Toward Unlinkable Bitcoin Transactions

Andrew Poelstra[*]and Gregory Maxwell[†]

2014-10-05 (commit `42acd2f`)

Abstract

The appearance of Bitcoin in 2009[Nak09] has enabled the trustless transfer of funds by means of a publically verifiable distributed ledger. However, this ledger exposes all transactions, resulting in extremely poor privacy for Bitcoin users.

In this paper, we describe some new technologies that would reduce the amount of publically inferrable information on the Bitcoin blockchain. We start with the selectively-linkable ring signatures first proposed in CryptoNote[vS13], and introduce modifications to (a) allow signatures by multisigner sets who satisfy arbitrary threshold circuits, such that the resulting signatures are indistinguishable from ordinary single-party ones; (b) combine signatures for multiple outputs to compress transactions.

We also introduce a novel mechanism for output value hiding, which allows an output of size N to be plausibly included in a ring signature for inputs whose sizes are any $M \leq N$.

Finally, we describe a mechanism for using *one-way aggregatable signatures* [Mou13] to remove the linking between inputs and outputs within a block, and introduce an improved version of this scheme which greatly improves efficiency by eliminating the need for bilinear groups. The cost is a minor trust requirement on miners to not reveal the original input/output mappings.

[*] apoelstra@math.utexas.edu

[†] greg@xiph.org

1 Introduction

Privacy-preserving digital money was first introduced by David Chaum in 1983[Cha83]. In Chaum's construction, every transaction was required to go through a central mint, whose job was to issue currency and prevent double-spending. To maintain privacy even against the mint itself, the mint issued currency by blindsigning tokens whose unblinded form was known only to the recipient. This allowed the mint to recognize its own signed tokens without being able to associate them to individual issuances.

This construction relied heavily on the central mint, but its honest operation could not be cryptographically enforced. This meant that the mint had to be an identifiable, trusted entity, limiting usefulness in an online world where such parties are rare.

Bitcoin[Nak09] was introduced in 2009 as a digital currency scheme with no trusted party. In this system, honest behaviour is verified by all participants in the system (*e.g.* to make sure that all spent coins have a real and valid history), and double-spending is prevented by use of a dynamic membership multiparty signature[BCD$^+$14] which eliminates both the trusted party and the central point of failure.

However, to allow public verifiability, Bitcoin's design requires all transactions be broadcast in cleartext, exposing a linkage between payer and payee to all participants. This leads to very poor privacy[Hea13]. Improving Bitcoin's privacy while maintaining public verifiability is an active area of research[Poe14], at the forefront of which is the use of ring signatures described by van Saberhagen[vS13] for use with the `CryptoNote` technology.

In this article we propose several improvements to van Saberhagen's scheme.

1.1 Related Work

research focused around inefficient accumulator designs (zerocoin), trusted setup (zerocash + SNARK noodling). more feasible improvements include he values (adam), owas ("y. m. mouton") (pairing => slow), MAST (partial script hiding, cite?)

cryptonote introduced a special form of linkable ring signatures suitable for use in Bitcoin. The bulk of this paper discusses incremenetal improvements on this design

our improvements:

Threshold-Circuit Multisignatures. Consider a set $S = \{S_i\}$ of n signers, and let C be a threshold circuit with n inputs. For a subset S' of the signer set, we say the subset satisfies the circuit C if C outputs true after setting each input wire i to true iff $S_i \in S$. We would like to produce a ring signature verification key $\mathtt{vk}$

2 Background

Definition 1. *(Monotone Access Structure[Bei96]) Let $\{P_i\}$ be a set of n parties. A collection $\mathbb{A} \subset \mathcal{P}(\{P_i\})$ is a* monotone access structure *if for all $B \subseteq C \in \mathcal{P}(\{P_i\})$, if $B \in \mathbb{A}$ then $C \in \mathbb{A}$.*

We will use monotone access structures to represent sets of signers who must cooperate to produce a signature; the monotonicity requirement reflects the fact that if any set of signers can produce a signature, certainly any superset of them will be able to.

