=1}^{2} P[Col_{Type\ i}] \leq n(\frac{Q_S Q_1 + \frac{Q_S^2}{2}}{2^k}) \equiv \delta(k)$$

This allows us to conclude that the below quantity is non-negligible in k (refer to section 6 of part 1 for a justification):

$$P_{\omega, r,' \mathcal{H}_1, \mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1, \mathcal{H}_T, \mathcal{S}^{\mathcal{H}_T}(r')} succeeds\ in\ EFACM\ \cap \overline{Col}] \geq \epsilon(k) - \delta(k).$$

Step 4 : In this step, our objective is to show that if $(\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*)$ is a successful tuple that generated a first EFACM forgery, then the following quantity is non-negligible in k:

$$P_{\mathcal{H}_1}[\mathcal{A}(\omega^*)^{\mathcal{H}_1, \mathcal{H}_T^*, \mathcal{S}^{\mathcal{H}_T}(r'^*)}\ succeeds\ in\ EFACM\ \cap\ (\rho_{\alpha(\mu_{\vec{\beta}})} \neq \rho^*_{\alpha(\mu_{\vec{\beta}})})\ |\ (\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*)\ is\ a\ succesfull\ first\ forgery,\ and\ (\rho_i = \rho_i^*)\ for\ i \in \{1, ...\alpha(\mu_{\vec{\beta}}) - 1\}]$$

Here $\alpha(\mu_{\vec{\beta}})$ is an appropriate index that we will define in the proof. To further simplify the notation, we let $\rho_i^* \equiv \mathcal{H}_1^*(q_i^*)$ and $\rho_i \equiv \mathcal{H}_1(q_i)$ for all $i \in 1, .., \alpha(\mu_{\vec{\beta}})$. ($q_i$ and q_i^* denote respectively the i^{th} query to $\mathcal{H}_1$ and to $\mathcal{H}_1^*$).

Let's take a closer look at $P_{\omega,r,'\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\mathcal{S}^{\mathcal{H}_T}(r')} succeeds\ in\ EFACM \cap \overline{Col}]$.

Any successful forgery $(I^1, ..., I^m, c_1, r_1^1, ..r_1^m, ...r_n^1, ..., r_n^m)$ must satisfy the verification equation $c_1 = \mathcal{H}_1(m, (L_n^1)', (R_n^1)', ..., (L_n^m)', (R_n^m)') \pmod l$ where we let $c_1' \equiv c_1$, and $\forall i \in \{1, .., n\}$:

$\{\ \forall j \in \{1, ..., m\}$, compute:

$\{\ (L_i^j)' \equiv (r_i^j \otimes G)\ \oplus\ (c_i \otimes y_i^j)$
$\{\ (R_i^j)' \equiv (r_i^j \otimes \mathcal{H}_2(y_i^j))\ \oplus\ (c_i \otimes I^j)$

$\{\ c_{i+1}' \equiv \mathcal{H}_1(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)') \pmod l$

We distinguish between 3 scenarios (without loss of generality, we assume that all $\mathcal{A}$-queries sent to RO $\mathcal{H}_1$ are distinct from each-other. Similarly, all $\mathcal{A}$-queries sent to RO $\mathcal{H}_T$ are distinct from each-other. This is because we can assume that $\mathcal{A}$ keeps a local copy of previous query results and avoid redundant calls):

- Scenario 1: $\mathcal{A}$ was successful in its forgery, and
  - No collisions occured, and
  - $\exists i \in \{1, .., n\}$ such that it never queried RO $\mathcal{H}_1$ on input $(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$.
- Scenario 2: $\mathcal{A}$ was successful in its forgery, and
  - No collisions occured, and
  - $\forall i \in \{1, .., n\}$ it queried RO $\mathcal{H}_1$ on input $(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$ during execution, and
  - $\exists i \in \{1, .., n\},\ j \in \{1, .., m\}$ such that it queried RO $\mathcal{H}_T$ on input y_i^j after it had queried RO $\mathcal{H}_1$ on input $(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$.
- Scenario 3: $\mathcal{A}$ was successful in its forgery, and
  - No collisions occured, and
  - $\forall i \in \{1, .., n\}$ it queried RO $\mathcal{H}_1$ on input $(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$ during execution, and
  - $\forall i \in \{1, .., n\},\ j \in \{1, .., m\}$, it queried RO $\mathcal{H}_T$ on input y_i^j before it queried RO $\mathcal{H}_1$ on input $(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$.

The probability of scenario 1 is upper-bounded by the probability that $\mathcal{A}$ picks c_{i+1}' such that it matches the value of $\mathcal{H}_1(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$. If the 2 values don't match, then c_1 will be different than c_{n+1}' (by the verification algorithm $\mathcal{V}$). It is upper-bounded because at the very least, this constraint must be observed to pass the verification test. $\mathcal{H}_1(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$ is the value that RO $\mathcal{H}_1$ returns to $\mathcal{V}$ (the verification algorithm) when verifying the validity of the forged signature. And since c_{i+1}' can be any value in the range of $\mathcal{H}_1$ (which was defined to be $\mathbb{F}_q$) we get:

$$P[Scenario\ 1] \leq \frac{1}{q} < \frac{1}{l} = \frac{1}{|\{G\}|} < \frac{1}{|\{G\}^*|} \leq \frac{1}{2^k}, \text{ which is negligible in } k.$$

In scenario 2, let $i \in \{1, .., n\}$ and $j \in \{1, .., m\}$ be 2 indices such that $\mathcal{A}$ queried RO $\mathcal{H}_T$ on input y_i^j after it had queried RO $\mathcal{H}_1$ on input $(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)')$. Note that during the verification process, $\mathcal{V}$ will calculate $(R_i^j)' \equiv (r_i^j \otimes \mathcal{H}_2(y_i^j)) \oplus (c_i' \otimes I^j)$ and hence will make a call to $\mathcal{H}_T$ on input y_i^j (*remember that $\mathcal{H}_2$ is derived from $\mathcal{H}_T$*). The probability that the resulting $(R_i^j)'$ matches the $(R_i^j)'$ argument previously fed to $\mathcal{H}_1$ is upper-bounded by $\frac{1}{|\{G\}^*|}$ (since the range of $\mathcal{H}_2 = |\{G\}^*|$). Moreover, i can be any index in $\{1, .., n\}$ and j any index in $\{1, .., m\}$. We get:

$$P[\textit{Scenario } 2] \leq \frac{mn}{|\{G\}^*|} \leq \frac{mn}{2^k}, \text{ which is negligible in } k.$$

So we assume that a successful forgery will likely be of the Scenario 3 type.

$$\begin{gathered} P[\textit{Scenario } 3] = \\ P_{\omega,r,'\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\mathcal{S}^{\mathcal{H}_T}(r')} \textit{succeeds in EFACM } \cap \overline{Col}] - P[\textit{Scenario } 1] \\ -P[\textit{Scenario } 2] \end{gathered}$$

$$\geq \epsilon(k) - \delta(k) - \frac{1}{2^k} - \frac{mn}{2^k} \equiv \nu(k), \text{ which is non-negligible in } k$$

The rest of the proof for Step 4 is exactly the same as the one outlined for the LSAG scheme. We will not reproduce it here (the reader can refer to the details in section 4 of part 5).

Step 5 : The final step uses the 2 forgeries obtained earlier to solve an instance of the Discrete Logarithm (DL) problem. Here is a recap of Step 4 results:

- With non-negligible probability of at least $\frac{\nu(k)}{4}$ we get a successful tuple $(\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*)$, s.t. $(\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*) \in (\Omega_{\vec{\beta}} \cap S_{\vec{\beta}})$ for some vector of indices $\vec{\beta} \in I$ (note that in this context, I refers to a specific set of indices and not to the key-image vector. Review section 4 of part 5 for a definition of I and $\vec{\beta}$). By running $\mathcal{A}$ a number of times polynomial in k, we can find such a tuple.
- Once we find such a tuple, we've also shown that with non-negligible probability of at least $\frac{\nu(k)}{4V_{(Q_1+Q_T),n}} - \frac{1}{2^k}$, we can find another successful tuple $(\omega^*, r'^*, \mathcal{H}_1^\sim, \mathcal{H}_T^*)$ s.t. $(\omega^*, r'^*, \mathcal{H}_1^\sim, \mathcal{H}_T^*) \in S_{\vec{\beta}}$ and $(\rho_1^\sim = \rho_1^*), .., (\rho_{\alpha(\mu_{\vec{\beta}})-1}^\sim = \rho_{\alpha(\mu_{\vec{\beta}})-1}^*)$, $(\rho_{\alpha(\mu_{\vec{\beta}})}^\sim \neq \rho_{\alpha(\mu_{\vec{\beta}})}^*)$.

