RING CT FOR MONERO

SHEN-NOETHER MRL

Abstract. In this note, I give a modification of gmaxwell's Confidential Transactions to ring signatures. This modification will work with either the Fujisaki/ Suzuki Ring signatures currently used in Monero, or the LLW signatures which have been proposed as a modification to Monero (c.f. mrl_notes v1,2,3). (In fact, I give a slight generalization to either of these schemes tentatively titled a "Mokume-Gane" signature. Example code in python is provided at [SN]. The security and anonymity proofs of this scheme are given in the random oracle model. Note that I created the basic description for this protocol shortly after the original confidential transactions were announced (see mrl_notes v.3 in [SN]), and the new content in this note are the security (coming soon, although it's not much a big expansion from Fujisaki Suzuki) and anonymity proofs under the random oracle model.

1. Introduction

The necessary language and definitions are taken from [FS, LWW, GM], and will be copied here later.

2. Protocol Description

Let G be the ed25519 basepoint. Let[1]

$$H = toPoint\left(cn\_fast\_hash\left(123456 \cdot G\right)\right)$$

Note on the choice of scalar 123456. In the curve group of ed25519, not every cn_fast_hash is itself a point in the group of the basepoint G. The scalar 123456 is chosen so that the hash is a point in the group of the basepoint, so that all the usual elliptic curve math holds. Under the discrete logarithm assumption on ed25519, the probability of finding an x such that $xG = H$ is negligible.

Define $C\left(a, x\right) = xG + aH$, the commitment to the value a with mask x. Note that as long as $log_G H$ is unknown, and if $a \neq 0$, then $log_G C\left(a, x\right)$ is unknown. On the other hand, if $a = 0$, then $log_G C\left(a, x\right) = x$, so it is possible to sign with sk-pk keypair $\left(x, C\left(0, x\right)\right)$.

In [**?**], there are input commitments, output commitments, and the network checks that

$$\sum Inputs = \sum Outputs.$$

However, this does not suffice in Monero: Since a given transaction contains multiple possible inputs $P_i, i = 1, ..., n$, only one of which belong to the sender, (see [CN, 4.4]), then if we are able to check the above equality, it must be possible for the network to see which P_i belongs to the sender of the transaction. This is undesirable, since it removes the anonymity provided by the ring signatures. Thus instead,

[1] $H = MiniNero.getHForCT()$

commitments for the inputs and outputs are created as follows (suppose first that there is only one input)

$$C_{in} = x_c G + aH$$

$$C_{out-1} = y_1 G + b_1 H$$

$$C_{out-2} = y_2 G + b_2 H$$

such that $x_c = y_1 + y_2 + z$, $x_c - y_1 - y_2 = z$, y_i are mask values, $z > 0$ and $a = b_1 + b_2$. Here x_c is a special private key the "amount key" known only to the sender, and to the person who sent them their coins, and must be different than their usual private key). In this case,

$$C_{in} - \sum_{i=1}^{2} C_{out-i}$$

$$= x_c G + aH - y_1 G - b_1 H - y_2 G - b_2 H$$

$$= zG.$$

Thus, the above summation becomes a commitment to 0, with $sk = z$, and $pk = zG$, rather than an actual equation summing to zero. Note that z is not computable to the originator of x_c's coins, unless they know both of the y_1, y_2, but then they are receiving the coins, and presumably remember which pubkey they sent them to originally, and so there is no additional unmasking.

Since it is undesirable to show which input belongs to the sender, a ring signature consisting of all the input commitments $C_i, i = 1, ..., s, ..., n$ (where s is the secret index of the commitment of the sender), adding the corresponding pubkey (so commitments and pubkeys are paired

(C_i, P_i) only being allowed to be spent together) and subtracting $\sum C_{out}$ is created:

$$\left\{P_1 + C_{1,in} - \sum_j C_{j,out}, ..., P_s + C_{s,in} - \sum_j C_{j,out}, ..., P_n + C_{n,in} - \sum_j C_{j,out}\right\}.$$

This is a ring signature which can be signed since we know one of the private keys (namely $z + x'$ with z as above and $x'G = P_s$). In fact, since we know, for each i both the private key for P_i and the private key for $P_i + C_{i,in} - \sum_j C_{j,out}$ we can perform a signature as in section 3.2.

