Anonymous Zether: Technical Report

Benjamin E. Diamond
J.P. Morgan

Abstract

We describe a protocol for the *anonymous Zether* payment system of Bünz, Agrawal, Zamani, and Boneh [BAZB]. We first discuss "Σ-Bullets"—which adapt *Bulletproofs* to the setting of ElGamal ciphertexts—and address shortcomings in the approach of [BAZB].

We further study the *anonymous* extension proposed in [BAZB, §D]. Extending Groth and Kohlweiss's *one out of many* proofs [GK15], we prove knowledge of a secret permutation (of a certain form) of ElGamal ciphertexts with heterogeneous keys. We finally give an efficient implementation based on the number-theoretic transform.

Introduction

Zether is a cryptographic protocol for confidential payment, described in a manuscript of Bünz, Agrawal, Zamani and Boneh [BAZB]. In contrast to well-known protocols like Monero and Zcash, Zether is "account-based", and features constant long-term space overhead per participant (though see also Quisquis [FMMO]); Zether also lacks a trusted setup.

The manuscript [BAZB] focuses on *basic Zether*, in which account balances and transfer amounts are concealed, but participants' identities are not. Its central cryptographic technique, called "Σ-Bullets", seeks to adapt Bulletproofs [BBB$^+$] (originally designed for Pedersen commitments) to the setting of ElGamal ciphertexts. We argue in Section 2 below that the Σ-Bullets protocol of [BBB$^+$] has security flaws, and describe concrete attacks on it (targeting both soundness and zero-knowledge). We also propose a secure amendment.

In Section 3, we turn to *anonymous* payment, as proposed in [BAZB, §D]. The appendix [BAZB, §D] suggests a relation, but no proof protocol; our work proposes one. We extend *one out of many proofs*—introduced in Groth and Kohlweiss [GK15] and extended by Bootle, Cerulli, Chaidos, Ghadafi, Groth, and Petit [BCC$^+$15]—so as to prove knowledge not just of a secret index but of a secret *permutation* (of a certain form) of ElGamal ciphertexts; we further allow ciphertexts encrypted under heterogeneous keys. Our protocol conducts a variant of basic Zether on these *permuted* (re-encrypted) ciphertexts. Our security proofs show, among other things, that knowledge of discrete logarithm relations on these permuted ciphertexts carries over to the originals (a technique which may be of independent interest).

We observe further that our permutation protocol entails, computationally, a handful of discrete circular convolutions; we make our protocol efficient using ideas related to the number-theoretic transform. We finally describe an implementation of our system, and report its performance characteristics.

1 Review of Basic and Anonymous Zether

We briefly summarize both basic and anonymous Zether; for further details we refer to [BAZB].

Zether's global state consists of a mapping acc from ElGamal public keys to ElGamal ciphertexts; y's table entry contains an encryption of y's balance b (in the exponent). Schematically, we can write:

$$\mathsf{acc}\colon \mathbb{G} \to \mathbb{G}^2,$$
$$y \mapsto \mathsf{acc}[y] = \mathsf{Enc}_y(b, r) = \left(g^b y^r, g^r\right),$$

for some randomness r which y in general does not know. (For details on the synchronization issues surrounding "epochs", we refer to [BAZB].)

1.1 Basic Zether

In "basic" (non-anonymous) Zether, a non-anonymous sender y may transfer funds to a non-anonymous recipient $\overline{y}$. To do this, y should publish the public keys y and $\overline{y}$, as well as a pair of ciphertexts (C, D) and $(\overline{C}, D)$ (the randomness may be shared). These should encrypt, under y and $\overline{y}$'s keys, the quantities g^{b^*} and g^{-b^*}, respectively, for some integer $b^* \in \{0, \dots \mathsf{MAX}\}$ (MAX is a fixed constant of the form $2^n - 1$). To apply the transfer, the administering system (e.g., smart contract) should group-subtract (C, D) and $(\overline{C}, D)$ from y and $\overline{y}$'s account balances (respectively). We denote by (C_{Ln}, C_{Rn}) y's balance *after* the homomorphic deduction is performed.

Finally, the prover should prove knowledge of:

- sk for which $g^{\mathsf{sk}} = y$ (knowledge of secret key),
- r for which:
  - $g^r = D$ (knowledge of randomness),
  - $(y \cdot \overline{y})^r = (C \cdot \overline{C})$ (ciphertexts encrypt opposite balances),
- b^* and b^* in $\{0, \dots, \mathsf{MAX}\}$ for which $C = g^{b^*} \cdot D$ and $C_{Ln} = g^{b'} \cdot C_{Rn}$ (overflow and overdraft protection).