Let $(\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*)$ correspond to forgery

$$\sigma_a(m_a, PK) \equiv ((I^1)_a, ..., (I^m)_a, (c_1)_a, (r_1^1)_a, .., (r_1^m)_a, ..., (r_n^1)_a, .., (r_n^m)_a)$$

and $(\omega^*, r'^*, \mathcal{H}^\sim, \mathcal{H}_T^*)$ correspond to forgery

$$\sigma_b(m_b, PK) \equiv ((I^1)_b, ..., (I^m)_b, (c_1)_b, (r_1^1)_b, .., (r_1^m)_b, ..., (r_n^1)_b, .., (r_n^m)_b).$$

Recall that $\alpha(\mu_{\vec{\beta}})$ is the index of the last $(m, (L_i^1)', (R_i^1)', ., (L_i^m)', (R_i^m)')$, $i \in \{1, ., n\}$ query that $\mathcal{A}$ sends to RO $\mathcal{H}_1$ ($\mu_{\vec{\beta}} = max_{i=1}^n(\vec{\beta})_i$). Since the 2 experiments corresponding to the 2 successful tuples have:

- The same random tapes ω^* and r'^*
- The same RO $\mathcal{H}_T^*$
- ROs $\mathcal{H}_1^*$ and $\mathcal{H}_1^\sim$ behave the same way on the first $\alpha(\mu_{\vec{\beta}}) - 1$ queries,

we can be confident that the first $\alpha(\mu_{\vec{\beta}})$ queries sent to the 2 ROs $\mathcal{H}_1^*$ and $\mathcal{H}_1^\sim$ are identical. In other words, we have $\forall i \in \{1, ..., n\}$

$$(m_a, (L_i^1)'_a, (R_i^1)'_a, ..., (L_i^m)'_a, (R_i^m)'_a) = (m_b, (L_i^1)'_b, (R_i^1)'_b, ..., (L_i^m)'_b, (R_i^m)'_b)$$

Without loss of generality, let $(m, (L_\zeta^1)', (R_\zeta^1)', ..., (L_\zeta^m)', (R_\zeta^m)')$, (where $\zeta \in \{1, ..., n\}$), correspond to the last query of this type sent to RO $\mathcal{H}_1$. That means that $(m, (L_\zeta^1)', (R_\zeta^1)', ..., (L_\zeta^m)', (R_\zeta^m)')$ is the $\mu_{\vec{\beta}}^{th}$ query sent to RO $\mathcal{H}_1$. We have:

$$(m_a, (L_{\zeta+1}^1)'_a, (R_{\zeta+1}^1)'_a, ..., (L_{\zeta+1}^m)'_a, (R_{\zeta+1}^m)'_a) = (m_b, (L_{\zeta+1}^1)'_b, (R_{\zeta+1}^1)'_b, ..., (L_{\zeta+1}^m)'_b, (R_{\zeta+1}^m)'_b)$$

$$\text{(where } (\zeta + 1) \equiv 1 \text{ whenever } \zeta = n)$$

$$\implies \forall j \in \{1, ..., m\},\ (L_{\zeta+1}^j)'_a = (L_{\zeta+1}^j)'_b$$

$$\implies ((r_{\zeta+1}^j)_a \otimes G) \oplus ((c'_{\zeta+1})_a \otimes y_{\zeta+1}^j) = ((r_{\zeta+1}^j)_b \otimes G) \oplus ((c'_{\zeta+1})_b \otimes y_{\zeta+1}^j),$$

$$\implies x_{\zeta+1}^j[(c'_{\zeta+1})_a - (c'_{\zeta+1})_b] = (r_{\zeta+1}^j)_b - (r_{\zeta+1}^j)_a \pmod{l} \text{ (by writing } y_{\zeta+1}^j = x_{\zeta+1}^j \otimes G)$$

Moreover, we have

$$(c'_{\zeta+1})_a = \mathcal{H}_1^*(m_a, (L_\zeta^1)'_a, (R_\zeta^1)'_a, ..., (L_\zeta^m)'_a, (R_\zeta^m)'_a) \pmod{l} \ \textit{(by definition of } c' \textit{ in } \mathcal{V})$$

$$= \rho^*_{\alpha(\mu_{\vec{\beta}})} \neq \rho^\sim_{\alpha(\mu_{\vec{\beta}})} \ \textit{(by design of the forgery tuples)}$$

$$= \mathcal{H}_1^\sim(m_b, (L_\zeta^1)'_b, (R_\zeta^1)'_b, ..., (L_\zeta^m)'_b, (R_\zeta^m)'_b) \pmod{l} = (c'_{\zeta+1})_b \ \textit{(by definition of } c' \textit{ in } \mathcal{V})$$

That means that $\forall j \in \{1, ..., m\}$, we can solve for $x_{\zeta+1}^j = \frac{(r_{\zeta+1}^j)_b - (r_{\zeta+1}^j)_a}{(c'_{\zeta+1})_a - (c'_{\zeta+1})_b} \pmod{l}$ in polynomial time, contradicting the intractability of DL on elliptic curve groups. We conclude that MLSAG is secure against EFACM in the RO model.

5 Security analysis - Exculpability

This section is concerned with the notion of exculpability from an unforgeability standpoint as described in [2]. A detailed discussion can be found in part 5 of this series. The setting is similar to the one previously described in parts 5 and 6, with a small nuance. Suppose all m private keys of each member of an $(n-1)$ subset of ring members have been compromised in an n-ring setting. Let π denote the index of the only non-compromised user with key-vector $\vec{x_\pi} \equiv [x_\pi^1 \ ... \ x_\pi^m]^T$, and let $\vec{I_\pi} \equiv [I_\pi^1 \ ... \ I_\pi^m]^T$ denote the key-image vector associated with it. We investigate whether it is likely to produce a valid forgery with key-image vector that includes at least one component equal to I_π^j for some $j \in \{1, ..., m\}$. In what follows, we show that this can only happen with negligible probability. In essence, this means that a non-compromised honest ring member (*by honest we mean a ring member that signs at most once using his private key-vector*) does not run the risk of encountering a forged signature that carries any component of his key-image vector. In the context of Monero, this implies that a non-compromised honest ring member cannot be accused of signing twice using the same key-image vector, and hence is exculpable.

Note that since the adversary $\mathcal{A}(\omega)$ has access to all the compromised private keys of $(n-1)$ members, it can easily calculate their corresponding public key vectors. Doing so will allow it to identify the public key vector $\vec{y_\pi} \equiv [y_\pi^1 \ ... \ y_\pi^m]^T$ of the non-compromised ring member and hence determine the index π. In order to prove the exculpability of the MLSAG scheme, we follow an almost identical proof to that of the previous section (i.e., unforgeability vis-a-vis EFACM) and apply the same 5-step approach. The objective is to show that this particular type of forgery would imply the ability to solve the DL of at least one component of $[y_\pi^1 \ ... \ y_\pi^m]^T$. The nuance resides in the specific index π for which the DL will be solved, as opposed to any other index. This is because we assume that all the other members are compromised and hence their DLs (i.e., private keys) are common-knowledge.

