Verifiable Mixing Protocol for Bitcoin

Morgan Locks

December 10, 2015

Abstract. I outline a protocol to allow mixing services in blockchain-based currencies such as Bitcoin to prove that they have paid every user of the service. This is done without revealing compromising information that would permit linking of users' inputs and outputs in a way that normal mixing is not vulnerable to. I also show an extension of this protocol to allow users to specify multiple output addresses.

This protocol relies on an efficient zero-knowledge set-membership proof (described in [1]) that drastically reduces the number of signatures that must be transmitted when multiple payments are being verified simultaneously.

1 Introduction

In the Bitcoin protocol, some amount of bitcoin is associated with a particular address, and can only be spent by users with access to the private key for that address (the "owner"). Addresses are not intrinsically linkable to a real individual or group, but may become linked due to identifying operations such as converting physical currency into bitcoins, or making transactions based on physical services.

Mixing services allow users to trade in coins that may be linkable to them in exchange for coins which are unlikely to be linkable to them. Due to the anonymous nature of this process, it is not possible for users to show that an illegitimate mixing service has failed to return money to the user, or for a mixer to show that users who falsely claim that their money was stolen are lying. As a result, trust can only be built slowly and unreliably. Furthermore, trust can only be destroyed slowly and unreliably, if the mixer begins to behave illegitimately.

In order for a mixer to quickly build trust, it would be useful for it to be able to prove to any third-party that it has paid any given user the amount which the user is due, without revealing the address to which the user requested their money be sent. That is the purpose of the Verifiable Mixing Protocol (VMP).

This protocol uses a set-membership proof technique created by Camenisch, Chaabouni, and Shelat [1]. By storing a commitment to an output address in the blockchain (the publicly visible and immutable record of transactions) it is possible to prove in zero knowledge that the address to which the user committed was later paid by the mixer.

2 Definitions

Set Membership Proofs and Zero-Knowledge Proofs. This paper uses definitions from [1] and [2]. In particular, a proof of set membership is defined as a proof of the following:

$$PK\{(\sigma, \rho) : x \leftarrow Com(\sigma, \rho) \wedge \sigma \in \Phi\}$$

where σ is the element which was committed to (which only the prover P knows), ρ is the commitment which both the verifier V as well as P knows, and Φ is the set of elements in question.

A *zero-knowledge set-membership proof* is any instantiation of the above proof system such that, after proof has been made, if V is an honest verifier, V gains minimal information about σ.

It is possible to convert an honest-verifier zero-knowledge proof into a general zero-knowledge proof [3]. Therefore, this paper will only discuss the simpler case of an honest verifier.

Bilinear Groups. VMP uses a bilinear group scheme for signature verification. In particular:

- G_1, G_2, and G_T are cyclic groups of prime order p.
- g_1 and g_2 are generators for G_1 and G_2 respectively
- e is a bilinear pairing $G_1 \times G_2 \rightarrow G_T$ such that the following hold:
  $\forall u \in G_1, \forall v \in G_2, \forall a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$
  $e(g_1, g_2) \neq 1$ i.e. g_1 and g_2 map to a generator of G_T

Commitments. The VMP protocol uses Pedersen commitments of the form $Com(\sigma) = g^\sigma h^r$, where g and h are group generators for G_1 in the bilinear mapping scheme described above.

Boneh-Boyen Signatures. Bilinear mappings are used in VMP to construct Boneh-Boyen signatures as described in [1] and [2]. In this case, $p = |G_1| = |G_2| = |G_T|$. The signer has a private key $x \in \mathbb{Z}_p$ and publishes some generator $g \in G_1$ such that $g \neq 1$. The signer also publishes a public key g^x. For a message m, $sig(m) = g^{1/x+m}$. The signature can be verified by checking that $e(\sigma, g^x g^m) = e(g, g)$. This scheme has been shown to be secure under the strong Diffie-Hellman assumption for the bilinear mapping scheme discussed above [1, 2].

Note that all calculations are performed within cyclic groups. The '/' operator denotes inversion.

3 VMP

3.1 Summary

The user stores a commitment to an output address in the transaction used to send the payment to the mixer. This transaction (and therefore the commitment) are visible to all users of the Bitcoin protocol. The mixer publishes all of its receiving addresses and outgoing transactions, without revealing which input payment corresponds to each output payment. Then, after a delay sufficient to allow the mixer to pay out, a zero-knowledge set membership proof is used to show that the address committed to by the user is a member of the set of addresses payed by the mixer.