Formally, we have the relation below, which essentially reproduces [BAZB, (2)]:

$$\mathsf{st}_{\mathsf{ConfTransfer}} : \left\{ y, \overline{y}, C_{Ln}, C_{Rn}, C, \overline{C}, D, g; \mathsf{sk}, b^*, b', r : \begin{array}{c} C = g^{b^*} \cdot D^{\mathsf{sk}} \wedge C_{Ln} = g^{b'} \cdot C_{Rn}^{\mathsf{sk}} \wedge (y \cdot \overline{y})^r = C \cdot \overline{C} \wedge \\ g^{\mathsf{sk}} = y \wedge D = g^r \wedge b^* \in \{0, \dots, \mathsf{MAX}\} \wedge b' \in \{0, \dots, \mathsf{MAX}\} \end{array} \right\}.$$

1.2 Anonymous Zether

In anonymous Zether [BAZB, §D], a sender may hide herself and the recipient in a larger "ring" $(y_i)_{i=0}^{N-1}$. To an observer, it should be impossible to discern which among a ring's members sent or received funds. Specifically, a sender should choose a list $(y_i)_{i=0}^{N-1}$, as well as indices idx_0 and idx_1 for which y_{idx_0} and y_{idx_1} belong to the sender and recipient, respectively. The sender should then publish this list, as well as a list of ciphertexts $(C_i, D)_{i=0}^{N-1}$ for which (C_{idx_0}, D) encrypts g^{b^*} under y_{idx_0}, (C_{idx_1}, D) encrypts g^{-b^*} under y_{idx_1}, and (C_i, D) for each $i \notin \{\mathsf{idx}_0, \mathsf{idx}_1\}$ encrypts g^0 under y_i. To apply the transfer, the contract should deduct each (C_i, D) from y_i's balance; we denote the list of *new* balances by $(C_{Ln,i}, C_{Rn,i})_{i=0}^{N-1}$.

Finally, the prover should prove knowledge of:

- $\mathsf{idx}_0, \mathsf{idx}_1 \in \{0, \dots, N-1\}$ (sender's and recipient's secret indices),
- sk for which $g^{\mathsf{sk}} = y_{\mathsf{idx}_0}$ (knowledge of secret key),
- r for which:
  - $g^r = D$ (knowledge of randomness),
  - $(y_{\mathsf{idx}_0} \cdot y_{\mathsf{idx}_1})^r = C_{\mathsf{idx}_0} \cdot C_{\mathsf{idx}_1}$ (sender's and receiver's ciphertexts encrypt opposite balances),
  - for each $i \notin \{\mathsf{idx}_0, \mathsf{idx}_1\}$, $y_i^r = C_i$ (all ciphertexts other than the sender's and recipient's encrypt 0),
- b^* and b^* in $\{0, \dots, \mathsf{MAX}\}$ for which $C_{\mathsf{idx}_0} = g^{b^*} \cdot D$ and $C_{Ln,\mathsf{idx}_0} = g^{b'} \cdot C_{Rn,\mathsf{idx}_0}$ (overflow and overdraft protection).

We group these facts into a formal relation, closely following [BAZB, (8)]. For technical reasons (described below), we actually prove a slight variant of this relation, in which N is required to be *even* and idx_0 and idx_1 are required to have opposite parity. Formally:

$$\mathsf{st}_{\mathsf{AnonTransfer}} : \left\{ \begin{array}{l} \left((y_i, C_i, C_{Ln,i}, C_{Rn,i})_{i=0}^{N-1}, D, u, g, g_{\mathsf{epoch}}; \mathsf{sk}, b^*, b', r, (s_i, t_i)_{i=0}^{N-1}\right) : \\ \prod_{i=0}^{N-1} C_i^{s_i} = g^{b^*} \prod_{i=0}^{N-1} y_i^{r \cdot s_i} \wedge \prod_{i=0}^{N-1} C_i^{s_i+t_i} = \prod_{i=0}^{N-1} y_i^{r\cdot(s_i+t_i)} \wedge D = g^r \wedge \\ \left(C_i^{(1-s_i)(1-t_i)} = y_i^{r\cdot(1-s_i)(1-t_i)}\right)_{i=0}^{N-1} \wedge \prod_{i=0}^{N-1} C_{Ln,i}^{s_i} = g^{b'} \left(\prod_{i=0}^{N-1} C_{Rn,i}^{s_i}\right)^{\mathsf{sk}} \wedge \\ g^{\mathsf{sk}} = \prod_{i=0}^{N-1} y_i^{s_i} \wedge g_{\mathsf{epoch}}^{\mathsf{sk}} = u \wedge b^* \in \{0, \ldots, \mathsf{MAX}\} \wedge b' \in \{0, \ldots, \mathsf{MAX}\} \\ (s_i \in \{0,1\} \wedge t_i \in \{0,1\})_{i=0}^{N-1} \wedge \sum_{i=0}^{N-1} s_i = \sum_{i=0}^{N-1} t_i = 1 \wedge \\ N \equiv 0 \mod 2 \wedge s_{i_0} = t_{i_1} = 1 \implies i_0 \not\equiv i_1 \mod 2 \end{array} \right\}. \tag{1}$$

The requirement that $\mathsf{idx}_0 \not\equiv \mathsf{idx}_1 \mod 2$ decreases privacy, but not by much. Indeed, this requirement decreases the cardinality of the set of possible pairs $(\mathsf{idx}_0, \mathsf{idx}_1) \in \{0, \ldots, N-1\}^2$ from N^2 to $\frac{N^2}{2}$; this latter cardinality of course still grows quadratically in N.