Step 1 : We proceed by contradiction and assume that there exists a PPT adversary $\mathcal{A}$ such that:

$$P_{\omega,r,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\{\vec{x_1},..,\hat{\vec{x_\pi}},..,\vec{x_n}\},\Sigma^{\mathcal{H}_1,\mathcal{H}_T}(r)} \ succeeds \ in \ creating \ a \ forgery \ \sigma(m,PK)] = \epsilon(k), \text{ for } \epsilon \text{ non-negligible in } k.$$

We refer to "*succeeds in creating a forgery* $\sigma(m, PK)$" as "*succeeds in* $EFACM_{Ex_\pi}$". We re-write the above equation as:

$$P_{\omega,r,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\{\vec{x_1},..,\hat{\vec{x_\pi}},..,\vec{x_n}\},\Sigma^{\mathcal{H}_1,\mathcal{H}_T}(r)} \ succeeds \ in \ EFACM_{Ex_\pi}] = \epsilon(k), \text{ for } \epsilon \text{ non-negligible in } k.$$

The notation used makes it explicit that $\mathcal{A}(\omega)$ can access the set of compromised key vectors $\{\vec{x_1}, .., \hat{\vec{x_\pi}}, .., \vec{x_n}\}$ with $\vec{x_\pi}$ excluded. Success is defined as issuing a forged signature with a key image vector that shares at least one component with the key image vector of the non-compromised user π. We let this component be $I_\pi^\delta \equiv x_\pi^\delta \otimes \mathcal{H}_2(y_\pi^\delta)$ for some $\delta \in \{1, ..., m\}$. (*Recall that* $\mathcal{H}_2$ *is derived from* $\mathcal{H}_T$).

Step 2 : The next step consists in building a simulator $\mathcal{S}(r')$ such that it:

- Does not have access to any component of the private key vector of any signer.
- Has the same range as the original signing algorithm Σ (i.e., they output signatures taken from the same pool of potential signatures over all possible choices of RO functions and respective random tapes r' and r).
- Has indistinguishable probability distribution from that of Σ over this range.

The simulator $S(r')$ is the same as the one we built in the previous section. The only difference is that $S(r')$ does not choose a random index π, since $\mathcal{A}(\omega)$ already knows the index of the non-compromised ring member.

Step 3 : The logical reasoning and procedure are identical to those of the previous section. We conclude that

$$P_{\omega,r,'\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\{\vec{x_1},..,\hat{\vec{x_\pi}},..,\vec{x_n}\},\mathcal{S}^{\mathcal{H}_T}(r')} succeeds\ in\ EFACM_{Ex_\pi} \cap \overline{Col}] \geq \epsilon(k) - \delta(k).$$

Step 4 : Here too, the logical reasoning and procedure are identical to those of the previous section. In particular, we define the following sets in a similar way:

- $S=$ $\{(\omega,r',\mathcal{H}_1,\mathcal{H}_T)|\ \mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\{\vec{x_1},..,\hat{\vec{x_\pi}},..,\vec{x_n}\},\ \mathcal{S}^{\mathcal{H}_T}(r')} succeeds\ in\ EFACM_{Ex x_\pi} \cap\ \overline{Col} \cap E \cap\ max_{i=1}^{n}[Ind(\omega,r',\mathcal{H}_1,\mathcal{H}_T)] \neq \infty\}$

- $S_{\vec{l}}=$ $\{(\omega,r',\mathcal{H}_1,\mathcal{H}_T)|\ \mathcal{A}(\omega)^{\mathcal{H}_1,\mathcal{H}_T,\{\vec{x_1},..,\hat{\vec{x_\pi}},..,\vec{x_n}\},\ \mathcal{S}^{\mathcal{H}_T}(r')} succeeds\ in\ EFACM_{Ex x_\pi} \cap\ \overline{Col} \cap E \cap\ Ind(\omega,r',\mathcal{H}_1,\mathcal{H}_T)=\vec{l}\}$

and conclude that:

$$P_{\mathcal{H}_1}[((\omega^*,r'^*,\mathcal{H}_1,\mathcal{H}_T^*) \in S_{\vec{\beta}}) \cap (\rho_{\alpha(\mu_{\vec{\beta}})} \neq \rho^*_{\alpha(\mu_{\vec{\beta}})})\ |\ (\omega^*,r'^*,\mathcal{H}_1^*,\mathcal{H}_T^*) \in S_{\vec{\beta}},\ \rho_1=\rho_1^* ...,\ \rho_{\alpha(\mu_{\vec{\beta}})-1}=\rho^*_{\alpha(\mu_{\vec{\beta}})-1})]$$

$$\geq \frac{\nu(k)}{4V_{(Q_1+Q_T),n}} - \frac{1}{2^k}, \text{ which is non-negligible in } k.$$

Here $\alpha(\mu_{\vec{\beta}})$, as before, is an appropriately defined index, $\rho_i^* \equiv \mathcal{H}_1^*(q_i)$, $\rho_i \equiv \mathcal{H}_1(q_i)$ for all $i \in 1,..,\alpha(\mu_{\vec{\beta}})$, and q_i denotes the i^{th} query sent to RO.

Step 5 : The final step uses the 2 forgeries obtained earlier to solve an instance of the Discrete Logarithm (DL) problem. Here is a recap of Step 4 results:

- With non-negligible probability of at least $\frac{\nu(k)}{4}$ we get a successful tuple $(\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*)$, s.t. $(\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*) \in (\Omega_{\vec{\beta}} \cap S_{\vec{\beta}})$ for some vector of indices $\vec{\beta} \in I$ (note that in this context, I refers to a specific set of indices and not to the key-image vector. Review section 4 of part 5 for a definition of I and $\vec{\beta}$). By running $\mathcal{A}$ a number of times polynomial in k, we can find such a tuple.
- Once we find such a tuple, we've also shown that with non-negligible probability of at least $\frac{\nu(k)}{4V_{(Q_1+Q_T),n}} - \frac{1}{2^k}$, we can find another successful tuple $(\omega^*, r'^*, \mathcal{H}_1^\sim, \mathcal{H}_T^*)$ s.t. $(\omega^*, r'^*, \mathcal{H}_1^\sim, \mathcal{H}_T^*) \in S_{\vec{\beta}}$ and $(\rho_1^\sim = \rho_1^*), .., (\rho^\sim_{\alpha(\mu_{\vec{\beta}})-1} = \rho^*_{\alpha(\mu_{\vec{\beta}})-1}),\ (\rho^\sim_{\alpha(\mu_{\vec{\beta}})} \neq \rho^*_{\alpha(\mu_{\vec{\beta}})})$.

Let $(\omega^*, r'^*, \mathcal{H}_1^*, \mathcal{H}_T^*)$ correspond to forgery

$$\sigma_a(m_a, PK) \equiv ((I^1)_a, ..., (I_\pi^\delta), ..., (I^m)_a, (c_1)_a, (r_1^1)_a, .., (r_1^m)_a, ..., (r_n^1)_a, .., (r_n^m)_a)$$

and $(\omega^*, r'^*, \mathcal{H}^\sim, \mathcal{H}_T^*)$ correspond to forgery

$$\sigma_b(m_b, PK) \equiv ((I^1)_b, ..., (I^\delta_\pi), ..., (I^m)_b, (c_1)_b, (r^1_1)_b, .., (r^m_1)_b, ..., (r^1_n)_b, .., (r^m_n)_b).$$

Recall that (I^δ_π) is the component of π's key image vector that appears in the forgery. $\alpha(\mu_{\vec{\beta}})$ is the index of the last $(m, (L^1_i)', (R^1_i)', ., (L^m_i)', (R^m_i)'),\ i \in \{1, ., n\}$ query that $\mathcal{A}$ sends to RO $\mathcal{H}_1$ ($\mu_{\vec{\beta}} = max^n_{i=1}(\vec{\beta})_i$). Since the 2 experiments corresponding to the 2 successful tuples have:

- The same random tapes ω^* and r'^*
- The same RO $\mathcal{H}^*_T$
- ROs $\mathcal{H}^*_1$ and $\mathcal{H}^\sim_1$ behave the same way on the first $\alpha(\mu_{\vec{\beta}}) - 1$ queries,

we can be confident that the first $\alpha(\mu_{\vec{\beta}})$ queries sent to the 2 ROs $\mathcal{H}^*_1$ and $\mathcal{H}^\sim_1$ are identical. In other words, we have $\forall i \in \{1, ..., n\}$

$$(m_a, (L^1_i)'_a, (R^1_i)'_a, ..., (L^m_i)'_a, (R^m_i)'_a) = (m_b, (L^1_i)'_b, (R^1_i)'_b, ..., (L^m_i)'_b, (R^m_i)'_b).$$

$$\implies \forall i \in \{1, .., n\},\ \forall j \in \{1, ..., m\},\ (L^j_i)'_a = (L^j_i)'_b, \text{ and } (R^j_i)'_a = (R^j_i)'_b$$