As noted in [GM], it is important to prove that the output amounts [2] $b_1, ...b_n$ all lie in a range of positive values, e.g. $(0, 2^{16})$. This can be accomplished essentially the same way as in [GM]:

- Prove first $C^{(j)}_{out-i} \in \{0, 2^j\}$ for all $j \in \{0, 1, ..., 16\}$. This is done as in [GM]: for example, $C^0_{out-i} = y^0_i G + b^0_i H$ where $b^0_i \in \{0, 1\}$. Let
$$C'^0_{out-i} = C^0_{out-i} - H = y_i G + b^0_i H - H$$
so that if $b^0_i = 0$, then $C'^0_{out} = y_i G$ and if $b^0_i = 1$, then $C^0_{out} = y_i G$, and in either case, the ring signature on$\{C^0_{out}, C'^0_{out}\}$ can be signed for.
  - Note that $\sum_j y^j_i = y_i$
- By carefully choosing the blinding values for each j, ensure that
$$\sum_{j=1}^{16} C^{(j)}_{out-i} = C_{out-i}.$$

[2] since input commitments could potentially be just inherited from the previous transaction, it suffices to consider the output amounts

- By homomorphicity of the commitments, $b_i = \sum_j \delta_{ji} 2^j$, where δ_{ji} is the j^{th} digit in the binary expansion of b_i.

Thus in total, by the above, the sum of inputs into a transaction equals the outputs, yet the specific input (and it's index!) is hidden. In addition, the outputs are positive values.

3. Mokume-Gane Signatures

In this section, I introduce a new type of ring signature, a Mokume-Gane[3] which is an extension of [LWW], but which has multiple layers, each of which must be completed by the signer. If an ordinary ring signature is the statement "one of these n people signed this," then a Mokume-Gane signature is the statement "one of these n people signed m things." In my brief search, I am not aware of any existing signature which does this (my usual area of research is in algebraic geometry), though it may exist, and in that case, this is a reinvention, created to deal with sweeping transactions in the Ring CT protocol."

3.1. **LWW Signatures.**

First I recall the method of [LWW]: This description is copied from an explanation (reinvention?) given by Adam Back in a bitcointalk.org. As of this draft (September 2015) an example implementation appears in the MiniNero repository at github.com/ShenNoether/MiniNero.

In the current Monero ring signature, which is the same as it's parent protocol (CryptoNote), ring signatures follow [CN, 4.4]. The proposed alternative algorithm is the four following steps:

[3]name credit fluffypony

Keygen: Find a number of public keys $P_i, i = 0, 1, ..., n$ and a secret index j such that $xG = P_j$ where G is the ed25519 basepoint and x is the signers spend key. Let $I = xH(P_j)$ where H is a hash function returning a point (in practice $toPoint(Keccak(P_k))$).

SIGN: Let $\alpha, s_i,\ i \neq j,\ i \in \{1, ..., n\}$ be random values in $\mathbb{Z}_q$ (the ed25519 base field).

Compute

$$L_j = \alpha G$$

$$R_j = \alpha H(P_j)$$

$$c_{j+1} = h(P_1, ..., P_n, L_j, R_j)$$

where h is a hash function returning a value in $\mathbb{Z}_q$. Now, working successively in j modulo n, define

$$L_{j+1} = s_{j+1}G + c_{j+1}P_{j+1}$$

$$R_{j+1} = s_{j+1}H(P_{j+1}) + c_{j+1} \cdot I$$

$$c_{j+2} = h(P_1, ..., P_n,\ L_{j+1},\ R_{j+1})$$

$$\dots$$

$$L_{j-1} = s_{j-1}G + c_{j-1}P_{j+1}$$

$$R_{j-1} = s_{j-1} H\left(P_{j-1}\right) + c_{j-1} \cdot I$$

$$c_j = h\left(P_1, ..., P_n, L_{j-1},\ R_{j-1}\right)$$

so that $c_1, ..., c_n$ are defined.

Let $s_j = \alpha - c_j \cdot x\ mod\ l$, ($l$ being the ed25519 curve order) hence $\alpha = s_j + c_j x\ mod\ l$ so that

$$L_j = \alpha G = s_j G + c_j x G = s_j G + c_j P_j$$

$$R_j = \alpha H\left(P_j\right) = s_j H\left(P_j\right) + c_j I$$

and

$$c_{j+1} = h\left(P_1, ..., P_n,\ L_j,\ R_j\right)$$

and thus, given a single c_i value, the P_j values, the key image I, and all the s_j values, all the other $c_k,\ k \neq i$ can be recovered by an observer. The signature therefore becomes:

$$\sigma = (I, c_1, s_1, ..., s_n)$$

which represents a space savings over [CN, 4.4].

Verification proceeds as follows. An observer computes L_i, R_i, and c_i for all i and checks that $c_{n+1} = c_1$. Then the verifier checks that

$$c_{i+1} = h\left(P_1, ..., P_n, L_i, R_i\right)$$

for all i.

LINK: Signatures with duplicate key images I are rejected.