2 Bulletproofs and ElGamal Ciphertexts

We now describe the "Σ-Bullets" protocol of [BAZB], and our improvements. In fact, the protocol as described in relation 10 (page 48) of [BAZB, §G] is not complete as written; to see this, note that $\left(\frac{C^c}{D^{s_{\mathsf{sk}}}}\right)^{z^2} \cdot \left(\frac{C_{Ln}^c}{C_{Rn}^{s_{\mathsf{sk}}}}\right)^{z^3} = \left(D^{z^2} \cdot C_{Rn}^{z^3}\right)^{-k_{\mathsf{sk}}} \cdot g^{c\cdot(b^*\cdot z^2 + b'\cdot z^3)}$, whereas $g^{s_b} = g^{k_b} \cdot g^{c\cdot(b^*\cdot z^2 + b'\cdot z^3)}$. We thus consider the version implemented in the repository `bbuenz / BulletProofLib`. In this version, the the prover sends A_t, where

$$A_t = \left(D^{z^2} \cdot C_{Rn}^{z^3}\right)^{k_{\mathsf{sk}}};$$

the verifier then checks

$$g^{c\cdot\hat{t}} \cdot h^{c\cdot\tau_x} \stackrel{?}{=} g^{c\cdot\delta(y,z)} \cdot A_t \cdot {c_{\mathsf{commit}}}^{-1} \cdot \left(T_1^x \cdot T_2^{x^2}\right)^c, \tag{2}$$

where $c_{\mathsf{commit}} = \left(D^{z^2} \cdot C_{Rn}^{z^3}\right)^{s_{\mathsf{sk}}} \cdot \left(C^{z^2} \cdot C_{Ln}^{z^3}\right)^{-c}$.

2.1 Attacks on [BAZB]

Attack (Soundness)**.** Informally, nothing prevents the message of (C, D) from having an "h part". Suppose that the prover were to construct (C, D) so as to encrypt the message $g^{b^*}h^\gamma$, for some nonzero γ (and b^* as usual). (The recipient's ciphertext, $(\overline{C}, D)$, would encrypt $g^{-b^*}h^{-\gamma}$.) By adding the constant term $z^2\cdot\gamma$ to τ_x (as implemented in `BulletProofLib`, τ_x has no constant term) the prover can preserve the validity of equation 2; on the other hand, the message $g^{b^*}h^\gamma$ would completely invalidate the recipient's ciphertext (and the sender's).

Attack (Zero knowledge)**.** Informally, the term $A_t = \left(D^{z^2} \cdot C_{Rn}^{z^3}\right)^{k_{\mathsf{sk}}}$ allows the verifier to decrypt the combined ciphertext $(C, D)^{z^2} \cdot (C_{Ln}, C_{Rn})^{z^3}$. Indeed, after honest behavior by the prover, the term $A_t \cdot {c_{\mathsf{commit}}}^{-1}$ equals $g^{c\cdot(b^*\cdot z^2 + b'\cdot z^3)}$; from this quantity, the verifier may brute-force the values b^* and b' using only 2^{64} work (and much less, if the verifier can "guess" these values sooner).

2.2 A refined "Σ-Bullets"

We propose a modified version of "Σ-Bullets" which addresses both of the issues. In fact, this is a *generic* approach for applying Bulletproofs to ElGamal-encrypted integers (in the exponent). For notational ease, we specialize to the case of basic Zether.

We informally describe our amended version; a detailed protocol (which also includes anonymity) can be found in Section 6 below. Alongside the initial commitments A and S, the prover should publish additional ciphertexts (C', D') and (C'_{Ln}, C'_{Rn}) which encrypt h^{γ^*} and $h^{\gamma'}$ respectively (for randomly chosen scalars γ^* and γ'). Upon receiving x, the prover should add to τ_x the constant term $z^2 \cdot \gamma^* + z^3 \cdot \gamma'$. During the sigma protocols, in addition to proving knowledge of sk for which $g^{\mathsf{sk}} = y$, the prover should *also* prove knowledge of b^*, b', γ^*, and γ' for which:

$$g^{b^*} = D^{-\mathsf{sk}} \cdot C, \quad g^{b'} = C_{Rn}^{-\mathsf{sk}} \cdot C_{Ln}, \quad h^{\gamma^*} = D'^{-\mathsf{sk}} \cdot C', \quad h^{\gamma'} = C'^{-\mathsf{sk}}_{Rn} \cdot C'_{Ln}.$$

Finally, the prover instead defines $A_t = \left((D \cdot D')^{z^2} \cdot (C_{Rn} \cdot C'_{Rn})^{z^3}\right)^{k_{\mathsf{sk}}}$, while in the check 2 the verifier instead uses $c_{\mathsf{commit}} = \left((D \cdot D')^{z^2} \cdot (C_{Rn} \cdot C'_{Rn})^{z^3}\right)^{s_{\mathsf{sk}}} \cdot \left((C \cdot C')^{z^2} \cdot (C_{Ln} \cdot C'_{Ln})^{z^3}\right)^{-c}$.