In particular, for the index $\delta \in \{1, ..., m\}$ that corresponds to the component I^δ_π that appears in the forgery, we let $(R^\delta_i)' \equiv (R^\delta_i)'_a = (R^\delta_i)'_b$, and $(L^\delta_i)' \equiv (L^\delta_i)'_a = (L^\delta_i)'_b$. For each $i \in \{1, .., n\}$, we get 2 identical systems of 2 equations dictated by $\mathcal{V}$'s verification computation:

First system of 2 linear equations

$$\{\ (R^\delta_i)' = ((r^\delta_i)_a \otimes \mathcal{H}_2(y^\delta_i)) \oplus ((c'_i)_a \otimes I^\delta_\pi)$$

$$\{\ (L^\delta_i)' = ((r^\delta_i)_a \otimes G) \oplus ((c'_i)_a \otimes y^\delta_i)$$

where $(c'_1)_a \equiv (c_1)_a$, and $(c'_{i+1})_a = \mathcal{H}_1(m_a, (L^1_i)'_a, (R^1_i)'_a, ., (L^m_i)'_a, (R^m_i)'_a)$ $\forall i \in \{1, ..., n\}$

Second system of 2 linear equations

$$\{\ (R^\delta_i)' = ((r^\delta_i)_b \otimes \mathcal{H}_2(y^\delta_i)) \oplus ((c'_i)_b \otimes I^\delta_\pi)$$

$$\{\ (L^\delta_i)' = ((r^\delta_i)_b \otimes G) \oplus ((c'_i)_b \otimes y^\delta_i)$$

where $(c'_1)_b \equiv (c_1)_b$, and $(c'_{i+1})_b = \mathcal{H}_1(m_b, (L^1_i)'_b, (R^1_i)'_b, ., (L^m_i)'_b, (R^m_i)'_b)$ $\forall i \in \{1, ..., n\}$

$\forall i \in \{1, .., n\}$, the first system is a linear system of 2 equations in variables $(r^\delta_i)_a$ and $(c'_i)_a$. Similarly, the second system is a linear system of 2 equations in variables $(r^\delta_i)_b$ and $(c'_i)_b$. The 2 systems are identical with different variable names. Hence, if $((r^\delta_i)^*_a, (c'_i)^*_a)$ is a unique solution to the first system and $((r^\delta_i)^*_b, (c'_i)^*_b)$ a unique solution to the second, we can be confident that $((r^\delta_i)^*_a = (r^\delta_i)^*_b$ and $(c'_i)^*_a = (c'_i)^*_b$. (Note that for any index $j \in \{1, ..., m\}$ other than δ, the 2 forged signatures do not necessarily share the same image-key component. In other terms $(I^j)_a$ is not necessarily equal $(I^j)_b$ and so the 2 systems of linear equations would be different from each other). For either system to admit a unique solution, the 2 equations must be linearly independent. We re-write the 2 systems as follows:

First system of 2 linear equations

$$\{\ (R^\delta_i)' = ((r^\delta_i)_a \otimes \mathcal{H}_2(y^\delta_i)) \oplus ((c'_i)_a \otimes I^\delta_\pi)$$

$$\{\ log_G(L_i\delta)' = (r^\delta_i)_a\ +\ (c'_i)_a\ x^\delta_i$$

where $(c'_1)_a \equiv (c_1)_a$, and $(c'_{i+1})_a = \mathcal{H}_1(m_a, (L^1_i)'_a, (R^1_i)'_a, ., (L^m_i)'_a, (R^m_i)'_a)$

$\forall i \in \{1, ..., n\}$

Second system of 2 linear equations

$$\{ \; (R_i^\delta)' = ((r_i^\delta)_b \otimes \mathcal{H}_2(y_i^\delta)) \oplus ((c_i')_b \otimes I_\pi^\delta)$$

$$\{ \; log_G(L_i\delta)' = (r_i^\delta)_b \; + \; (c_i')_b \; x_i^\delta$$

where $(c_1')_b \equiv (c_1)_b$, and $(c_{i+1}')_b = \mathcal{H}_1(m_b, (L_i^1)_b', (R_i^1)_b', ., (L_i^m)_b', (R_i^m)_b')$ $\forall i \in \{1, ..., n\}$

If we multiply the second equation by $\mathcal{H}_2(y_i^\delta)$ (multiplication refers to $\otimes$), we see that a sufficient condition for the system to be linearly independent is to have $[x_i^\delta \otimes \mathcal{H}_2(y_i^\delta)] \; \neq \; I_\pi^\delta \equiv [x_\pi^\delta \otimes \mathcal{H}_2(y_\pi^\delta)]$. Next, we show that with overwhelming probability, the system of linear equations is indeed independent $\forall i \in \{1, .., n\}, \; i \neq \pi$:

- Recall that the range of $\mathcal{H}_2$ is $\{G\}^*$ and that the order of $\{G\}^* = (l-1)$.
- Therefore, $\exists \; v_i^\delta, \; v_\pi^\delta \; \in \mathbb{F}_l^*$ such that $\mathcal{H}_2(y_i^\delta) = v_i^\delta \otimes G$ and $\mathcal{H}_2(y_\pi^\delta) = v_\pi^\delta \otimes G$.
- We can then re-write the sufficient condition as $x_i^\delta v_i^\delta \neq x_\pi^\delta v_\pi^\delta \pmod{l}$.
- Note that given x_i^δ, x_π^δ, and v_π^δ, there is at most one value of $v_i^\delta \in \mathbb{F}_l^*$ that satisfies $x_i^\delta v_i^\delta = x_\pi^\delta v_\pi^\delta \pmod{l}$. Otherwise, we would have $v_i^\delta, (v_i^\delta)' \in \mathbb{F}_l^*$, $v_i^\delta \neq (v_i^\delta)' \pmod{l}$, and $x_i^\delta v_i^\delta = x_\pi^\delta v_\pi^\delta = x_i^\delta (v_i^\delta)' \pmod{l}$. This would imply that $v_i^\delta \equiv (v_i^\delta)' \pmod{l}$, a contradiction.
- Noting that each v_i^δ corresponds to a distinct $\mathcal{H}_2(y_i^\delta)$, we conclude that given x_i^δ, x_π^δ and $\mathcal{H}_2(y_\pi^\delta)$ there is at most one $\mathcal{H}_2(y_i^\delta)$ s.t. $[x_i^\delta \otimes \mathcal{H}_2(y_i^\delta)] \; = \; I_\pi^\delta \equiv [x_\pi^\delta \otimes \mathcal{H}_2(y_\pi^\delta)]$.
- Since $\mathcal{H}_2$ is a RO outputing random values, the probability of getting the right value of $\mathcal{H}_2(y_i^\delta)$ is $\leq \frac{1}{|\{G\}^*|} < \frac{1}{|\{G\}|} < \frac{1}{2^k}$ (negligible in k).

$\forall i \in \{1, .., n\}, \; i \neq \pi$, we therefore conclude that with overwhelming probability we have $[x_i^\delta \otimes \mathcal{H}_2(y_i^\delta)] \neq I_\pi^\delta$. We can then be confident that the linear system of 2 equations has a unique solution. Hence, $\forall i \in \{1, .., n\}, \; i \neq \pi$, we have $(r_i^\delta)_a = (r_i^\delta)_b$, and $(c_i')_a = (c_i')_b$.