As shown in [BL88], any monotone access structure can be described as the set of satisfying inputs to a threshold circuit[1]. We assume from here on that our monotone access structures are described by threshold circuits.

Let $\mathcal{T}$ be a threshold circuit; for a node $x \in \mathcal{T}$ we write n_x for the number of inputs to x, k_x for the number of accepting inputs required for x to accept, and Parent(x) for x's parent node, if applicable.

Definition 2. *(Ring Signature[RST01]) A* ring signature scheme *consists of the following three algorithms:*

- $\texttt{Gen}(1^\lambda)$ *takes a security parameter and outputs a pair* $(\texttt{sk}, \texttt{pk})$ *of a signing and public key.*
- $\texttt{Sign}(m, \{\texttt{pk}_i\}_{i=1}^n, s, \texttt{sk})$ *takes a message* m*, set* $\{\texttt{pk}_i\}$ *of public keys (called a* ring*), the secret key for public key* $\texttt{pk}_s$*. It outputs a signature* σ*.*
- $\texttt{Verify}(m, \sigma)$ *takes a message* m *and signature* σ *and outputs either true or false.*

Intuitively, a ring signature is a signature from a set of signers such that any one of the set could have produced the signature; however, no adversary is able to determine which one with probability non-negligibly different from guessing. We will later define a specific type of ring signature and provide precise security definitions.

Definition 3. *(Lagrange coefficient). For a finite set* $S \subset \mathbb{N}$*, we write* $\Delta_{i,S} = \prod_{\substack{j \in S \\ j \neq i}} \frac{-j}{i-j}$*. Then for a polynomial* p *of degree* $(|S|+1)$ *and evaluations* p_i *of* p *at* $i \in S$*, we have that*

$$p(0) = \sum_{i \in S} \Delta_{i,S} p_i$$

(When p *is a random polynomial in* $\mathbb{Z}_q$ *for large prime* q*, this is Shamir's threshold secret sharing scheme [cite] for the secret value* $p(0)$*.)*

3 Bytecoin Ring Signatures (BRS)

In [vS13], van Saberhagen developed a selectively-linkable ring signature scheme for the specific application of blockchain-based currencies. In this scheme, each signature contains a *key image* which uniquely specifies the actual signing key used for the signature, allowing the network to detect double-spends, since they would appear as the use of the same key twice. However, the key image is constructed so that identifying the signing key (in the absense of a double-spend attempt) is equivalent to the Diffie-Hellman problem.

The signature scheme is based on the traceable ring signature scheme by Fujisaki and Suzuki[FS06], who developed a ring signature for the purpose of voting such that any reuse of the

[1] A threshold circuit is one for which each gate is labelled by a number k such that it accepts iff at least k of its inputs are enabled. For a gate with n inputs, $\texttt{AND}$ is obtained by setting $k = n$ and $\texttt{OR}$ by setting $k = 1$.

same key in the same vote (as identified by a *tag*, or arbitrary string, embedded in each signature) with the same ring could be detected; van Saberhagen introduced the key image and simplified the protocol by removing this tag-linkability.

In this section we improve these ring signatures to allow the use of multiple signers; that is, signing keys associated to sets of signers that can only be used by a subset which satisfies some threshold circuit determined at the time the key is created. Since the resulting keys are ordinary discrete-log keys, the resulting signatures are indistinguishable from single-signer ones. This is good for privacy and also implies $O(1)$ signature size and verification cost in the number of signers.

We also introduce a method of compressing multiple signatures which use the same ring, to create "k-of-m" ring signatures for which the exact subset of keys used cannot be determined. However, the signatures still have m key images embedded in them, and therefore double-spending remains impossible. This gives significant space savings for transactions with multiple inputs.