The additional sigma-proofs mitigate the soundness attack: they ensure that (C, D) and (C_{Ln}, C_{Rn}) only have "g parts" (and that (C', D') and (C'_{Ln}, C'_{Rn}) only have "h parts"). The ciphertexts (C', D') and (C'_{Ln}, C'_{Rn}) themselves serve to blind the messages of (C, D) and (C_{Ln}, C_{Rn}). Indeed, the verifier may only decrypt a linear combination of all four ciphertexts; we note that $A_t \cdot {c_{\mathsf{commit}}}^{-1} = \left(g^{b^*} h^{\gamma^*}\right)^{c \cdot z^2} \cdot \left(g^{b'} h^{\gamma'}\right)^{c \cdot z^3}$. This decryption reveals nothing about b^* and b', and can be "fed into" the standard Bulletproofs protocol.

3 Anonymous Payments

3.1 One-out-of-many proofs and vector rotations

We now discuss our approach to anonymity. We begin with *one out of many proofs*, introduced by Groth and Kohlweiss [GK15] and extended by Bootle, Cerulli, Chaidos, Ghadafi, Groth, and Petit [BCC+15]. These techniques serve to prove knowledge—given a fixed list of commitments—of a secret element among this list and an opening for this element to the message 0.

In fact, these techniques admit more flexibility than is explicitly described in [GK15] and [BCC+15]. An *intermediate* step of these protocols entails the construction of a "re-encryption" of the secret list element (same message but new randomness); in the final step, the prover simply reveals this re-encryption's randomness (the verifier may check explicitly whether it opens to 0). Instead of revealing the re-encryption's randomness, however, the prover may use it in further protocols. This idea will be key in our work.

The first challenge is that [GK15] and [BCC+15] require homomorphic commitment schemes, like Pedersen commitments or ElGamal encryptions under *the same* key. Using minor adjustments, we extend these protocols so as to support lists of ElGamal encryptions belonging to *arbitrary* public keys.

The most significant challenge of [GK15] and [BCC+15] is that only *one* re-encryption is delivered, whereas we want something more like a permutation. Informally, we want to first run [BCC+15] twice, so as to deliver to the verifier re-encryptions of (C_{idx_0}, D) and (C_{idx_1}, D) (while concealing these elements' indices in the original list); using fairly simple extensions, we may also deliver a re-encryption of $(C_{Ln,\mathsf{idx}_0}, C_{Rn,\mathsf{idx}_0})$ and "re-encryptions" (i.e., exponentiations) of y_{idx_0} and y_{idx_1}, where the secrets idx_0 and idx_1 are provably chosen consistently throughout. Having in hand these re-encrypted values, the prover and verifier may conduct basic Zether on them. (That the protocol remains sound even when conducted on *re-encryptions* is non-trivial, but true, as we show below.)

All this says nothing, however, about the remaining equalities $y_i^r = C_i$ for $i \notin \{\mathsf{idx}_0, \mathsf{idx}_1\}$, which present a serious difficulty. A naïve approach would essentially conduct [BCC+15] N times, adopting certain additional modifications to ensure that the secret index is chosen distinctly each time. These modifications—at least as we are able to see—are incompatible with the logarithmic optimizations of

[GK15], however, so that this approach would demand $\mathcal{O}(N^2)$ communication ($\mathcal{O}(N \log N)$ communication would *perhaps* be possible, though non-trivial).

We reduce the communication to $\mathcal{O}(N)$ using the following trick, which necessitates that we further deconstruct the protocols [GK15] and [BCC+15]. A key step in these protocols is the definition of a vector $(p_i(X))_{i=0}^{N-1}$ of polynomials, and the delivery to the verifier of their *evaluations* $(p_i(w))_{i=0}^{N-1}$ at a challenge w. Because $p_i(X)$ is of "high degree" just when i equals the secret index idx_0, the verifier can use $(p_i(w))_{i=0}^{N-1}$ (plus some pre-challenge "correction terms") to pick out a re-encryption of the desired list element. Our observation is essentially that this vector may be reused and "rotated" in order to systematically pick out the *other* terms. Thus, the protocol need only be conducted once—or twice, rather—to handle (C_{idx_0}, D) and (C_{idx_1}, D); once this is done, the *same* vectors $(p_i(w))_{i=0}^{N-1}$ and $(q_i(w))_{i=0}^{N-1}$ (let's say) may be rotated by the verifier to obtain the remaining terms (in some still-unknown order). In fact, in each step we rotate both $(p_i(w))_{i=0}^{N-1}$ and $(q_i(w))_{i=0}^{N-1}$ by *two positions*; in precisely the case that N is even and idx_0 and idx_1 have opposite parity, this approach then yields each element of the original list exactly once. That the parities of the secret indices idx_0 and idx_1 are indeed opposite may be proven using variants of ideas which are already present in [GK15] and [BCC+15].