Moreover, by design of the 2 forgeries, we know that there exists one and only one $\zeta \in \{1, ..., n\}$ (corresponding to the $\mu_{\vec{\beta}}^{th}$ query sent to RO $\mathcal{H}_1$) that satisfies

$$(c_{\zeta+1}')_a = \mathcal{H}_1^*(m_a, (L_\zeta^1)_a', (R_\zeta^1)_a', ..., (L_\zeta^m)_a', (R_\zeta^m)_a') \pmod{l} \; \textit{(by definition of } c' \textit{ in } \mathcal{V})$$

$$= \rho_{\alpha(\mu_{\vec{\beta}})}^* \; \neq \; \rho_{\alpha(\mu_{\vec{\beta}})}^\sim \; \textit{(by design of the forgery tuples)}$$

$$= \mathcal{H}_1^\sim(m_b, (L_\zeta^1)_b', (R_\zeta^1)_b', ..., (L_\zeta^m)_b', (R_\zeta^m)_b') \pmod{l} = (c_{\zeta+1}')_b \; \textit{(by definition of } c' \textit{ in } \mathcal{V})$$

But $\forall i \in \{1, .., n\}, \; i \neq \pi$, we showed that with overwhelming probability $(c_i')_a = (c_i')_b$. Therefore, it must be that $(\zeta + 1) = \pi$ and so $(c_\pi')_a \neq (c_\pi')_b$.

Going back to the system of 2 equations associated with $i = \pi$, we write:

$$(r_\pi^\delta)_a + (c_\pi')_a \; x_\pi^\delta = log_G(L_\pi^\delta)' = (r_\pi^\delta)_b + (c_\pi')_b \; x_\pi^\delta$$

That means that we can solve for $x_{\pi}^{\delta} = \frac{(r_{\pi}^{\delta})_b - (r_{\pi}^{\delta})_a}{(c'_{\pi})_a - (c'_{\pi})_b}$ (mod l) in polynomial time, contradicting the intractability of DL on elliptic curve groups. We conclude that the MLSAG scheme is exculpable and secure against $EFACM_{Ex_{\pi}}$ in the RO model.

6 Security analysis - Anonymity

In this section, we show that the MLSAG scheme satisfies the weaker anonymity definition #2 introduced in part 3 of this series. Note that as we previously observed in part 5, linkable signatures cannot satisfy anonymity definition #1.

More formally, let $\mathcal{A}(\omega)$ be a PPT adversary with random tape ω that takes 4 inputs:

- Any message m.
- A public key matrix $PK \equiv [\vec{y_1} \ ... \ \vec{y_n}]$ that includes $\vec{y_{\pi}} \equiv [y_{\pi}^1 \ ... \ y_{\pi}^m]^T$ of the signer.
- A list $\mathcal{D}_t \equiv \{\hat{\vec{x_1}}, ..., \hat{\vec{x_t}}\}$ of compromised private key vectors $(0 \leq t \leq n)$. $\mathcal{D}_t$ can be empty and $\hat{\vec{x_i}}$ may be different than $\vec{x_i}$ for $i \in \{1, ..., n\}$. But $\mathcal{D}_t \subseteq \{\vec{x_1}, ..., \vec{x_n}\}$
- A valid signature $\sigma_{\pi}(m, PK)$ on message m, public key matrix PK and actual signer private key vector $\vec{x_{\pi}} \equiv [x_{\pi}^1 \ ... \ x_{\pi}^m]^T$.

$\mathcal{A}(\omega)$ outputs an index in $\{1, ..., n\}$ that it thinks is that of the actual signer. Definition # 2 mandates that for any polynomial $Q(k)$ in security parameter k, we have:

$$\frac{1}{n-t} - \frac{1}{Q(k)} \leq P[\mathcal{A}(\omega)(m, PK, \mathcal{D}_t, \sigma_{\pi}(m, PK)) = \pi \mid \sigma_{\pi}(m, PK) \ is \ valid] \leq \frac{1}{n-t} + \frac{1}{Q(k)}$$
$$\text{if } \vec{x_{\pi}} \notin \mathcal{D}_t \text{ and } 0 \leq t < n-1.$$

$$P[\mathcal{A}(\omega)(m, PK, \mathcal{D}_t, \sigma_{\pi}(m, PK)) = \pi \mid \sigma_{\pi}(m, PK) \ is \ valid] > 1 - \frac{1}{Q(k)}$$
$$\text{if } \vec{x_{\pi}} \in \mathcal{D}_t \text{ or } t = n-1.$$

In the RO model, $\mathcal{A}(\omega)$ can send a number of queries (polynomial in k) to RO $\mathcal{H}_1$ and RO $\mathcal{H}_T$. The probability of $\mathcal{A}$'s success is computed over the distributions of $\omega, \mathcal{H}_1$ and $\mathcal{H}_T$. Making explicit the dependence on the ROs, definition # 2's condition becomes:

$$\frac{1}{n-t} - \frac{1}{Q(k)} \leq P_{\omega, \mathcal{H}_1, \mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1, \mathcal{H}_T}(\omega)(m, PK, \mathcal{D}_t, \sigma_{\pi}(m, PK)) = \pi \mid \sigma_{\pi}(m, PK) \ is \ valid] \leq \frac{1}{n-t} + \frac{1}{Q(k)}, \text{ if } \vec{x_{\pi}} \notin \mathcal{D}_t \text{ and } 0 \leq t < n-1.$$

$$P_{\omega, \mathcal{H}_1, \mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1, \mathcal{H}_T}(\omega)(m, L, \mathcal{D}_t, \sigma_{\pi}(m, PK)) = \pi \mid \sigma_{\pi}(m, PK) \ is \ valid] > 1 - \frac{1}{Q(k)}$$
$$\text{if } \vec{x_{\pi}} \in \mathcal{D}_t \text{ or } t = n-1.$$

In order to prove that anonymity holds in the above sense, we proceed by contradiction and rely on the intractability of the *Decisional Diffie Hellman* problem or *DDH* for short (refer to part 5 for a discussion of DDH). We consider 3 separate cases:

- Case 1: $\vec{x_{\pi}} \notin \mathcal{D}_t$ and $0 \leq t < n-1$.

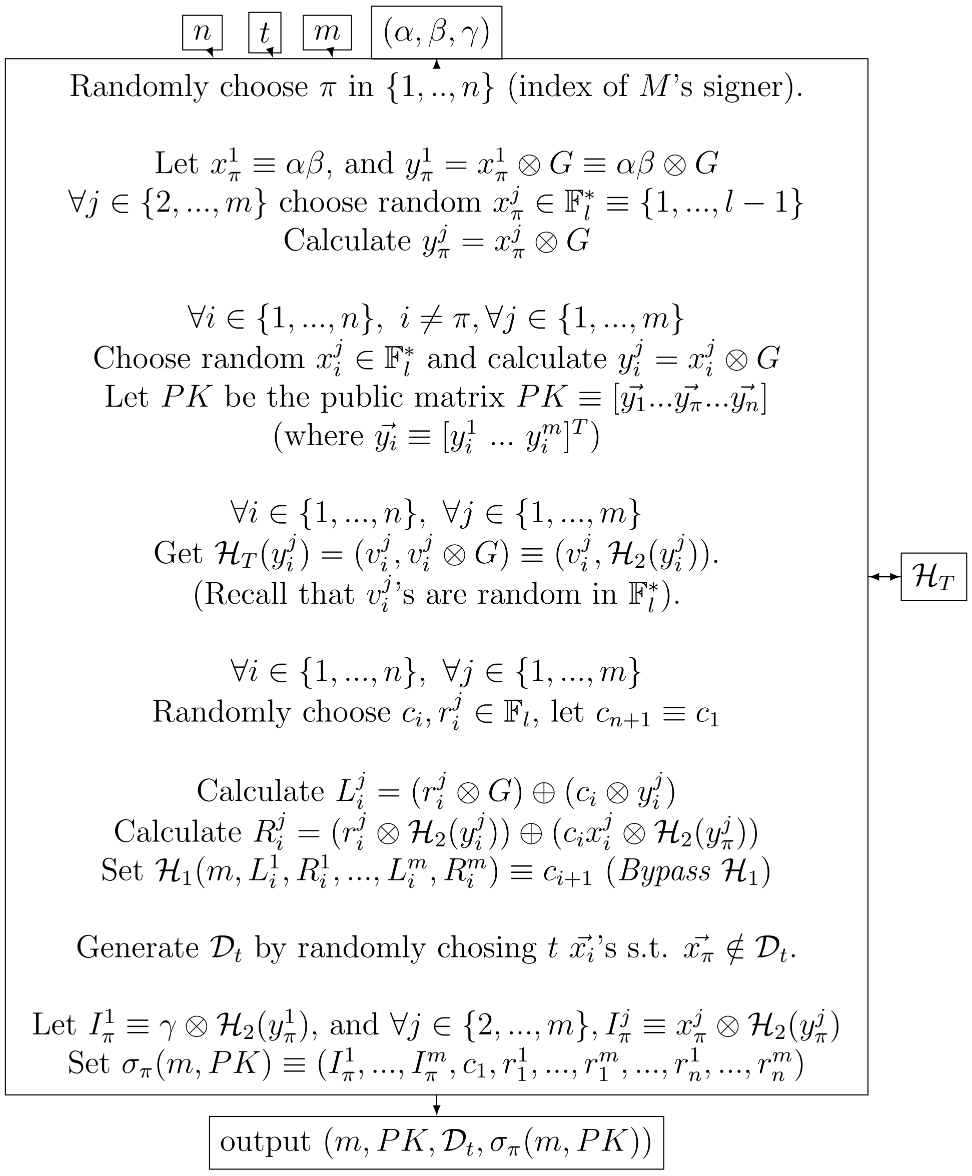
  Suppose that $\exists$ $\mathcal{A}(\omega)$ in PPT(k) and $\epsilon(k)$ non-negligible in k such that

