Borromean Signatures

Sammy, Hao Xu

2017-09-21

1 Concrete Algorithm

All the works listed in this section is from [MP16].

1.1 Signing

Suppose a signer has a collection of verification keys $P_{i,j}$ for $0 \le i \le n-1$, $0 \le j \le m_i - 1$, and wants to create a signature of knowledge of the n keys $\{P_{i,j_i^*}\}_{i=0}^{n-1}$ where the j_i^*'s are some fixed and unknown (to a verifier) indices. Denote the secret key to P_{i,j_i^*} by x_i. He acts as follows:

1. Compute M as the hash of the message to be signed and the set of verification keys.
2. For each $0 \le i \le n-1$:
   (a) Choose a scalar k_i uniformly at random.
   (b) Set $R_{i,j^*} = k_i G$.
   (c) For j such that $j_i^* + 1 \le j \le m_i - 1$ choose $s_{i,j}$ at random and compute

$$\begin{aligned} e_{i,j} &= H(M \| R_{i,j-1} \| i \| j-1) && (1) \\ R_{i,j} &= s_{i,j} G - e_{i,j} P_{i,j} && (2) \end{aligned}$$

3. Finally, set

$$e_{i,0} = H(R_{0,m_0-1} \| \cdots \| R_{n-1,m_{n-1}-1})$$

   That is, $e_{i,0}$ commits to several (s, P) pairs, one from each ring.

4. For each $0 \le i \le n-1$:
   (a) Let $R_{i,-1} = R_{i,m_i-1}$
   (b) For j such that $0 \le j \le j_i^* - 1$ choose $s_{i,j}$ at random and compute

$$\begin{aligned} R_{i,j} &= s_{i,j} G - e_{i,j} P_{i,j} && (3) \\ e_{i,j+1} &= H(M \| R_{i,j} \| i \| j) && (4) \end{aligned}$$

   Note that this calculation is identical to the one in Step 2c.

   (c) To wrap around making $R_{i,j_i^*} = s_{i,j} G - e_{i,j_i^*} P_{i,j_i^*}$, we should set

$$s_{i,j_i^*} = k_i + x_i e_{i,j_i^*} \quad (5)$$

The resulting signature on m consists of

$$\sigma = \{e_0, s_{i,j} \mid 0 \le i \le n-1, 0 \le j \le m_i - 1\} \quad (6)$$

where e_0 means any of $e_{i,0}$. We should publish

$$\{M, \{P_{i,j}\}, \sigma \mid 0 \le i \le n-1, 0 \le j \le m_i - 1\} \quad (7)$$

1.2 Verification

Since verification does not depend on which specific keys are known, it avoids the "two-phase" structure of signing and is therefore much simpler.

We assume we have a message m, a collection $\{P_{i,j}\}$ of verification keys whose indices range as before, and a signature σ whose notation is the same as before. The verifier acts as follows:

1. Compute M as the hash of the message to be signed and the set of verification keys.
2. For each $0 \le i \le n-1$,
   (a) For each $0 \le j \le m_i - 1$, compute
   $$R_{i,j} = s_{i,j}G - e_{i,j}P_{i,j} \tag{8}$$
   $$e_{i,j+1} = H(M \| R_{i,j} \| i \| j) \tag{9}$$
   (As before, we always take $e_{i,0}$ to be e_0.)
3. Compute
   $$e'_0 = H(M \| R_{0,m_0-1} \| \cdots \| R_{n-1,m_{n-1}-1}) \tag{10}$$
   and return 1 iff $e'_0 \stackrel{?}{=} e_0$.

A visualization of the whole scheme is as Figure 1

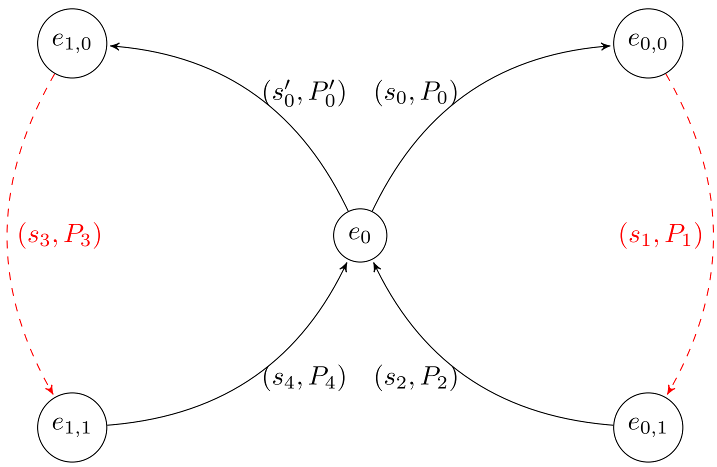

Figure 1: A Borromean ring signature for $(P_0|P_1|P_2)\&(P'_0|P_3|P_4)$

2 Implementation in Monero

Summarized from the official codebase [SML].

References

[MP16] Gregory Maxwell and Andrew Poelstra. *Borromean Ring Signatures*. 2016. URL: https://github.com/ElementsProject/borromean-signatures-writeup.

[SML] Noether Shen, Adam Mackenzie, and The Monero Lab. *monero*. URL: https://github.com/monero-project/monero.