3.1 Precursor: Schnorr Signatures for Threshold Circuits

Before describing the full construction, we provide an example of our threshold circuit multisignature scheme applied to Schnorr signatures[Sch89][2]. For a threshold circuit $\mathcal{T}$, these signatures will require an interactive setup phase with Depth($\mathcal{T}$) rounds of interaction; signing is $O(n)$ in the number of signers used to satisfy $\mathcal{T}$; signature size and verification time are $O(1)$.

The scheme works as follows. Suppose we have a discrete-log group $\mathcal{G}$ with generator G and order q. Suppose also that we have a hash function H modelled as a random oracle.

- $\texttt{Gen}(1^\lambda, \mathcal{T})$. Let n be the number of inputs to $\mathcal{T}$. Each signer $i \in [1, n]$ generates[3] a uniformly random keypair $(\texttt{sk}_i, \texttt{pk}_i = \texttt{sk}_i G)$ and sends $\texttt{pk}_i$ to all members of the group.

  Then consider the arithmetic circuit $\mathcal{T}'$ on $\mathbb{Z}_q$ constructed from $\mathcal{T}$ by copying its graph structure and considering every node to be an addition gate. Then the output of $\mathcal{T}'$ after setting each input wire i to $\texttt{pk}_i$ is the verification key $\texttt{vk}$.

- $\texttt{Setup}(\mathcal{T})$. Again, n is the number of inputs to $\mathcal{T}$. Let $d = \text{Depth}(\mathcal{T})$.

  Each party $i \in [1, n]$ chooses a nonce $r_i \in \mathbb{Z}_q$ and publishes $r_i G$. All parties are able to compute a shared nonce rG as follows: evaluate the addition circuit $\mathcal{T}'$ from $\texttt{Gen}$ with the ith input set to $r_i G$.

  Next, each party i distributes her secret key x_i and secret nonce r_i recursively as follows. Label the output wire of $\mathcal{T}$ with (r_i, x_i). Then for each layer of the circuit, starting from the topmost (*i.e.* closest to the output), consider the set $\{d_j\}$ of gates in that layer. For each d_j, let k_{d_j} be its threshold number of accepting inputs and n_{d_j} be its total number of inputs. d_j's output wire will be labelled with some number o_j from the previous step (or the initial labelling of the circuit's output wire). Split o_j according to a linear secret sharing scheme into n_j shares: choose random $(k_j + 1)$-degree polynomials p and q and

[2] This construction is based on an idea communicated to us in person by Dan Boneh.

[3] Note that the actual number of signers may be less than the number of inputs, if some signers occur multiple times. In this case they should generate a new keypair for each input wire that they are assigned: for the purposes of the protocol, they should act as multiple distinct signers.

write $(x^i_\ell, r^i_\ell) = (p(\ell), q(\ell))$ for $\ell \in [1, n_j]$. Then k_j shares are required to construct the original secrets. Label each input wire ℓ with share (x^i_ℓ, r^i_ℓ).

After this process, every input wire $\mathcal{T}$ will be labelled with some secret share. For each wire, party i gives its share to the party corresponding to that wire. (Note that party i may be distributing a share to herself.)

- $\texttt{Sign}(m, I \subset [1, n], \{x_i\}_{i \in I})$. Assume that I is an accepting input to $\mathcal{T}$; otherwise output $\perp$.

  Each party $i \in I$ is assigned a single input wire of $\mathcal{T}$. Call this input wire's gate d_i. Then party i has n_{d_i} secret shares (x^ℓ_i, r^ℓ_i) for $\ell \in [1, n_{d_i}]$. She computes

  $$\sigma_i = r_i + \Delta_{i,I} \sum_{\ell=1}^{n_{d_i}} [r^\ell_i + x^\ell_i e], \qquad e = H(m, rG)$$

  All parties in I compute $e = H(m, rG)$, where rG is the shared nonce from the setup phase. Then each party $i \in I$ computes $s_i = r_i - x^0_i e$. Each s_i is added together, and each x_i for $j \notin I$ is added to this, to obtain a value s. The signature is published as (s, e).