3.2. **Mokume-Gane Signatures.** Now suppose that each signer of a (generalized) ring containing n members has exactly m keys $\left\{P_i^j\right\}_{j=1,...,m}^{i=1,...,n}$. The intent of the Mokume-Gane ring signature is the following:

- Exactly one of the n signers has given a signature on all m of their keys.
- If the signer uses any one of their m keys in another Mokume-Gane signature, then the two rings are linked.

The algorithm proceeds as follows: Let π be a secret index corresponding to the signer of the generalized ring. For $j = 1, ..., m$, let $I_j = x_j H\left(P_\pi^j\right)$, and for $j = 1, ..., m$, $i = 1, ..., \hat{\pi}, ...n$ (where $\hat{\pi}$ means omit the index π) let s_i^j be some random scalars. Now, in an analogous manner to section 3.1, define

$$L_\pi^j = \alpha_j G$$

$$R_\pi^j = \alpha_j H\left(P_\pi^j\right)$$

random scalars $\alpha_j, j = 1, ..., m$. Now, again analogously to section 3.1, set:

$$c_{\pi+1} = H\left(P_j | \sum_j \left(L_\pi^j + R_\pi^j\right)\right).$$

$$L_{\pi+1}^j = s_{\pi+1}^j G + c_{\pi+1} P_{\pi+1}^j$$

$$R_\pi^j = s_{\pi+1}^j H\left(P_{\pi+1}^j\right) + c_{\pi+1} I_j$$

and repeat this incrementing i modulo n until we arrive at

$$L_{\pi-1}^j = s_{i-1}^j G + c_{i-1} P_{i-1}^j$$

$$R^j_{\pi-1} = s^j_{i-1} H\left(P^j_{i-1}\right) + c_{i-1} \cdot I_j$$
$$c_\pi = h\left(P^j_i | \sum_j \left(L^j_{\pi-1} + R^j_{\pi-1}\right)\right).$$

Finally, solve for each s^j_π using $\alpha_j = s^j_\pi + c_\pi x_j \; mod \; q$. Now the signature is given as $(I_1, ..., I_m, c_1, s^1_1, ..., s^m_1, s^1_2, ..., s^m_2, ..., s^1_n, ..., s^m_n)$, so the complexity is $O\left(m\left(n+1\right)\right)$. Now verification proceeds by regenerating all the L^j_i, R^j_i starting from $i = 1$ as in section 3.1(where $m = 1$) and verifying the hash $c_{n+1} = c_1$. One can easily show, in a manner similar to [LWW]:

- The probability of a signer generating a valid signature without knowing all "m" private keys for index π is negligible.
- The probability of a signer signing for keys not of index π is negligible.
- If a signer signs two rings using at least one of the same public keys, then the two rings are linked.

4. Anonymity

To prove the anonymity of the above protocol in the random oracle model, let H_1, H_2 be random oracles modeling discrete hash functions. Let $\mathcal{A}$ be an adversary against anonymity. I construct an adversary $\mathcal{M}$ against decisional diffie helman assumption assumption as follows. (Note, for this proof I use the ring signature style of [FS], rather than the ring signatures of [LWW] for simplicity, in fact the protocol description is independent of the choice of linkable ring signature. Recall that a DDH triple is a tuple of group elements (A, B, C, D)

such that $log_A C = log_B D$ the DDH asumption says that given a tuple $(G, aG, bG, \gamma G)$, the probability of determining whether $\gamma G = abG$ is negligible.

Theorem 1. *Ring CT protocol is anonymous under the random oracle model in a group where the DDH assumption holds.*

Proof. Let (G, aG, bG, abG) a tuple of group elements. Suppose there is an adversary $\mathcal{A}$ against anonymity. I work with signatures of size two for simplicity, though the general case follows in the same manner. Thus given a signature $\left((p_1 + c_{in,1} - c_{out,1} - c_{out,2}, p_2 + c_{in,2} - c_{out,1} - c_{out,2}\right), I, s_1, s_2, c_2, c_2)$, $\mathcal{A}$ is able to determine with non-negligible probabilty ϵ, which index i corresponds to the private key x_i of the signer. Assume that $\mathcal{A}$ does not have access to either $x_{cout,i}$ for at least one output or $x_{c_{in,i}}$ for either input and that $\mathcal{A}$ does not have access to x_{p_i} the private key of P_i for either i.

