LEDGER
ledgerjournal.org

ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

# Ring Confidential Transactions: Open Review

Authors: Shen Noether,† Adam Mackenzie, the Monero Research Lab*

Reviewers: Reviewer A, Reviewer C

**Abstract.** The final version of the paper "Ring Confidential Transactions" can be found in Ledger Vol. 1 (2016) 1-18, DOI 10.5915/LEDGER.2016.34. There were two reviewers who responded, neither of whom have requested to waive their anonymity at present, and are thus listed as A and C. After initial review (1), the authors submitted a revised submission. This was reviewed a second time by reviewer C (2A). The assigned Ledger editor then recapitulated key points of the reviews, which the authors responded to point-by-point (2B). After this the assigned Ledger editor determined that the authors had adequately addressed the reviewer concerns and asked the authors for minor revisions which were carried out by the authors, completing the peer-review process. Authors' responses in 2B are in bullet form.

## 1. Review, Initial Round

**Reviewer A:**

A very nice paper, with interesting new ideas and sound security proofs.

Cryptography is ok and the extension to greater anonimity is indeed helpful.

I suggest acceptance, especially if Section 5 is removed (which doesn't fit so well with the remainder of the paper) and the "I" is substituted with "we".

**Reviewer C:**

This paper presents a method to hide transaction amounts in Monero i.e. allow for "confidential transactions". In order to achieve such a property the authors propose a new ring signature scheme that is linkable which is essential in order to achieve double-spending security. At the same time, the natural properties of ring signatures provide the necessary anonymity required. The new signature scheme is "multilayered" which essentially means that it provides two layers of anonymity: one that hides who is the user who signed the signature

† S. Noether (shen.noether@gmx.com) is a cryptography researcher at the Monero Research Lab
*1KTexdemPdxSBcG55heUuTjDRYqbC5ZL8H

and one that hides the payment denomination used. The proposed scheme called MLSAG is very close to an older scheme by LWW'04 but it allows the ring to consist of a vector of vectors of keys. Although I think that MLSAG is useful (although not particularly novel - it is a pretty straightforward modification of LWW'04) and it does enhance the anonymity of Monero, I think that the current shape of the submitted manuscript needs major changes. I believe that the changes needed are both in presentation level (explaining what is the goal of the paper, have clear contributions and comparisons etc) as well as in definitional/formality level.
That said, my suggestion is to revise and resubmit the paper for review.
Below I give detailed comments and suggestions.

Specific comments
The literature on mixing services should be updated since not all mixing services require a trusted party: in schemes like XIM for example the third party might be malicious so the anonymity of the users is not affected in case of a compromise. Thus, maybe a better argument when comparing with such schemes would be in terms of efficiency (time it takes to complete a transaction).

It would be useful to add a short explanation of how Monero works and especially how it achieves to implement hidden amounts in the transactions. This would set the ground for the reader to better understand the difference between original Monero and the modification suggested in the current paper (thus would be nice to be included in the introduction).

Page 6, end of section 1.2. Is there a reference for the protocol presented by C. Frenkencht?

I feel that by reading the current draft one might be confused between ring and group signatures. Monero actually needs a type of linkable *ring* signatures (since anonymity is important). In the LWW04 paper this is also referred as a "linkable spontaneously anonymous group (LSAG)signature". I think that some clarification here would be helpful.

Section 2.1 title is misleading. What the authors claim to described here is a modification of FS07 (traceable ring signature) given by Adam Back in order to reduce storage cost (using a trick used in LWW04 that allows any 3 party to reconstruct on the fly parts of the signature instead of receiving them as part of the signature). What is not clear here is what's the goal of this modification: if the goal is to get a linkable ring signature then LWW04 is already a linkable ring signature. Actually I think that the scheme in Bac15 is essentially the same as LWW04 - what is the difference, if any?

Moreover, in FS07 it is observed that LWW04 is vulnerable against an attack by a dishonest signer that makes a honest signer accused of "double-spending". A note of why this is not relevant here would be useful.
Also a definition (even informal) of a linkable ring signature (i.e. what algorithms/protocols does it consist of and what are the security properties) would be helpful.
Page 8, In keygen algorithm: define index j - is it in {1,..,n}? Add reference for ed25519 (and maybe add a sentence explaining what it is). State size of keys. Hash function returns a point in where? Since the description is in regards a signature scheme maybe call x the "signer's


ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

secret/signing key"? Denote $I$ as the "key image" since this notation is used later. Also, is $I$ public? It would be helpful to state public and secret keys of users at the end.
Sentence "let me be a given message" is irrelevant to Keygen - it can be moved to Sign.
Page 8-9, Sign algorithm. What's the message space? (is it messages of arbitrary size as in LSAG?) Are s_j values really randomly selected as stated in the beginning? My understanding is that every s_i is defined by \alpha, c_j and x.
End of page 9: should also be given message m.
Page 10: Not sure why Link needs to reject a signature (this seems to be needed later but seems irrelevant in the description of a signature scheme). Link algorithm should just decide if two signatures given as input were generated by the same user or not.