We thus construct a secret permutation $\kappa\colon \{0, \ldots, N-1\} \to \{0, \ldots, N-1\}$ of a special form. We begin with secret elements idx_0 and idx_1 of opposite parities; for arbitrary i, we then have that:

$$i \mapsto \kappa(i) = \begin{cases} \mathsf{idx}_0 + 2 \cdot k & \text{if } i = 2 \cdot k \\ \mathsf{idx}_1 + 2 \cdot k & \text{if } i = 2 \cdot k + 1. \end{cases}$$

We observe that such permutations κ are exactly those residing in the subgroup of $\mathbf{S}_N$ described by the generators $\langle (0, 1, 2, \ldots, N-1), (0, 2, 4, \ldots, N-2) \rangle$. In fact, these generators are "tight", in the sense that the natural map $\langle (0, 1, 2, \ldots, N-1) \rangle \times \langle (0, 2, 4, \ldots, N-2) \rangle \to \mathbf{S}_N$ is injective.

A further way to understand these permutations is through permutation *matrices*. Effectively, we send only the top two rows of an unknown matrix; by performing two-step rotations, we obtain the remaining $N-2$ rows. The ultimate matrix is a permutation matrix if and only if the top two rows attain the value 1 at indices of opposite parity. This is described in the figures below:

$$\begin{bmatrix} \underbrace{0, \ldots\ldots\ldots\ldots, 1, \ldots\ldots\ldots\ldots, 0}_{\text{1 only at index } \mathsf{idx}_0} \\ \underbrace{0, \ldots\ldots, 1, \ldots\ldots\ldots\ldots\ldots, 0}_{\text{1 only at index } \mathsf{idx}_1} \\ 0, \ldots\ldots\ldots\ldots\ldots\ldots, 1, \ldots, 0 \\ 0, \ldots\ldots\ldots\ldots\ldots, 1, \ldots\ldots\ldots, 0 \\ \vdots \\ 0, \ldots\ldots\ldots, 1, \ldots\ldots\ldots\ldots\ldots, 0 \\ 0, \ldots, 1, \ldots\ldots\ldots\ldots\ldots\ldots, 0 \end{bmatrix}$$

Figure 1: "Prover's view".

$$\begin{bmatrix} \text{———————} & (p_i(w))_{i=0}^{N-1} & \text{———————} \\ \text{———————} & (q_i(w))_{i=0}^{N-1} & \text{———————} \\ \text{———} \; \text{———————} & (p_i(w))_{i=0}^{N-1} & \text{———} \\ \text{———} \; \text{———————} & (q_i(w))_{i=0}^{N-1} & \text{———} \\ & \vdots & \\ \text{———} \; (p_i(w))_{i=0}^{N-1} & \text{———————} & \text{———————} \\ \text{———} \; (q_i(w))_{i=0}^{N-1} & \text{———————} & \text{———————} \end{bmatrix}$$

Figure 2: "Verifier's view".

The intuition behind our technique is that the prover doesn't *need* the freedom of choosing an arbitrary permutation. Indeed, as far as the prover and verifier are concerned, two permutations are "essentially the same" as long as they agree at the points 0 and 1. The prover and verifier may thus agree in advance on a scheme which, given prescribed (and secret!) values only at 0 and 1, constructs from these a full permutation.

We remark finally upon our choice of parameters in [BCC+15, Fig. 4, Fig. 5] (where the notation $N = n^m$ is used). [BCC+15] typically takes $n = \mathcal{O}(1), m = \mathcal{O}(\log N)$, though [BCC+15, p. 13] mentions the choice $m = \mathcal{O}(1)$. Indeed, this latter choice requires transmitting only "a constant number of group elements"; [BCC+15] cites its utility in settings where group elements are large compared to field elements.

Strikingly, this setting mirrors ours, in that we reuse our field elements, whereas we have to transmit

group elements for each among our $\mathcal{O}(N)$ re-encryptions. We accordingly adopt the choice $n = N, m = 2$, yielding $\mathcal{O}(N)$-sized proofs.

3.2 Circular convolutions and the number-theoretic transform

The main *computational* bottleneck of the above scheme concerns the matrix of Fig. 2 above. Indeed, in practice the verifier must "multiply" the vectors $(C_i)_{i=0}^{N-1}$ and $(y_i)_{i=0}^{N-1}$ of curve points by this matrix of scalars (we view the elliptic curve point group $\mathbb{G}$ as a module over $\mathbb{F}_q$). The prover too has to perform a similar matrix multiplication, in the process of deriving the correction terms. In fact, the above scheme implemented *naïvely* requires $\mathcal{O}(N^2)$ computation for both the prover and the verifier.

Our second key observation is that the matrix of Fig. 2 is a "circulant" matrix, or rather the interleaving of the even-indexed rows of two such matrices. The multiplication of $(C_i)_{i=0}^{N-1}$ (say) by this matrix, then, is exactly an interleaving of (the even indices of) two discrete circular convolutions—namely of $(C_i)_{i=0}^{N-1}$ by the vectors $(p_i(w))_{i=0}^{N-1}$ and $(q_i(w))_{i=0}^{N-1}$, respectively. This matrix multiplication can thus be evaluated in $\mathcal{O}(N \log N)$ time using the number-theoretic transform (or rather two of them), provided that N is a power of 2 and $\mathbb{F}_q$'s 2-adic roots of unity number at least N. Fortuitously, the curve used in Ethereum precompiles has a prime order q for which the 2-power-torsion of $(\mathbb{F}_q)^*$ has order 2^{28}—more than sufficient for our purposes. Indeed, this curve was selected with FFTs in mind.