- $\texttt{Verify}(m, (s, e), \texttt{vk})$. As with the ordinary Schnorr signature scheme, $\texttt{Verify}$ computes $rG = sG + e\texttt{vk}$ and checks that $e = H(m, rG)$.

AND gate optimization. We observe that for AND gates (*i.e.* threshold gates for which $k = n$), there is a simpler linear secret sharing scheme: for a secret x, the dealer's share is x, and no other party gets a share. This is an important optimization because it eliminates a round of interaction.

3.2 Construction

Bytecoin Ring Signatures. We term our construction a *Bytecoin Ring Signature scheme* or *BRS scheme*, where the name Bytecoin is an homage to the cryptocurrency bytecoin in which van Saberhagen's ring signatures were first applied. A BRS scheme consists of the following algorithms:

- $\texttt{Gen}(1^\lambda, \mathcal{T}, \texttt{PP})$. Inputs a security parameter, public parameters $\texttt{PP}$, and a circuit $\mathcal{T}$ whose inputs correspond to signers and whose accepting inputs correspond to subsets of signers who may generate signatures.

  For each signer i, outputs a pair $(\texttt{sk}_i, \texttt{pk}_i)$: $\texttt{pk}_i$ should be sent to the other signers while $\texttt{sk}_i$ should be secret. Also output a verification key $\texttt{vk}$.

- $\texttt{Setup}(\mathcal{T})$. An interactive protocol in which each of m parties input

- $\texttt{Sign}(\texttt{PP})$.

- $\texttt{Verify}(m, \sigma)$.

- $\texttt{Link}($

3.3 Multiple-Input Ring Signatures

4 Output Value Hiding

5 One-Way Aggregatable Signatures

References

[BCD$^+$14] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille, *Enabling blockchain innovations with pegged sidechains*, 2014, `http://www.blockstream.com/sidechains.pdf`.

[Bei96] A. Beimel, *Secure schemes for secret sharing and key distribution*, Ph.D. thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[BL88] J. Benaloh and J. Leichter, *Generalized secret sharing and monotone functions*, Advances in Cryptology — CRYPTO, vol. 403, Springer-Verlag, January 1988, `http://research.microsoft.com/apps/pubs/default.aspx?id=68345`, pp. 27–36.

[Cha83] D. Chaum, *Blind signatures for untraceable payments*, Advances in Cryptology Proceedings of Crypto **82** (1983), no. 3, 199–203.

[FS06] E. Fujisaki and K. Suzuki, *Traceable ring signature*, Cryptology ePrint Archive, Report 2006/389, 2006, `http://eprint.iacr.org/2006/389`.

[Hea13] M. Hearn, *Merge avoidance: Privacy enhancing techniques in the bitcoin protocol*, 2013, `http://www.coindesk.com/merge-avoidance-privacy-bitcoin/`.

[Mou13] Y. M. Mouton, *Increasing anonymity in Bitcoin ... (possible alternative to Zerocoin?)*, 2013, BitcoinTalk post, `https://bitcointalk.org/index.php?topic=290971`.

[Nak09] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2009, `https://www.bitcoin.org/bitcoin.pdf`.

[Poe14] A. Poelstra, *Is there any _true_ anonymous cryptocurrencies?*, Bitcoin.SE, 2014, `http://bitcoin.stackexchange.com/a/29473`.

[RST01] R. L. Rivest, A. Shamir, and Y. Tauman, *How to leak a secret*, Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology (London, UK, UK), ASIACRYPT '01, Springer-Verlag, 2001, pp. 552–565.

[Sch89] C. P. Schnorr, *Efficient identification and signatures for smart cards*, Proceedings of CRYPTO
'89, 1989, ftp://utopia.hacktic.nl/pub/mirrors/Advances%20in%20Cryptology/HTML/PDF/C89/239.PDF.

[vS13] N. van Saberhagen, *Cryptonote v 2.0*, https://cryptonote.org/whitepaper.pdf, 2013.