RESEARCH BULLETIN MRL-0008

Dual linkable ring signatures

Sarang Noether* and Brandon Goodell†
Monero Research Lab
October 24, 2018

Abstract

This bulletin describes a modification to Monero's linkable ring signature scheme that permits dual-key outputs as ring members. Key images are tied to both output one-time public keys in a dual, preventing both keys in that transaction from being spent separately. This method has applications to non-interactive refund transactions. We discuss the security implications of the scheme.

1 Introduction

The original CryptoNote protocol describes the use of a type of one-time linkable spontaneous anonymous group (LSAG) signature. A signer chooses a so-called ring of public output keys, one of which is her own, and fashions a signature on the message of her choice. The scheme is such that any verifier can be assured that one of the keys in the ring is the true signer (that is, the signer knows the corresponding private key), and that this key was not used to sign any other message with any other ring. For space efficiency, early Monero transactions used a ring-independent one-time version of the LSAG signature scheme in [2] to direct funds using the ring as a sender anonymity set.

To handle confidential transactions, Monero uses a variation of this scheme called multilayered linkable spontaneous anonymous group (MLSAG) signatures. These signatures allow the signer to include vectors of keys that include Pedersen commitments to amounts, as described in [4].

In both LSAG and MLSAG signatures, ring members (with the exception of amount commitments) are output public keys, which are generated in Monero transactions. A given transaction typically has multiple outputs, where the sender directs a portion of funds from a previous output to some address, and sends the change back to herself so the transaction balances. In each case, the recipient can recover the output's private key and use it in a later ring signature.

In this research bulletin, we describe a modification to the construction of transaction outputs and to the construction of ring signatures. We first describe a change whereby a sender generates a dual-key output and a specified trigger block height that "switches" the validity between the two. Further, we describe a modification to LSAG and MLSAG signatures, a dual linkable spontaneous anonymous group (DLSAG) signature scheme, that allows a sender to include one of these dual-key outputs in a linkable ring signature. When a dual, which is comprised of two separate output public keys, is the true spender, any verifier can link two ring signatures if they were computed by either of the keys in the dual. Additionally, it is possible to include both dual-key outputs and non-dual (that is, single) outputs as ring members in a DLSAG signature, ensuring that signers have the largest possible set of potential ring members available to them.

This signature scheme has applications to refund transactions in Monero, which themselves are important for certain second-layer solutions. To generate such a refund transaction, a sender generates a dual-key output

*sarang.noether@protonmail.com
†surae.noether@protonmail.com

and specifies a trigger block height. The consensus protocol could dictate that prior to the trigger height, only one of the keys in the dual is valid as a spender; after the trigger height, only the other key in the dual is valid. This allows the sender to reclaim the funds if they are not spent by the recipient in time.

A related version of this scheme was originally described in personal communication with Pedro Moreno-Sanchez, and was in collaboration with pseudonymous coauthor `donut`. This related scheme considered the use of commitments to block trigger heights in order to hide the actual switching height.

2 Description

Let $\mathbb{G}$ be an additive group of prime order ℓ. Let $H_s : \{0,1\}^* \to \mathbb{Z}_\ell$ and $H_p : \{0,1\}^* \to \mathbb{G}$ be cryptographic hash functions. Let $G \in \mathbb{G}$ be a publicly-known group generator.

When sending funds in a Monero transaction, the sender uses the recipient's public user address $(A, B) := (aG, bG) \in \mathbb{G} \times \mathbb{G}$, along with a random nonce r, to generate a one-time output public key defined as $P := H_s(rA, t)G + B$, where t is the output's index within the transaction. The recipient uses her private user key (a, b) and the point $R := rG$ to recover the output private key $p := H_s(aR, t) + b$, which is used to spend the funds as part of a later ring signature.

To facilitate refund transactions, we assume that a sender has generated a modified output consisting of two one-time public keys. One such key is directed to the recipient's address, while the other is typically directed back to the sender. These one-time keys are called a *dual*, and are designated as such in the transaction structure for later identification. The dual also has an associated trigger block height value, with the intent that prior to the trigger, only the first one-time key is valid and can be used to spend the associated funds, and after the trigger, only the second is valid. This has the effect of permitting the recipient to claim funds up to a certain time, after which the sender can reclaim them.

Despite the simplicity of generating separate outputs representing the spend authority of the same funds based on a given block height, it does not suffice to use a traditional LSAG or MLSAG ring signature that includes one of the public keys in a dual without modifying the key image computation. Because key images are computed for each output spent in a ring signature, it would be possible for both the pre-trigger and post-trigger keys to be used in separate signatures undetected. To combat this, we modify the LSAG scheme such that one-time output public keys come in pairs that share the same key image. We later discuss the MLSAG equivalent to the construction.