End of Section 2.1. It would be useful to highlight the differences between the described signature and LWW04. This would make it more clear to see whether the proofs of LWW04 still hold.

Section 2.2. Some motivation on why vectors of keys are needed would be helpful. Also, state again then format of secret/public keys. Is it same as in 2.1? Finally, is the same hash function H used for both the computation of R's and c's?
Section 2.3 Defining MLSAG (security model). This section will have to be remodeled and should be moved before the construction. As mentioned above the standard way to describe a new scheme is: (1) define scheme and security model, (2) construct and (3) prove construction secure according to security model of (1). This section should start by defining the algorithms/protocols of an MLSAG (i.e. as it is done in LWW04 beginning of Section 3).

Section 2.3 I found the unforgeability definition kind of odd. For instance how can A know/select the public keys without knowing the secret keys? In fact think that the unforgeability definition is misleading even in LWW04. Authors should look at BKM'05 (`https://eprint.iacr.org/2005/304.pdf`)
and AOS'02 (`http://link.springer.com/chapter/10.1007%2F3-540-36178-2_26`)
for a formal version of unforgeability.
Definitions of Linkability and Signer Ambiguity could be further formalized as in LWW04.

In proof of unforgeability in Appendix A.1 notation is inconsistent to the construction: i.e. hash functions are H_1 and H_2 when just H is used in construction (which as I mention above I think is wrong). Calligraphic L never defined. Also should be stated (in definition of Thm 11) that proof is given in the random oracle model. Finally, in the proof it seems that the PPT adversary M is given as input a list L of public keys (by a challenger) and tries to find the discrete log of one of them. Thus, the list L should be forward as input to A. However, this does not much the unforgeability definition given in main body (please also see note above - I believe this definition needs to be fixed).

Lemma 10. What is the cited Theorem out of which proof easily follows? I actually do not think that proof of Lemma 10 is trivial. However, it is indeed given in LWW04 - so a citation to them is probably enough.

Page 14: what does SUM of Inputs/Outputs notation stands for? Is it the total sum of bitcoin value?


ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

Page 17 Definition 6: this is a construction not a definition.

Section 4: I was expecting a more clear description of how n and m (from the key vector of the new signature scheme) are mapped to monero. Is $n$ the number of users in the system and $m$ the number of possible denominations?

Regarding Section 5:
First I do not understand why this section is called “Range Proofs”. I essentially see a new signature being proposed here called aggregate shnorr non-linkable ring signature. First it would be very useful to define what an aggregate non-linkable ring signature is - such a primitive does not exist in the literature so relevant definitions and security model does not exist. After the construction a proof of unforgeability is given (Also in proof of Thm 9, analysis of prob. of success is missing.). However, this is certainly not enough. Unlinkability should also be proven and potentially other properties depending how one defines “aggregate non-linkable ring signatures”.

Moreover, regarding Section 5 I do not understand the advantages or the trade offs by using this method instead of the protocol of Section 4. I overall think that that some sort of comparison of original CT and the 2 new methods presented in the paper is needed in order to understand the contributions of this work.

Typos/Minors
* Page 5, 2nd line of Sec 1.2. Typo in CryptoNote name
* Page 6, 2nd line of Sec 1.3 “Ring CT” notation have been used before. Maybe denote when first refer to “Ring Confidential Transactions” above.
* Page 6, 5th line of Sec 1.3, delete “in”
* Page 8, in 3rd line of Keygen, should be “signer’s”. Also, next sentence there is a period missing after footnote.
* Pages 8-10, be consistent on capitalizing (or not) names of algorithms/protocols
* Page 13, 2nd to last line, should be “to the Confidential..”
* Inconsistency between “proof of work” and “proof-of-work” (in various places) same with “multisignature” and “multi-signature”.
* I found the use of first-person singular throughout the paper a bit odd (especially given that there are 2 authors).

## 2A. Review, Second Round

Sec 1.1, last paragraph: 2nd line needs editing - doesn’t make much sense towards the end.

Nice that XIM reference is added. However, I think this section could be further updated to better capture recent work. Figure 1 from BOLL14 or Table 1 from `https://eprint.iacr.org/2016/575.pdf` would help.



ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

Section 1.4, maybe add a note that if trusted setup compromised then anonymity is still guaranteed (although coins can be now forged).