These ideas originated with Pollard [Pol71], and are surveyed in [Nus82, §8]. In both settings, only the convolution of two *field-element* vectors is discussed. Our version—in which a vector of *module* elements (i.e., curve points) is convolved with a vector of field elements—appears to be absent from the literature, despite our having made a cursory search; in any case it naturally complements well-known techniques.

The authors [BAZB] claim that $\mathcal{O}(\log N)$ communication and $\mathcal{O}(N)$ computation are possible. We are not able to achieve these asymptotics.

4 Protocol Specification

We now specify in detail a zero-knowledge proof protocol for the statement $\mathsf{st}_{\mathsf{AnonTransfer}}$ 1 above, following the *Bulletproofs* paper of Bünz, Bootle, Boneh, Poelstra, Wuille and Maxwell [BBB$^+$] where applicable. We denote by n that integer for which $\mathsf{MAX} = 2^n - 1$, and by idx_0 and idx_1 those indices for which $s_{\mathsf{idx}_0} = 1$ and $t_{\mathsf{idx}_1} = 1$, respectively. We define and make use of the following functions:

- $\mathsf{Shift}(\mathbf{v}, i)$ circularly shifts the vector $\mathbf{v}$ of field elements (i.e., $\mathbb{F}_q$) by the integer i.
- $\mathsf{MultiExp}(\mathbf{V}, \mathbf{v})$ multi-exponentiates the vector $\mathbf{V}$ of curve points by the vector $\mathbf{v}$ of field elements.
- $\mathsf{Hadamard}(\mathbf{v_0}, \mathbf{v_1})$ returns the Hadamard (element-wise) product of two field vectors.

We mark in blue font those steps which do not appear in [BAZB], [BBB$^+$], or [BCC$^+$15].

Protocol Anonymous Zether

$\mathcal{P}$ **computes...** ▷ Begin Bulletproof [BBB$^+$, §4]
$\alpha, \rho \leftarrow_\$ \mathbb{Z}_q$
$\mathbf{a}_L \in \{0,1\}^{2 \cdot n}$ s.t. $\langle \mathbf{a}_{L[:n]}, \mathbf{2}^n \rangle = b^*, \langle \mathbf{a}_{L[n:]}, \mathbf{2}^n \rangle = b'$
$\mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^{2 \cdot n}$
$A = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R}$
$\mathbf{s}_L, \mathbf{s}_R \leftarrow_\$ \mathbb{Z}_q^{2 \cdot n}$
$S = h^\rho \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R}$
$r_P, r_Q, r_U, r_V, r_X, r_Y, \sigma_0, \ldots, \sigma_{\frac{N}{2}-1} \leftarrow_\$ \mathbb{Z}_q$ ▷ Begin one-out-of-many proof [BCC$^+$15]
for all $j \in \{0,1\}$ **do**
  sample $p_{j,1}, \ldots, p_{j,N-1} \leftarrow_\$ \mathbb{Z}_q$, set $p_{j,0} = -\sum_{i=1}^{N-1} p_{j,i}$
end for