$$P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m, PK, \mathcal{D}_t, \sigma_\pi(m, PK)) = \pi \mid \sigma_\pi(m, PK) \textit{ is valid}] > \frac{1}{n-t} + \epsilon(k)$$
$$\text{if } \vec{x_\pi} \notin \mathcal{D}_t \text{ and } 0 \leq t < n-1$$

Recall that since $\vec{x_\pi} \notin \mathcal{D}_t$, one can automatically rule out all the compromised ring members as possible signers (the logic is similar to what was described in the anonymity section of part 5). One can then limit the guessing range of the identity of the signer to the uncompromised batch of $(n-t)$ remaining members.

We now build $M \in \text{PPT}(k)$ that colludes with $\mathcal{A}(\omega)$ to solve the DDH problem. M's input consists of 1) The tuple (α, β, γ) being tested for DDH, 2) A certain ring size n (randomly chosen), 3) A number $0 \leq t < n-1$ of compromised members (randomly chosen), and 4) A message m (randomly chosen).

M outputs a tuple consisting of 1) The message m, 2) A randomly generated public key matrix $PK \in (\{G\}^*)^{(m \times n)}$, 3) A randomly chosen set $\mathcal{D}_t$ of t compromised secret key vectors, and 4) A not-necessarily valid signature $\sigma_\pi(m, PK)$ assigned to ring member π s.t. $\vec{x_\pi} \notin \mathcal{D}_t$. We let M do the following:


$\boxed{n}$ $\boxed{t}$ $\boxed{m}$ $\boxed{(\alpha, \beta, \gamma)}$

Randomly choose π in $\{1, .., n\}$ (index of M's signer).

Let $x_\pi^1 \equiv \alpha\beta$, and $y_\pi^1 = x_\pi^1 \otimes G \equiv \alpha\beta \otimes G$
$\forall j \in \{2, ..., m\}$ choose random $x_\pi^j \in \mathbb{F}_l^* \equiv \{1, ..., l-1\}$
Calculate $y_\pi^j = x_\pi^j \otimes G$

$\forall i \in \{1, ..., n\},\ i \neq \pi, \forall j \in \{1, ..., m\}$
Choose random $x_i^j \in \mathbb{F}_l^*$ and calculate $y_i^j = x_i^j \otimes G$
Let PK be the public matrix $PK \equiv [\vec{y_1} ... \vec{y_\pi} ... \vec{y_n}]$
(where $\vec{y_i} \equiv [y_i^1 \ ...\ y_i^m]^T$)

$\forall i \in \{1, ..., n\},\ \forall j \in \{1, ..., m\}$
Get $\mathcal{H}_T(y_i^j) = (v_i^j, v_i^j \otimes G) \equiv (v_i^j, \mathcal{H}_2(y_i^j))$.
(Recall that v_i^j's are random in $\mathbb{F}_l^*$).

$\forall i \in \{1, ..., n\},\ \forall j \in \{1, ..., m\}$
Randomly choose $c_i, r_i^j \in \mathbb{F}_l$, let $c_{n+1} \equiv c_1$

Calculate $L_i^j = (r_i^j \otimes G) \oplus (c_i \otimes y_i^j)$
Calculate $R_i^j = (r_i^j \otimes \mathcal{H}_2(y_i^j)) \oplus (c_i x_i^j \otimes \mathcal{H}_2(y_\pi^j))$
Set $\mathcal{H}_1(m, L_i^1, R_i^1, ..., L_i^m, R_i^m) \equiv c_{i+1}$ (*Bypass* $\mathcal{H}_1$)

Generate $\mathcal{D}_t$ by randomly chosing t $\vec{x_i}$'s s.t. $\vec{x_\pi} \notin \mathcal{D}_t$.

Let $I_\pi^1 \equiv \gamma \otimes \mathcal{H}_2(y_\pi^1)$, and $\forall j \in \{2, ..., m\}, I_\pi^j \equiv x_\pi^j \otimes \mathcal{H}_2(y_\pi^j)$
Set $\sigma_\pi(m, PK) \equiv (I_\pi^1, ..., I_\pi^m, c_1, r_1^1, ..., r_1^m, ..., r_n^1, ..., r_n^m)$

$\boxed{\mathcal{H}_T}$

$\boxed{\text{output } (m, PK, \mathcal{D}_t, \sigma_\pi(m, PK))}$

M feeds its output $(m, PK, \mathcal{D}_t, \sigma_\pi(m, PK))$ to $\mathcal{A}(\omega)$. In order for $A(\omega)$ to use its advantage in guessing the signer's identity, it must be given a valid signature (i.e., a signature that is an element of the range of acceptable signatures over all RO $\mathcal{H}_1$. For $\sigma_\pi(m, PK)$ to be a valid signature, $(G, \alpha \otimes G, \beta \otimes G, \gamma \otimes G)$ must be a DDH instance. Indeed, let $\mathcal{H}_1$ be partially defined as per the design of M. We show that for such an $\mathcal{H}_1$, the signature obtained is an element of the range of acceptable signatures. First note the following (all the primed quantities below are as defined in the verification algorithm $\mathcal{V}$):

If $(\gamma = \alpha\beta) \cap (c'_i = c_i) \cap ((L_i^j)' = L_i^j) \cap ((R_i^j)' = R_i^j), \ \forall j \in \{1, ., m\}$ then:

$\{ \ c'_{i+1} = \mathcal{H}_1(m, (L_i^1)', (R_i^1)', ., (L_i^m)', (R_i^m)') \pmod{l} = \mathcal{H}_1(m, L_i^1, R_i^1, ., L_i^m, R_i^m) \pmod{l} = c_{i+1}$

$\{ \ \forall j \in \{1, ., m\}, \ (L_{i+1}^j)' \equiv (r_{i+1}^j \otimes G) \oplus (c'_{i+1} \otimes y_{i+1}^j) = (r_{i+1}^j \otimes G) \oplus (c_{i+1} \otimes y_{i+1}^j) = L_{i+1}^j$

$\{ \ \forall j \in \{1, ., m\}, \ (R_{i+1}^j)' \equiv (r_{i+1}^j \otimes \mathcal{H}_2(y_{i+1}^j)) \oplus (c'_{i+1} \otimes I_\pi^j) =$

$\{ \ (r_{i+1}^1 \otimes \mathcal{H}_2(y_{i+1}^1)) \oplus (c_{i+1}\alpha\beta \otimes \mathcal{H}_2(y_\pi^1)) = R_{i+1}^1, \text{ if } j = 1$

$\{ \ (r_{i+1}^j \otimes \mathcal{H}_2(y_{i+1}^j)) \oplus (c_{i+1}x_i^j \otimes \mathcal{H}_2(y_\pi^j)) = R_{i+1}^j, \text{ if } j \neq 1$

Since $(G, \alpha \otimes G, \beta \otimes G, \gamma \otimes G)$ is a DDH instance then we necessarily have $\gamma = \alpha\beta$

Moreover, recall that $c'_1 = c_1$ (by design of $\mathcal{V}$). And so $\forall j \in \{1, ..., m\}, \ (L_1^j)' = L_1^j$ and $(R_1^j)' = R_1^j$. We therefore conclude by induction on c'_i that $\forall i \in \{1, ..., n+1\}$, $c'_i = c_i$. In particular, $c'_{n+1} = c_{n+1} \equiv c_1$. This in turn implies that $\sigma_\pi(m, PK)$ is a valid signature.