Section 1.5, minor editorial comment: say in the first couple of lines what's the advantage of new proposal over Monero (i.e. what does the modification offer?)

Page 5, 2nd to last par, line 7 needs editing

What is a "back signature", only mentioned in Title of Sec. 2.1.1

Sec 2.1 could maybe moved to a prelims section and keep Section 2 just defining the new primitive?

Sec 2.2. In SIGN algorithm definition - what is G? Keep definitions clear of construction specifics. Do you need the public key vector that corresponds to the secret key vector x given as input? I am confused with the input of SIGN

In VER algorithm, the key matrix sigma should be public keys, right?

LINK algorithm is undefined. What does it mean for two signatures to be linked? Under the same sk, under the same public keys vector - what?

Actually for the whole definitional part I am not sure you need to define the keys as vectors - you could just define them as single values and thus make it less confusing. The vectors are only needed as part of the construction as far as I can tell. An example of where they make things very confusing is the unforgeability description - is it problem if the adversary knows only a subset of a sk vector for the corresponding pk?

Unforgeability is ill-defined. Given that the adversary is given the signing oracle he can obviously output valid signatures. The when does he win? There multiple scenarios here given the type of unforgeability you aim for. I.e. in existential unforgeability Adversary only wins if outputs (m, sigma) for m *not* queried to SO. Then, there is also the notion of strong unforgeability, where A is succesful is ouputs another sigma for a message m that he sent as query to oracle SO.

Also in unforgeability does oracle SO outputs valid signatures for any of the public keys in list Sigma?

Def 3 - Q should be defined here.

Def 4 - does the set D_t consist of vectors (see also above for vectors/non-vectors on the definition)?

Overall regarding definitions the only way to be very specific with such properties is to write them more formal as security games - it would be much easier to understand and properly prove them.



ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

Page 10 please edit text in 1st line right below bullets.

You should state in introduction that the new ring signature scheme is claimed secure in the random oracle model.

Please include a better citation for range proofs (they did not originate at Max15 - although used there)

Maybe consider a different title for section 3 - when you say "background..." it sounds like you just describe pre-existing work but you do suggest new stuff here

Finally, as a general note. I do appreciate the authors effort to provide definitions and proofs for the new type of ring signature they introduce. However, no definitions and proofs are given for the overall scheme (Ring CT as a currency). This is a long and challenging process and even for Zcash is still under development (in FC'16 the authors of Zerocash did a first attempt for a full simulation based definition and proof).

The absence of such a proof makes the statements in sections 1.4 and 1.5 sound very strong. I am not suggesting that this work is needed as part of this paper but I think it is important to highlight that the overall security should still be studied as a future work.

## 2B. Authors' Responses

Dear Shen and Adam,

Thanks for sending a revised version, you've improved the paper substantially and addressed most of the reviewers' suggestions.

After looking at the diff between the original and recent revisions, there are still several suggestions that I believe were left unaddressed.

For each of these, if you think you did addressed them and I just missed it, or you've deliberately chosen not to address these, could you explain why?

- Include a citation to `https://dl.acm.org/citation.cfm?id=2665955`, at least, to improve the background on mixing alternatives

- I cited this in section 1.2 (BOLL14), based on previous reviewers suggestions.

- Could you provide some clarification on the terminology used, especially between "ad hoc group", "spontaneous anonymous group", and "ring" signatures? At one point you use the phrase "ad hoc ring signature", which might be redundant if "ring" means "ad hoc group".

- This was clarified on the footnote on page 6, which is at the start of section 2, where I talk about the . If this is not an obvious place, let me know and I can move it somewhere else.



ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

- Section 2.1 title is misleading. Given that LWW04 is already a linkable ring signature, then what exactly is the difference between Bac15 and LWW04? You have referred to the differences compared to LWW04 as “insignificant”, and “similar, but more efficient”, but could you identify the exact difference more clearly? This will make it more straightforward for reviewers to confirm that the proof still holds.

- This was clarified at page 8, after the link phase: “The very slight difference between the above signature of [Bac15] and that of [LWW04] is in the key image. In the scheme described in [LWW04], I is defined by the equation I = x j H p (P 1 ,P 2 ,...,P n ). In other words, that scheme proves that a signer can only sign once with respect to a given set of keys (although they could sign with the same key in many different sets of keys, each having different hashes). In the above scheme of [Bac15], the key image is I = x j H p (P j ) which enforces that a signer can only sign with a key once, without their sig- nature being rejected in the LINK phase of the algorithm. This modification, which is similar, but more efficient to the one made in [FS07, Section 3], is more suited for digital currency use, since it prevents double-spending of the value stored in a given key. Note that proofs of unforgeability, anonymity, and linkability hold for the above protocol which are only insignificant modi- fications to the proofs given in [LWW04]. We will give a more general version of these proofs for the MLSAG’”
  Note-> In the version I am sending you now, I have made this a subsubsection (2.1.1) called “Back signatures vs LWW signatures.”