set $(q_{0,0}, \ldots, q_{1,N-1}) = (s_0, \ldots, s_{N-1}, t_0, \ldots, t_{N-1})$
$P = \text{Com}(p_{0,0}, \ldots, p_{1,N-1}; r_P)$
$Q = \text{Com}(q_{0,0}, \ldots, q_{1,N-1}; r_Q)$
$U = \text{Com}((p_{j,i}(1 - 2q_{j,i}))_{j,i=0}^{1,N-1}; r_U)$
$V = \text{Com}(-p_{0,0}^2, \ldots, -p_{1,N-1}^2; r_V)$
$X = \text{Com}\left(\{(\sum_{k \equiv i \mod 2} p_{0,k}) \cdot (\sum_{k \equiv i \mod 2} p_{1,k})\}_{i \in \{0,1\}}, r_X\right)$
$Y = \text{Com}\left(\{\sum_{k \equiv i \mod 2} p_{\mathsf{idx}_i \not\equiv i \mod 2, k}\}_{i \in \{0,1\}}, r_Y\right)$ ▷ $\mathsf{idx}_i \not\equiv i \mod 2$ is interpreted as a bit.
$\widetilde{C_{Ln}} = \mathsf{MultiExp}\left((C_{Ln,i})_{i=0}^{N-1}, p_0\right) \cdot \left(g^{-b'} \cdot C_{Ln,\mathsf{idx}_0}\right)^{\sigma_0}$
$\widetilde{C_{Rn}} = \mathsf{MultiExp}\left((C_{Rn,i})_{i=0}^{N-1}, p_0\right) \cdot (C_{Rn,\mathsf{idx}_0})^{\sigma_0}$
for all $j \in \{0,1\}, i \in \{0, \ldots, \frac{N}{2} - 1\}$ **do**
$\widetilde{C_{j,i}} = \mathsf{MultiExp}\left((C_k)_{k=0}^{N-1}, \mathsf{Shift}(p_j, 2 \cdot i)\right) \cdot \left(y^r_{\mathsf{idx}_j + 2 \cdot i}\right)^{\sigma_i}$
$\widetilde{y_{j,i}} = \mathsf{MultiExp}\left((y_k)_{k=0}^{N-1}, \mathsf{Shift}(p_j, 2 \cdot i)\right) \cdot \left(y_{\mathsf{idx}_j + 2 \cdot i}\right)^{\sigma_i}$
end for
$\widetilde{D} = D^{\sigma_0}$
$\widetilde{g} = g^{\sigma_0}$
end $\mathcal{P}$
$\mathcal{P} \to \mathcal{V} : A, S, P, Q, U, V, X, Y, \widetilde{C_{Ln}}, \widetilde{C_{Rn}}, \left(\widetilde{C_{j,i}}, \widetilde{y_{j,i}}\right)_{j,i=0}^{1,N-1}, \widetilde{D}, \widetilde{g}$
$\mathcal{V} : w \leftarrow_\$ \mathbb{Z}_q$
$\mathcal{V} \to \mathcal{P} : w$
$\mathcal{P}$ **sets...**
for all $j \in \{0,1\}, i \in \{0, \ldots, N-1\}$ **do**
$f_{j,i} = q_{j,i} \cdot w + p_{j,i}$
end for
$z_P = r_Q \cdot w + r_P$
$z_U = r_U \cdot w + r_V$
$z_X = r_Y \cdot w + r_X$
$\gamma^*, \gamma', \zeta^*, \zeta' \leftarrow_\$ \mathbb{Z}_q$ ▷ Begin computation of blinding ciphertexts
$(C', D') = \left(h^{\gamma^* \cdot w} \cdot \overline{y_{0,0}}^{\zeta^*}, \overline{g}^{\zeta^*}\right)$
$(C'_{Ln}, C'_{Rn}) = \left(h^{\gamma' \cdot w} \cdot \overline{y_{0,0}}^{\zeta'}, \overline{g}^{\zeta'}\right)$
end $\mathcal{P}$
$\mathcal{P} \to \mathcal{V} : (f_{j,1}, \ldots, f_{j,N})$ **for** $j \in \{0,1\}, z_P, z_U, z_X, C', D', C'_{Ln}, C'_{Rn}$
$\mathcal{V} : y, z \leftarrow_\$ \mathbb{Z}_q$
$\mathcal{V} \to \mathcal{P} : y, z$
$\mathcal{P}$ **:**
$l(X) = (\mathbf{a}_L - z \cdot \mathbf{1}^n) + \mathbf{s}_L \cdot X$
$r(X) = \mathbf{y}^n \circ (\mathbf{a}_R + z \cdot \mathbf{1}^n + \mathbf{s}_R \cdot X) + z^2 \cdot (\mathbf{2}^n \parallel \mathbf{0}^n) + z^3 \cdot (\mathbf{0}^n \parallel \mathbf{2}^n)$
$t(X) = \langle l(X), r(X) \rangle = t_0 + t_1 \cdot X + t_2 \cdot X^2$ ▷ l and r are elements of $\mathbb{Z}_q^{2 \cdot n}[X]$; $t \in \mathbb{Z}_q[X]$
$\tau_1, \tau_2 \leftarrow_\$ \mathbb{Z}_q$
$T_i = g^{t_i} h^{\tau_i}$ **for** $i \in \{1, 2\}$
end $\mathcal{P}$
$\mathcal{P} \to \mathcal{V} : T_1, T_2$
$\mathcal{V} : x \leftarrow_\$ \mathbb{Z}_q$
$\mathcal{V} \to \mathcal{P} : x$
$\mathcal{P}$ **sets...**
$\mathbf{l} = l(x) = \mathbf{a}_L - z \cdot \mathbf{1}^{2 \cdot n} + \mathbf{s}_L \cdot x$
$\mathbf{r} = r(x) = \mathbf{y}^{2 \cdot n} \circ (\mathbf{a}_R + z \cdot \mathbf{1}^{2 \cdot n} + \mathbf{s}_R \cdot x) + z^2 \cdot (\mathbf{2}^n \parallel \mathbf{0}^n) + z^3 \cdot (\mathbf{0}^n \parallel \mathbf{2}^n)$

$\hat{t} = \langle \mathbf{l}, \mathbf{r} \rangle$ ▷ $\mathbf{l}$ and $\mathbf{r}$ are elements of $\mathbb{Z}_q^{2 \cdot n}$; $\hat{t} \in \mathbb{Z}_q$

$\tau_x = \tau_2 \cdot x^2 + \tau_1 \cdot x + z^2 \cdot \gamma^* + z^3 \cdot \gamma'$