On the other hand, if $(G, \alpha \otimes G, \beta \otimes G, \gamma \otimes G)$ is not a DDH instance, then $\forall j \in \{1, ..., m\}, \ R_i^j \neq (R_i^j)'$ and with overwhelming probability $\sigma_\pi(m, PK)$ is not a valid signature.

Recall that $\mathcal{A}(\omega)$ can send queries to $\mathcal{H}_1$ and $\mathcal{H}_T$ during execution. It is important to enforce consistency between M and $\mathcal{A}(\omega)$'s query results obtained from RO $\mathcal{H}_1$ and RO $\mathcal{H}_T$ on the same input. There are no risks of faulty collisions in so far as $\mathcal{H}_T$ is concerned (by design of M). However, M bypasses RO $\mathcal{H}_1$ and conducts its own backpatching to $\mathcal{H}_1(m, L_i^1, R_i^1, ., L_i^m, R_i^m), \ \forall i \in \{1, ..., n\}$. If $\exists i \in \{1, ..., n\}$ such that $\mathcal{A}(\omega)$ queries $\mathcal{H}_1$ on input $(m, L_i^1, R_i^1, ., L_i^m, R_i^m)$, then with overwhelming probability, it will conflict with M's backpatched value causing the execution to halt. The aforementioned collision must be avoided. In order to do so, we first calculate the probability of its occurence. We assume that during execution, $\mathcal{A}(\omega)$ can make a maximum of Q_1 queries to RO $\mathcal{H}_1$. Q_1 is assumed to be polynomial in the security parameter k, since the adversary is modeled as a PPT Turing machine.

$$P[Col] \;= P[\cup_{all\ (m,L_i^1,R_i^1,.,L_i^m,R_i^m),\ i\in\{1,...,n\}}\{(m,L_i^1,R_i^1,.,L_i^m,R_i^m)\ appeared\ in\ M\ and\ in\ at\ least\ one\ of\ the\ Q_1\ queries\ to\ RO\ \mathcal{H}_1\}]$$

$$\leq\ \sum_{i=1}^{n} P[\cup_{all\ L_i}\{L_i\ appeared\ in\ M\ and\ was\ part\ of\ at\ least\ one\ of\ the\ Q_1\ queries\ to\ RO\ \mathcal{H}_1\}]$$

$$\leq \sum_{i=1}^{n}\sum_{all\ L_i\in\{G\}} P[\cup_{(j=1,..,Q_1)}\{L_i\ appeared\ in\ M\ and\ was\ part\ of\ at\ least\ the\ j^{th}\ query\ to\ RO\ \mathcal{H}_1\}]$$

$$\leq \sum_{i=1}^{n}\sum_{all\ L_i\in\{G\}}\sum_{j=1}^{Q_1} P[L_i\ appeared\ in\ M\ and\ was\ part\ of\ at\ least\ the\ j^{th}\ query\ to\ RO\ \mathcal{H}_1]$$

$$\leq \sum_{i=1}^{n}\sum_{all\ L_i\in\{G\}}\sum_{j=1}^{Q_1}\frac{1}{|\{G\}|^2} \;= n|\{G\}|\times\frac{Q_1}{|\{G\}|^2} \;= \frac{nQ_1}{|\{G\}|} < \frac{nQ_1}{2^k}.$$

$$(since\ k < log_2(|\{G\}^*|) < log_2(|\{G\}|)\ by\ design).$$

and so we conclude that:

$$P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[(\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m,PK,\mathcal{D}_t,\sigma_\pi(m,PK))=\pi)\ \cap\ \overline{Col}\mid\sigma_\pi(m,PK)\ is\ valid] =$$

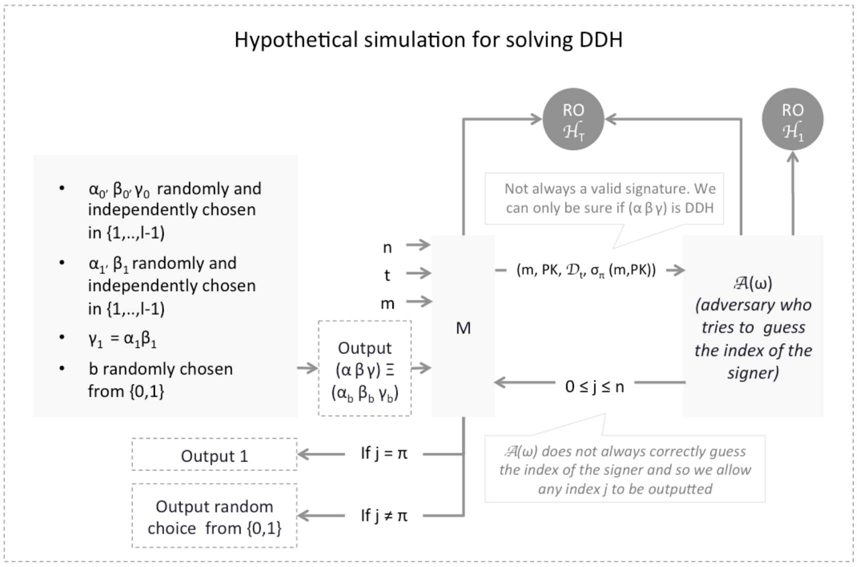
$$P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m,PK,\mathcal{D}_t,\sigma_\pi(m,PK))=\pi\mid\sigma_\pi(m,PK)\ is\ valid] - P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[(\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m,PK,\mathcal{D}_t,\sigma_\pi(m,PK))=\pi)\ \cap\ Col\mid\sigma_\pi(m,PK)\ is\ valid]$$

$$> P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m,PK,\mathcal{D}_t,\sigma_\pi(m,PK))=\pi\mid\sigma_\pi(m,PK)\ is\ valid]\ -\ P[Col]$$

$$> P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m,PK,\mathcal{D}_t,\sigma_\pi(m,PK))=\pi\mid\sigma_\pi(m,PK)\ is\ valid]\ -\ \frac{nQ_1}{2^k}$$

$$> \frac{1}{n-t}+\nu(k).\ \text{Here, } \nu(k)\equiv\epsilon(k)-\frac{nQ_1}{2^k}\ \text{non-negligibale in } k$$

After execution, $\mathcal{A}(\omega)$ returns to M an integer $1 \leq j \leq n$. M outputs 1 if $j = \pi$, or outputs 0/1 with equal probability otherwise:


Hypothetical simulation for solving DDH
RO $\mathcal{H}_T$
RO $\mathcal{H}_1$
$\alpha_0, \beta_0, \gamma_0$ randomly and independently chosen in {1,..,l-1)
α_1, β_1 randomly and independently chosen in {1,..,l-1)
$\gamma_1 = \alpha_1\beta_1$
b randomly chosen from {0,1}
Output $(\alpha\,\beta\,\gamma) \Xi (\alpha_b\,\beta_b\,\gamma_b)$
n
t
m
M
Not always a valid signature. We can only be sure if $(\alpha\,\beta\,\gamma)$ is DDH
$(m, PK, \mathcal{D}_t, \sigma_\pi (m,PK))$
$\mathcal{A}(\omega)$ (adversary who tries to guess the index of the signer)
$0 \leq j \leq n$
$\mathcal{A}(\omega)$ does not always correctly guess the index of the signer and so we allow any index j to be outputted
Output 1
If $j = \pi$
Output random choice from {0,1}
If $j \neq \pi$

Using the setting described above, one can now calculate the probability of M guessing whether $(G, \alpha \otimes G, \beta \otimes G, \gamma \otimes G)$ is DDH or not. The calculation is the same as the one previously conducted in section 6 of parts 5 and 6 and leads us to conclude:

$$P[M(G, \alpha \otimes G, \beta \otimes G, \gamma \otimes G) = b] \geq \frac{1}{2} + \frac{\nu(k)}{4}$$

Since $\nu(k)$ is non-negligible in k, the above probability outperforms random guessing. This contradicts the intractability of DDH. Similarly, we can show $P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m, PK, \mathcal{D}_t, \sigma_\pi(m, PK)) = \pi] \mid \sigma_\pi(m, PK)\ is\ valid]$ is also bounded from below. We finally conclude that for any polynomial $Q(k)$:

$$\frac{1}{n-t} - \frac{1}{Q(k)} \leq P_{\omega,\mathcal{H}_1,\mathcal{H}_T}[\mathcal{A}^{\mathcal{H}_1,\mathcal{H}_T}(\omega)(m, PK, \mathcal{D}_t, \sigma_\pi(m, PK)) = \pi \mid \sigma_\pi(m, PK)\ is\ valid] \leq \frac{1}{n-t} + \frac{1}{Q(k)}$$

- Case 2: $\vec{x_\pi} \notin \mathcal{D}_t$ and $t = n - 1$.