2.1 Signature generation

We show the steps of generating the DLSAG signature in Table 1, allowing for any of the ring members (including the real spender) to be either part of a dual or not. In the following notation, we assume the spender wishes to spend funds associated to a one-time key $P_\pi = p_\pi G$ with a ring size n, where $1 \leq \pi \leq n$ is a secret index. We also assume that any key P_i that is part of a dual has a partner key Q_i. The roles of P_i and Q_i in a dual are arbitrary.

Note that a DLSAG signature fully reduces to the LSAG case when all keys in the ring, including the true spender, are not part of a dual; that is, when the signature follows only the right-hand side of the above diagram.

The sender is free to sign with either type of key. However, she must consider a restriction on the allowed ring members. If a potential ring member is part of a dual, she must examine the original transaction and determine whether that key is valid at the current block height. If it is not, then she must choose the partner key within the dual. She may wish to further avoid keys whose trigger is very close to the current block

Dual-key		Single-key
$m_\pi \equiv H_s(\mathsf{txid}, \mathsf{index})$		
$J \equiv m_\pi p_\pi Q_\pi$		$J \equiv p_\pi H_p(P_\pi)$
$F_\pi \equiv m_\pi Q_\pi$		$F_\pi \equiv H_p(P_\pi)$
	choose random u	
	$c_{\pi+1} \equiv H_s(\mathsf{txdata}, uG, uF_\pi)$	
	for each $i \neq \pi$	
	choose random s_i	
$m_i \equiv H_s(\mathsf{txid}, \mathsf{index})$		
$F_i \equiv m_i Q_i$		$F_i \equiv H_p(P_i)$
	$c_{i+1} \equiv H_s(\mathsf{txdata}, s_i G + c_i P_i, s_i F_i + c_i J)$	
	$s_\pi \equiv u - c_\pi p_\pi$	
	output $(c_1, \{s_i\}_{i=1}^n, J)$	

Table 1: DLSAG signature generation

Dual-key		Single-key
	for each $1 \leq i < n$	
$m_i \equiv H_s(\mathsf{txid}, \mathsf{index})$		
$F_i \equiv m_i Q_i$		$F_i \equiv H_p(P_i)$
	$c_{i+1} \equiv H_s(\mathsf{txdata}, s_i G + c_i P_i, s_i F_i + c_i J)$	
$m_n \equiv H_s(\mathsf{txid}, \mathsf{index})$		
$F_n \equiv m_n Q_n$		$F_n \equiv H_p(P_n)$
	$c_1^* \equiv H_s(\mathsf{txdata}, s_n G + c_n P_n, s_n F_n + c_n J)$	
	accept only if $c_1^* = c_1$	

Table 2: DLSAG signature verification

height in case her transaction is not included in a new block quickly enough. These restrictions ensure that adversarial verifiers of the signature cannot easily eliminate invalid ring members when trying to determine the true spender.

Also observe that partner keys P and Q in a dual share the same key image when generated in the same transaction, since $mpQ = mqP = mpqG$ by construction. The inclusion of the hash m, which encodes both the originating transaction as well as the index of the output within that transaction, prevents one of the two recipients of a dual from burning the funds of the other by generating another dual with the same key image that the evil recipient spends first.

2.2 Signature verification

Verification of a DLSAG proceeds similarly to that of an LSAG signature, and may be done by any observer. When presented with a list of output public keys that are used in the ring signature, the verifier first ensures that for any keys that are part of a dual, the chosen key is valid at the transaction's block height. If not, the verifier rejects the signature. The verifier also examines the key image J; if it appears as part of any previous valid LSAG or DLSAG signature, the signature is rejected. The verifier next completes the steps in Table 2.

3 Application to refund transactions

A useful application of the DLSAG scheme is to refund transactions. Some constructions of payment channels, which permit off-chain transactions between two parties that are later settled, require the use of non-interactive refund transactions as well. Suppose that Alice wishes to send funds to Bob, but wants to ensure that they are returned to her after an agreed-upon amount of time if Bob does not spend them. To do so, Alice constructs a transaction where the funds destined for Bob are part of a dual: one key P is directed to Bob, and the partner key Q is directed to her. The two keys in the dual share a range proof. A block height h is included in the transaction data, where h is greater than the current block height of the network.