First I claim that if $\mathcal{A}$ is able to compute the unknown $x_{cout,i}$ or the unknown $x_{c_{in,i}}$, then it is possible to construct an adversary against the discrete logarithm problem (so that clearly there is an adversary against DDH). Without loss of generality assume $\mathcal{A}$ always knows $x_{c_{in,1}}$ and $x_{c_{out,1}}$ but not $x_{c_{in,2}}$ or $x_{cout,2}$. Suppose first that $\mathcal{A}$ is always able to uncover $x_{c_{in,2}}$ with non-negligible probability. Let P a random element of G in the given group satisfying DDH assumption, $P = aG$. Set $a$equal to our mask $x_{c_{in,2}} = x_{c_{out,1}} + x_{c_{out,2}} + a$ as in the Ring CT protocol description. I construct an adversary $\mathcal{M}$ to compute a. Write $c_{out,i} = x_{c_{out,i}}G + y_{out,i}H$. Assume without loss of generality that $\mathcal{A}$ can guess $y_{in,2}$ and $y_{out,2}$ which are the input and output amounts as some

commonly spent amounts (Note, by the properties of the pedersen commitment, $\mathcal{A}$ will not know for certain what they are, but perhaps the possible number of output amounts is much smaller than the security parameter of the group, and so they can try all possible output amounts for a given algorithm of deciding $x_{c_{in,i}}$). Let $c_{in,2} = aG + y_{in,2}H$. Now, as p_2 is known, we subtract p_2 from the equation, so that in the above signature, we have

$$x_{c_{in,2}}G + y_{in,2}H - x_{cout,1}G + y_{out,1}H + x_{cout,2}G + y_{out,2}H.$$

Subtracting the known input and output amounts, this becomes

$$x_{c_{in,2}}G - x_{cout,1}G - x_{cout,2}G.$$

By the protocol description, this is

$$aG$$

and $\mathcal{A}$ knows $x_{c_{in,2}} - x_{c_{out,1}} - x_{cout,2} = a$, then $\mathcal{A}$ can compute the $log_G aG = a$, contradicting the discrete logarithm assumption, thus contradicting DDH assumption. The proof that $\mathcal{A}$ can $x_{c_{out,2}}$ only with negligible probability is similar.

Now I claim if $\mathcal{A}$ is an adversary against anonymity, that there exists an adversary $\mathcal{M}$ against DDH. Let $(G, aG, bG, \gamma G)$ a given tuple of group elements (computed as random scalars and then turned into multiples of the basepoint), and we construct $\mathcal{M}$ to decide whether $\gamma P = abP$ with non-negligible probability.

Define SIM-NIZKP as in [FS] as follows: Let c_1, c_2, s_1, s_2 random scalars. Given P_1, P_2, and keyimage I belonging to one of the P_i, set $L_i = s_i G + c_i P_i$, $R_i = s_i H_1(P_i) + c_i I$. Now (using the random oracle model assumption that the hash functions are determined as random oracles) set $\sum c_i = H_2(m, L_1, L_2, R_1, R_2)$, which is random as the c_i, s_i are random. Under the random oracle assumption $\mathcal{A}$ verifies (I, c_1, c_2, s_2, s_2) as a valid signature. Note that $log_{(s_i+c_i)G} L_i = log_{(s_i+c_i)} H_1(P_i)$ for the index corresponding to the signer.

Compute relevant commitments so that $p_1 + c_{out,1} - c_{in,1} - c_{in,2} = xG$,$(s_1 + c_1 x_1) G = bG$, $(s_1 + c_1 x_1) aG = \gamma G$, and using the random oracle model, $H_2(p_1 + c_{out,1} - c_{in,1} - c_{in,2}) = aG$. Now choose random other $P_1, c_{in,1}$ and feed the result of SIM-NIZKP on $((P_1, c_{in,1}), (P_2, c_{in,2}), c_{out,1}, c_{out,2})$ to $\mathcal{A}$. By assumption that $\mathcal{A}$ is an adversary against anonymity, then $\mathcal{A}$ will output 1 as the signer if $log_G bG = log_{aG} \gamma G$ with non-negligible probability, thus creating an adversary against DDH. □

5. Tag Linkability

6. Exculpability

7. Unforgeability

8. Appendix A: Example Code

Example code can be found in [SN]. (I will include something in the actual writeup later).

References

[CN] van Saberhagen, Nicolas. "Cryptonote v 2. 0." HYPERLINK "https://cryptonote. org/whitepaper. pdf" https://cryptonote. org/whitepaper. pdf (2013).

[FS] Fujisaki, Eiichiro, and Koutarou Suzuki. "Traceable ring signature." Public Key Cryptography–PKC 2007. Springer Berlin Heidelberg, 2007. 181-200.

[GM] Maxwell, Gregory. "Confidential Transactions." https://people.xiph.org/~greg/confidential_values.txt

[LWW] Liu, Joseph K., Victor K. Wei, and Duncan S. Wong. "Linkable spontaneous anonymous group signature for ad hoc groups." Information Security and Privacy. Springer Berlin Heidelberg, 2004.

[SN] Noether, Shen. MiniNero, (2015), GitHub repository, https://github.com/ShenNoether/MiniNero