  In this case, $\mathcal{A}(\omega)$ can check if the key-image vector $\vec{I_\pi} \equiv [I_\pi^1\ ...\ I_\pi^m]^T$ appearing in $\sigma_\pi(m, PK)$ matches the key-image vector of any of the compromised users (i.e., $[\hat{x_i^1} \otimes \mathcal{H}_2(\hat{x_i^1})\ ...\ \hat{x_i^m} \otimes \mathcal{H}_2(\hat{x_i^m})]^T$, for $i \in \{1, ., t = (n-1)\}$). With overwhelming probability, none of them will match since we proved that the scheme is exculpable and so no one can forge a signature with a tag of a non-compromised member. Proceeding by elimination, $\mathcal{A}(\omega)$ can then conclude that the signer is π.

- Case 3: $\vec{x_\pi} \in \mathcal{D}_t$.

  In this case, $\mathcal{A}(\omega)$ can check which of the compromised key-image vectors (i.e., $[\hat{x_i^1} \otimes \mathcal{H}_2(\hat{x_i^1})\ ...\ \hat{x_i^m} \otimes \mathcal{H}_2(\hat{x_i^m})]^T$, for $i \in \{1, .., t\}$) matches the key-image vector appearing in $\sigma_\pi(m, PK)$). Only one of them will match (due to exculpability), subsequently revealing the identity of the signer.

7 Security analysis - Linkability

The *linkability* property means that if a secret key of a given secret key vector is used in more than one signature, then the resulting signatures will be linked and flagged by $\mathcal{L}$ (the linkability algorithm).

We proved in part 5 of this series that a signature scheme is linkable if and only if $\forall n \in \{1, .., l-1\}, \forall L \equiv \{y_1, .., y_n\}$ a ring of n members, it is not possible to produce $(n+1)$ valid signatures with pairwise different key-images such that all of them get labeled *independent* by $\mathcal{L}$. This result can be easily adapted to the case of a public key matrix $PK \equiv [\vec{y_1}...\vec{y_n}]$ (where $\vec{y_i} \equiv [y_i^1\ ...\ y_i^m]^T,\ i \in \{1, ..., n\}$) and secret key vectors $\{\vec{x_1}, ..., \vec{x_n}\}$:

A signature scheme is linkable

$$\Longleftrightarrow$$

$\forall n \in \{1,..,l-1\}, \forall PK \equiv [\vec{y_1}...\vec{y_n}]$ a ring of n members, it is not possible to produce $(n+1)$ valid signatures such that no 2 key image vectors share a component in common.

The proof of the above equivalence is identical to the one previously outlined in section 7 of part 5. To prove that the MLSAG scheme is linkable we follow a *reductio ad absurdum* approach, similar to the one described in part 5:

- Assume that the MLSAG signature scheme is not linkable.
- The equivalence above would imply that $\exists PK \equiv [\vec{y_1}...\vec{y_n}]$ such that it can produce $(n+1)$ valid signatures such that no 2 key image vectors share a component in common. This means that

$$\forall i,u \in \{1,..,n\},\ \forall j,v \in \{1,...,m\},$$
$$i \neq u \Rightarrow (I_i^j \equiv x_i^j \otimes \mathcal{H}_2(y_i^j)) \neq (I_u^v \equiv x_u^v \otimes \mathcal{H}_2(y_u^v))$$

- This implies that there must exist a signature (from the set of $(n+1)$ valid signatures) with key-image vector $\vec{I_\delta} \equiv [I_\delta^1\ ...\ I_\delta^m]^T$ for which $\exists k \in \{1,...,m\}$ such that

$$\forall i \in \{1,..,n\},\ \forall j \in \{1,...,m\},$$
$$I_\delta^k \neq [I_i^j \equiv x_i^j \otimes \mathcal{H}_2(y_i^j)]$$

  Denote this signature by $\sigma_\delta \equiv (I_\delta^1, ..., I_\delta^m, c_1, r_1^1, .., r_1^m, ..., r_n^1, ..., r_n^m)$.

- When verifying the validity of σ_δ, $\mathcal{V}$ first computes the following:

```
{ Let $c_1' = c_1$

{ $\forall i \in \{1,..,n\}$:

    { $\forall j \in \{1,...,m\}$, compute:

        { $(L_i^j)' \equiv (r_i^j \otimes G)\ \oplus\ (c_i' \otimes y_i^j)$

        { $(R_i^j)' \equiv (r_i^j \otimes \mathcal{H}_2(y_i^j))\ \oplus\ (c_i' \otimes I_\delta^j)$

    { $c_{i+1}' \equiv \mathcal{H}_1(m, (L_i^1)', (R_i^1)', ..., (L_i^m)', (R_i^m)') \pmod{l}$
```

- $\forall i \in \{1,...,n\}$, and for k as identified above, the system of 2 equations given by $(L_i^k)'$ and $(R_i^k)'$ can be equivalently written as:

  { $r_i^k + c_i' x_i^k = log_G((L_i^k)')$
  { $r_i^k \otimes \mathcal{H}_2(y_i^k)) \oplus (c_i' \otimes I_\delta^k) = (R_i^k)'$

  For a given $(L_i^k)'$, $(R_i^k)'$, and $i \in \{1,..,n\}$, this constitutes a system of 2 equations in variables r_i^k and c_i'.

- Since $\forall i \in \{1, .., n\}$, $\forall j \in \{1, ..., m\}$, $I_\delta^k \neq x_i^j \otimes \mathcal{H}_2(y_i^j)$, the system of 2 equations corresponding to each i is independent and admits a unique solution $((r_i^k)^*, (c_i')^*)$ for any given $(L_i^k)'$ and $(R_i^k)'$. In particular, that means that the value $c_1' \equiv c_1$ is well defined and equal to $(c_1')^*$.

- By virtue of being a valid signature, σ_δ must satisfy $\mathcal{V}$'s verification equation requiring that

  $$c_1 = c_{n+1}' \equiv \mathcal{H}_1(m, (L_n^1)', (R_n^1)', ., (L_n^m)', (R_n^m)') \pmod{l}$$

  But RO $\mathcal{H}_1$ is random by definition. The probability that it outputs a specific value is eqal to $\frac{1}{q}$ (recall that the range of $\mathcal{H}_1 = \mathbb{F}_q$). Since by design we have $2^k < l - 1 < l < q$, we conclude that the probability that

  $$\mathcal{H}_1(m, (L_n^1)', (R_n^1)', ., (L_n^m)', (R_n^m)') \pmod{l} = (c_1')^*$$

  is upper-bounded by $\frac{1}{2^k}$ and is hence negligible. In other terms, the probability that σ_δ is a valid signature is negligible.

We hence conclude that the MLSAG scheme is linkable.

References

[1] E. Fujisaki and K. Suzuki. Traceable ring signatures. *Public Key Cryptography*, pages 181–200, 2007.

[2] S. Noether and A. Mackenzie. Ring confidential transactions. *Monero Research Lab*, 2016.