If Bob wishes to claim the funds, he must spend them prior to block height h in a transaction that uses the key P in a DLSAG signature. Verifiers see that the transaction includes P among its ring members, and that this output is still valid. The key image is of the form $J = mpQ$, which has not been used before. The verifiers accept the transaction as valid.

However, if Bob does not claim the funds prior to block height h, Alice may claim them by spending her key Q using a DLSAG signature. Verifiers now see that the transaction includes Q, and that this key is now valid. The key image is $J = mqP$, which has not been used since Bob did not spend P. The verifiers accept this transaction as valid.

Notice that if Bob spent P prior to height h but Alice becomes evil and wishes to spend Q after height h (which would constitute a double spend), the process will properly fail. Verifiers will find Alice's key image $J = mqP = mpQ$ to be the same as Bob's, and will reject her transaction. This construction of a shared key image is essential, as otherwise verifiers would not reject Alice's evil transaction. On the other hand, if the chain on which Bob signed the transaction is later overtaken by a chain with greater cumulative difficulty that does not include Bob's transaction, Alice can always claim the funds later; we consider this possibility outside the scope of this work.

4 Multi-input extension

The DLSAG construction presented above applies only to the now defunct CryptoNote ring signature implementation. However, modern transactions require a more robust multi-key signature that can accommodate the amount commitments used in the Monero confidential transaction model. We therefore must ensure that there is an appropriate generalization of the DLSAG construction.

In a general MSLAG signature, each of the k inputs has an associated ring containing n public output keys (in addition to a separate amount commitment that we do not consider here). The sender chooses a secret index $1 \leq \pi \leq n$ such that she controls the public keys $P_{\pi,j} \equiv p_{\pi,j}G$ for $1 \leq j \leq k$. Signature generation proceeds according to the steps in Table 3.

As before, this reduces completely to an MLSAG signature if no ring members are part of a dual. Further, it reduces entirely to the single-input DLSAG construction shown above in the case $k = 1$. Given a signature and description of the corresponding set of public one-time output keys, signature verification follows the steps in Table 4.

5 Security

The similarity between DLSAG signatures and the LSAG (from [2]) and MLSAG (from [4]) schemes leads to similar proofs of security. As in the cited cases, we want to show that our signatures are unforgeable,

Dual-key	Single-key
for each $1 \le j \le k$	
$m_{\pi,j} \equiv H_s(\mathsf{txid}, \mathsf{index})$	
$J_j \equiv m_{\pi,j} p_{\pi,j} Q_{\pi,j}$	$J \equiv p_{\pi,j} H_p(P_{\pi,j})$
$F_{\pi,j} \equiv m_{\pi,j} Q_{\pi,j}$	$F_{\pi,j} \equiv H_p(P_{\pi,j})$
choose random u_j	
$c_{\pi+1} \equiv H_s(\mathsf{txdata}, \{u_j G, u_j F_{\pi,j}\}_{j=1}^{k})$	
for each $i \neq \pi, 1 \le j \le k$	
choose random $s_{i,j}$	
$m_{i,j} \equiv H_s(\mathsf{txid}, \mathsf{index})$	
$F_{i,j} \equiv m_{i,j} Q_{i,j}$	$F_{i,j} \equiv H_p(P_{i,j})$
$c_{i+1} \equiv H_s(\mathsf{txdata}, \{s_{i,j} G + c_i P_{i,j}, s_{i,j} F_{i,j} + c_i J_j\}_{j=1}^{k})$	
for each $1 \le j \le k$	
$s_{\pi,j} \equiv u_j - c_\pi p_{\pi,j}$	
output $(c_1, \{s_{i,j}\}_{i,j=1}^{n,k}, \{J_j\}_{j=1}^{k})$	

Table 3: Generalized DLSAG signature generation

Dual-key	Single-key
for each $1 \le i < n, 1 \le j \le k$	
$m_{i,j} \equiv H_s(\mathsf{txid}, \mathsf{index})$	
$F_{i,j} \equiv m_{i,j} Q_{i,j}$	$F_{i,j} \equiv H_p(P_{i,j})$
$c_{i+i} \equiv H_s(\mathsf{txdata}, \{s_{i,j} G + c_i P_{i,j}, s_{i,j} F_{i,j} + c_i J_j\}_{j=1}^{k})$	
for each $1 \le j \le k$	
$m_{n,j} \equiv H_s(\mathsf{txid}, \mathsf{index})$	
$F_{n,j} \equiv m_{n,j} Q_{n,j}$	$F_{n,j} \equiv H_p(P_{n,j})$
$c_1^* \equiv H_s(\mathsf{txdata}, \{s_{n,j} G + c_n P_{n,j}, s_{n,j} F_{n,j} + c_n J_j\}_{j=1}^{k})$	
accept only if $c_1^* = c_1$	

Table 4: Generalized DLSAG signature verification

linkable, and signer-ambiguous. As the proofs are nearly identical to those presented for the original signature schemes, we highlight only the notable differences.