- It seems like the generalization that makes this article a contribution vs LWW04 and Bac15 is that n keys are generalized to n key-vectors. What is the significance of a key-vector? You've defined it, and it shows up briefly in the introduction (as "To deal with this issue.... [we allow] a user to sign with a vector of keys, rather than a single key"), but I don't think this is clear enough.

- This version I am sending clarifies: To deal with this issue, a generalization of the signatures of [LWW04] is created, which allows a user to sign with a vector of keys, rather than a single key. The point of using a key-vector is so that a user can prove they have simultaneously have knowledge of both the secret key of the input address and the secret key of the input commitment. (Contrast this with signing two distinct ring signatures: one for the commitment, and one for the input key. This would prove that you know keys for an amount and an address, but not necessarily link the two). This generalization is...

I think the root cause of this confusion is due to the fact that in Confidential Transactions, "ring signatures" are sort of repurposed/overloaded to actually serve as a zero-knowledge range proof. In conventional use, each public key in a ring is an ordinary public key, typically each generated by a different user or at a different time --- the public keys are entirely independent and unrelated. But here, each "public key" in a row is actually a sort of ephemeral



ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

commitment to a possible value, and in fact they're all related and derived from by the same user, and by "signing" one of them signer proves that the committed value is in a range.

My suggestion is to clear this up with a more explicit explanation in Section 3.3 where you describe how ring signatures are used as range proofs in CT. And then refer to this section and give a shorter description when mentioning "key-vectors" and why the generalization is needed.

- A good point, since this is possibly confusing. I have included in beginning of 3.3: "Note that this range proof construction also uses a type of ring-signature, which does not need linkability. This ring signature is used for a completely different purpose than the MLSAG's (namely the range proof ring signatures are used to prove each bit of the output amount is in the set {0,1} without giving away which number it actually is) and have no interaction with the MLSAG part of the Ring CT Protocol."

- In FS07 it is observed that LWW04 is vulnerable against an attack by a dishonest signer that makes an honest signer accused of "double-spending". A note of why this is not relevant here would be useful.

- This is covered in 2.1.1 where I discuss the difference between LWW and Back signatures (the Back signature applies the fix of the FS07 to LWW).

- Typo: MSLAG at the beginning of section 2.2 should be MLSAG

- Fixed

For messages, you do now define the message space (basically, sha256 hashes of arbitrary messages), but that's in a weird place, in the Linkability property. Why not move it up to where message m first appears?

- I've updated with a sentence to this effect on page 7, where message first appears.

Also you switched between two different fonts, like ordinary math $m$, and $\cal{m}$.

- (I've fixed the two different fonts to be using mathfrak only)

Also in Definition 3, you use the prime versions (e.g., m', and \sigma') without introducing them.

- I've update this by introducing them.

Also the equation in this definition is repeated twice

- Intentional, it's "if it's linked then you can tell, if it's not linked, then you can tell"


ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34

- For notation on the hash functions, it looks like in the scheme you now use lowercase h to denote a sha2-like hash that returns a string, and H_p to denote a hash that randomly samples group elements. In the proof, you still use H1 and H2.

- You're right, this is annoying – I've updated this everywhere so that H_s is hash to scalar, and H_p is hash to point

In section 4.4 you use H(), where I think you meant H_p.

- Fixed.

- You improved the definition of Unforgeability. Instead of A choosing public keys, it chooses from among a set of public keys in some public set, e.g. on the blockchain. This is still imprecise. Can A include its own bogus public keys to this set? Look at Section 3.2 `https://eprint.iacr.org/2005/304.pdf` for a discussion of subtleties in this definition.

- "A" is allowed to include bogus public keys in the set as long as they don't know an entire private key-vector (they can even know all but one of the secret keys in a key-vector) - this is the last sentence in my definition- the logic here is that, since they must solve, at the end of the signature s = a – cx, then as "c" is a hash, which they cannot control, if they don't know x they can't find s. Note that LWW have a similar definition to the one I am making "List of public keys chosen by A", and their paper is cited in your link ( [20], with respect to the final definition in 3.2). Specifically, the wording "don't know an entire secret key-vector" corresponds to R\subseteq S\C, where C is the set of corrupted users, in definition 7, section 3.2. – I can probably change this wording if desired.





ISSN 2379-5980 (online)
associated article DOI
10.5915/LEDGER.2016.34