The mixer can prove to third parties not only that it payed the user, but that it payed the user *enough*. This is achieved by reducing the set of outgoing transactions to the set of outgoing transactions of sufficient value. This potentially reduces the anonymity of the mixing operation, simply due to restrictions in the size of the set being analyzed. This is discussed in section 4: Losses in Anonymity.

This protocol assumes secure communication channels between the mixer and users, as well as between the mixer and verifiers. All assumptions inherent in the security of Bitcoin are also held here.

3.2 User-Mixer Interaction

The goal of this interaction is to establish a commitment to an output address. First, the user sends the mixer an address, m, which is where the user wants the mixer to send the 'mixed' coins. The mixer should require that this address has never been used before, to guarantee uniqueness, which greatly simplifies the proof process. Then, the mixer sends the user a random r. I am not aware of any possible exploitation based on users selecting particular values for r, but this allows each party a highly unpredictable input to the commitment. The mixer and the user both construct the commitment $C = g^m h^r$. This commitment will be included in the transaction by having a "fake" output in the transaction point to the commitment string instead of a real address.

When the mixer tells the user where to send the input coins, the user checks to see if that address is one of the receiving addresses published by the mixer. The purpose of publishing the receiving addresses is to allow verifiers to know that they are looking at all input transactions.

If a payment is made to an address that verifiers don't know about, it would be possible to steal it without being provably untrustworthy. If a mixer ever removes an address from its published set of receiving addresses, verifiers should immediately label that mixer untrustworthy, with one exception. If the mixer has already paid back every input transaction to that address, it can be removed. Verifiers should continue to watch the removed address for a few blocks after removal to make sure the mixer does not receive any payments from users who did not detect that the address was removed in time.

Publishing input addresses also protects the mixer from attacks on its reputation by liars. It is no longer possible to claim that a transaction was a payment to the mixer unless the transaction pays one of the mixer's actual addresses. Furthermore, in the case of an attacker paying one of a mixer's real addresses but using an improper commitment (by committing to something that isn't an address, for instance) the mixer should simply return the coins to sender.

Finally, the user sends the coins to the mixer and the commitment is locked into the blockchain.

3.3 Verifier-Mixer Interaction

After some amount of time (specified by the mixer) has passed, it is guaranteed that the mixer will have payed out to the intended address for the transaction in question. At that point, any third-party verifier may confirm that the mixer did indeed pay out.

The mixer maintains a publicly available list of its outgoing payment transactions, T. The output address of any transaction T_i is T_{Ai} and the value of that transaction is T_{Vi}. All elements in T_A are unique. In order to confirm that at least the expected value v was paid, T may be reduced to only include each T_i for which $T_{Vi} \geq v$. The expected value is determined by mixer fees and may be a range, in which case the minimum expected value should be used. T is trusted to be complete because the mixer only stands to fail verification by

omitting elements from T.

Information known only to the mixer is:

- σ, the address committed to by the user
- r, the random value such that $C = g^{\sigma} h^{r}$
- x, a private key used to generate signatures

Information published by the mixer (or in the blockchain) is:

- g and h, the group generators used to create the commitment
- C, the commitment
- T, the set of all outgoing transactions from the mixer to its users where $\forall T_i \in T : T_{Vi} \geq v$. v is the minimum expected value.

The following process is exactly the set membership proof constructed by Camenisch, Chaabouni, and Shelat [1]. For VMP, the mixer is the prover, and $\Phi = T_A$. It is included here for completeness.

Common Input: g, h, a commitment C, and a set Φ
Prover Input: σ, r such that $C = g^{\sigma} h^{r}$ and $\sigma \in \Phi$.

P	$\xleftarrow{y, \{A_i\}}$ V	Verifier picks $x \in_R \mathbb{Z}_p$ and sends $y \leftarrow g^{x}$ and $A_i \leftarrow g^{\frac{1}{x+i}}$ for every $i \in \Phi$.
P	$\xrightarrow{V}$ V	Prover picks $v \in_R \mathbb{Z}_p$ and sends $V \leftarrow A_{\sigma}^{v}$.
Prover and Verifier run $\mathrm{PK}\{(\sigma, r, v) : C = g^{\sigma} h^{r} \ \wedge \ V = g^{\frac{v}{x+\sigma}}\}$		
P	$\xrightarrow{a, D}$ V	Prover picks $s, t, m \in_R \mathbb{Z}_p$ and sends $a \leftarrow e(V, g)^{-s} e(g, g)^{t}$ and $D \leftarrow g^{s} h^{m}$.
P	$\xleftarrow{c}$ V	Verifier sends a random challenge $c \in_R \mathbb{Z}_p$.
P	$\xrightarrow{z_{\sigma}, z_v, z_r}$ V	Prover sends $z_{\sigma} \leftarrow s - \sigma c, z_v \leftarrow t - vc$, and $z_r \leftarrow m - rc$. Verifier checks that $D \stackrel{?}{=} C^{c} h^{z_r} g^{z_{\sigma}}$ and that $a \stackrel{?}{=} e(V, y)^{c} \cdot e(V, g)^{-z_{\sigma}} \cdot e(g, g)^{z_v}$

For a discussion of the security of all steps described above, please consult that paper.