5.1 Unforgeability

The LSAG/MLSAG proofs of unforgeability use the number of calls to random oracles to establish bounds on a defined adversary's advantage in breaking the discrete logarithm problem in recovering a signer's private key. These oracles represent the behavior of the hash-to-scalar function H_s, hash-to-point function H_p, and valid signing. We observe that for a dual key image of the form $J = mpqG$, the value m is uniformly distributed under the random oracle model. For a uniformly distributed value xG, the distributions of (pG, qG, J) and (pG, qG, xG) are computationally indistinguishable under the decisional Diffie-Hellman assumption.

The proof of unforgeability in [4] therefore holds with only minor modification, where we consider calls to the hash-to-point random oracle for dual keys replaced by calls to a decisional Diffie-Hellman oracle.

5.2 Linkability

If an adversary is able to create two signatures signed by key vectors sharing a common true signing key, there are two cases. If the common signing key is not part of a dual, the proof continues identically to [4]. If the common signing key is part of a dual, then we observe that

$$\log_G(s_iG + c_iP_i) = \log_{m_iQ_i}(s_iF_i + c_iJ)$$

where i is the index of the common signing key in one of the signatures; this leads to the same conclusion as in [4].

5.3 Signer ambiguity

We observe that the ambiguity proof in [4] does not rely on the particular structure of the base point used in terms passed into the hash-to-scalar function H_s to generate commitments. The proof is trivially modified to account for the F_i terms presented here.

5.4 Heuristic attacks

The DLSAG scheme provides a guarantee that, absent external information, any public key referenced in a signature is equiprobable as the true signer. However, an adversary may use such external information to undermine this guarantee via heuristics.

- *Spend time.* If Alice initiates a refund transaction to Bob and Bob does not spend the funds before the trigger height, Alice may attempt to spend the funds shortly after this occurs. If an adversary sees a ring containing a dual whose trigger was recently reached (or perhaps is about to be reached), the adversary may conclude that the dual is the true spender. Moreover, one of the keys in a dual is valid for only a small amount of time, while its partner is available in perpetuity. Such heuristics should be considered in relation to the suspected spend patterns discussed elsewhere, as in [3, 1].
- *Availability of dual outputs.* Dual outputs are trivially distinguishable from single outputs. If the number of available dual outputs in the blockchain is small relative to single outputs, they will be chosen less frequently as ring members. An adversary may conclude that any ring containing dual outputs is more likely to have such an output as the true spender.

5.5 Key reuse

The current Monero transaction implementation mitigates against both double-spending and one-time key reuse. If a user's wallet sees multiple outputs paying to the same one-time key, it chooses the one with the largest amount; otherwise, it risks losing a larger amount since all such outputs have the same key image. This mitigation is not possible with the modified key image computation considered with dual-key outputs, which only prevents double-spend attacks. Alternate protocol-level rules involving one-time key reuse, or the inclusion of a second key image and more complex signature, may be required to mitigate.

6 Concluding remarks

The signature scheme presented here offers an interesting and novel approach to non-interactive refund transactions for use in Monero. However, implementation choices like block height commitments and requirements for outputs would have an effect on transaction complexity, size, and adversarial heuristics. It is not known whether a space-efficient key generation method could be used to describe dual-key outputs in a more efficient way. The cost in verification complexity is likely to be unavailable without novel signature schemes.

References

[1] Amrit Kumar, Clément Fischer, Shruti Tople, and Prateek Saxena. A Traceability Analysis of Monero's Blockchain. Cryptology ePrint Archive, Report 2017/338, 2017. `https://eprint.iacr.org/2017/338`.

[2] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *Information Security and Privacy*, pages 325–335, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[3] M. Möser, K. Soska, E. Heilman, K. Lee, H. Heffan, S. Srivastava, K. Hogan, J. Hennessey, A. Miller, A. Narayanan, and N. Christin. An Empirical Analysis of Traceability in the Monero Blockchain. *ArXiv e-prints*, April 2017.

[4] Shen Noether, Adam Mackenzie, and the Monero Research Lab. Ring confidential transactions. *Ledger*, 1(0):1–18, 2016.