This process is highly efficient when multiple payments are being verified. The bottleneck in this exchange is the verifier sending $|T|$ elements to the mixer. However, those signatures can be reused for further checks as long as no elements in a previous T are removed from the T for the current check. In other words, as long as verification starts with large transactions and includes progressively smaller ones, it is possible to efficiently verify a large number of payments without resending any signatures.

4 Losses of Anonymity

4.1 Reduced Anonymity Sets

The largest drawback of VMP is that is requires the mixer to publish all of their incoming and outgoing payments. This is necessary to enable payment verification, but it can limit the size of the anonymity set for any given input transaction.

For the naive implementation with a set fee, it is in fact possible to link each input with an output, by identifying the output corresponding to the most expensive input ($|T|$ would be 1) and repeating on the now reduced set of unlinked transactions.

This problem isn't solved by a changed to VMP, but instead by good application of traditional mixing methods of maintaining anonymity. Large throughput volume, highly variable fees, or specific allowed payments sizes can all increase the size of the anonymity set for input transactions. In particular, it would make sense to have a cap on payment size so that there is no single largest transaction. Large payments can always be split up into several smaller mixing demands for this purpose.

4.2 Multiple Output Addresses

One common traditional mixing method of improving anonymity is to allow users to specify multiple output addresses, and pay part of the mixed coins to each address. Then, given an expected value v based on an input payment, instead of looking at all T_i such that $T_{Vi} \geq v$, an attacker would have to look at all T_i, T_j such that $T_{Vi} + T_{Vj} \geq v$.

It is possible to implement arbitrarily many output addresses with VMP. The extension to the protocol is as follows:

- For single output addresses, $\Phi = T_A$. For multiple output addresses, Φ is the set of concatenated addresses $T_{Ai}|T_{Aj}$ where $T_{Ai}, T_{Aj} \in T_A$ and $i \neq j$. If the number of addresses $N \geq 3$, N distinct addresses would be concatenated. The addresses should be sorted before concatenation to eliminate redundancy.

  The size of Φ increases exponentially with N. However, each signature only ever has to be constructed once, so this would be feasible to perform over time for small N. If the number of output addresses is unknown to the verifier, it is possible to construct Φ as the union of the signature sets for each $N \leq k$, where k is the maximum number of output addresses permitted. k must be public information.

  Φ would be limited to only contain groups of elements in T which have great enough value.

  Formally, given a set $S \subset T$ such that $|S| \leq N$, $cat(S_A) \in \Phi$ if and only if $\sum_{S_i \in S} S_{Vi} \geq v$, where v is the minimum expected payout value.

- The commitment C now commits to multiple output addresses (again, each should never have been used before). This is done by concatenating the (sorted) addresses and using the result of the concatenation as σ in $C = g^{\sigma}h^{r}$. This does not allow the user to specify how many bitcoins are sent to each address, as long as the sum is large enough. However, the mixer improves its effectiveness by splitting reasonably, and therefore has an incentive to do so. It could already expose users if it wanted to.

To verify that a user was paid in a multiple output scheme, perform the exact same set-membership proof as described for the single-address case. Because Φ consists of groups of addresses, just the same as the commitment, the verifier will be able to verify if the set of addresses committed to by the user is a member of the set of sets of addresses which were paid sufficient funds to potentially be a payout for that input transaction.

5 References

1. Jan Camenisch, Rafik Chaabouni, and Abhi Shelat. Efficient Protocols for Set Membership and Range Proofs. In *Advances in Cryptology - ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*, pages 234-252. Springer, 2008.

2. Dan Boneh and Xavier Boyen. Short Signatures Without Random Oracles and the SDH Assumption in Bilinear Groups. In *Journal of Cryptology*, volume 21, issue 2, pages 149-177. Springer, 2007.

3. Halgren, Kolla, Sen, and Zhang. Making Classical Honest Verifier Zero Knowledge Protocols Secure Against Quantum Attacks. In Automata, Languages, and Programming volume 5126 of *Lecture Notes in Computer Science*, pages 592-603. Springer, 2008.