$\mu = \alpha + \rho \cdot x$

$\overline{C_{Rn}} = (C_{Rn,\mathsf{idx}_0})^{w-\sigma_0}$ ▷ Prover "anticipates" certain secondary re-encryptions

for $j \in \{0,1\}, i \in \{0, \ldots, \frac{N}{2} - 1\}$ **do**

  $\overline{y_{j,i}} = \left(y_{\mathsf{idx}_j - 2i}\right)^{w-\sigma_i}$

end for

$\overline{D} = D^{w-\sigma_0}$

$\overline{g} = g^{w-\sigma_0}$

$A_y = \overline{g}^{k_{\mathsf{sk}}}$ ▷ Begin sigma protocol proving

$A_D = \overline{g}^{k_r}$

$A_u = g_{\mathsf{epoch}}^{k_{\mathsf{sk}}}$

$A_B = (\overline{y_{0,0}} \cdot \overline{y_{1,0}})^{k_r}$

$A_t = \left((\overline{D} \cdot D')^{z^2} \cdot (\overline{C_{Rn}} \cdot C'_{Rn})^{z^3} \right)^{k_{\mathsf{sk}}}$

for all $j \in \{0,1\}, i \in \{1, \ldots, \frac{N}{2} - 1\}$ **do**

  $A_{C_{j,i}} = (\overline{y_{j,i}})^{k_r}$

end for

$A_{\overline{C_{0,0}}} = g^{k_{v^*}} \cdot \overline{D}^{k_{\mathsf{sk}}}$

$A_{\overline{C_{Ln}}} = g^{k_{v'}} \cdot \overline{C_{Rn}}^{k_{\mathsf{sk}}}$

$A_{C'} = h^{k_{\nu^*}} \cdot D'^{k_{\mathsf{sk}}}$

$A_{C'_{Ln}} = h^{k_{\nu'}} \cdot C'^{k_{\mathsf{sk}}}_{Rn}$

end $\mathcal{P}$

$\mathcal{P} \to \mathcal{V} : \hat{t}, \tau_x, \mu, A_y, A_D, A_u, A_B, \left(A_{C_{j,i}}\right)_{j=0,i=1}^{1,\frac{N}{2}-1}, A_t, A_{\overline{C_{0,0}}}, A_{\overline{C_{Ln}}}, A_{C'}, A_{C'_{Ln}}$

$\mathcal{V} : c \leftarrow_{\$} \mathbb{Z}_q$

$\mathcal{V} \to \mathcal{P} : c$

$\mathcal{P}$ **sets...**

$s_{\mathsf{sk}} = k_{\mathsf{sk}} + c \cdot \mathsf{sk}$

$s_r = k_r + c \cdot r$

$s_{v^*} = k_{v^*} + c \cdot w \cdot b^*$

$s_{v'} = k_{v'} + c \cdot w \cdot b'$

$s_{\nu^*} = k_{\nu^*} + c \cdot w \cdot \gamma^*$

$s_{\nu'} = k_{\nu'} + c \cdot w \cdot \gamma'$

end $\mathcal{P}$

$\mathcal{P} \to \mathcal{V} : s_{\mathsf{sk}}, s_r, s_{v^*}, s_{v'}, s_{\nu^*}, s_{\nu'}$

$\mathcal{V}$ **:**

for all $j \in \{0,1\}$ **do**

  set $f_{j,0} = w - \sum_{i=1}^{N-1} f_{j,i}$

end for

$Q^w P \stackrel{?}{=} \mathrm{Com}(f_{0,0}, \ldots, f_{1,N-1}; z_P)$

$U^w V \stackrel{?}{=} \mathrm{Com}((f_{j,i}(w - f_{j,i}))_{j,i=0}^{1,N-1}; z_U)$

$Y^w X \stackrel{?}{=} \mathrm{Com}(\{(\sum_{k \equiv i \mod 2} f_{0,k}) \cdot (\sum_{k \equiv i \mod 2} f_{1,k})\}_{i \in \{0,1\}}; z_X)$ ▷ Opposite parity check

$\overline{C_{Ln}} = \mathsf{MultiExp}\left((C_{Ln,i})_{i=0}^{N-1}, f_0\right) \cdot (\widetilde{C_{Ln}})^{-1}$ ▷ Begin computation of secondary re-encryptions

$\overline{C_{Rn}} = \mathsf{MultiExp}\left((C_{Rn,i})_{i=0}^{N-1}, f_0\right) \cdot (\widetilde{C_{Rn}})^{-1}$

for all $j \in \{0,1\}, i \in \{0, \ldots, \frac{N}{2} - 1\}$ **do**

  $\overline{C_{j,i}} = \mathsf{MultiExp}\left((C_k)_{k=0}^{N-1}, \mathsf{Shift}(f_j, 2 \cdot i)\right) \cdot (\widetilde{C_{j,i}})^{-1}$

  $\overline{y_{j,i}} = \mathsf{MultiExp}\left((y_k)_{k=0}^{N-1}, \mathsf{Shift}(f_j, 2 \cdot i)\right) \cdot (\widetilde{y_{j,i}})^{-1}$

end for
$\overline{D} = D^w \cdot \widetilde{D}^{-1}$
$\overline{g} = g^w \cdot \widetilde{g}^{-1}$
$A_y \stackrel{?}{=} \overline{g}^{s_{\mathsf{sk}}} \cdot (\overline{y_{0,0}})^{-c}$ ▷ Begin sigma protocol verification
$A_D \stackrel{?}{=} \overline{g}^{s_r} \cdot \overline{D}^{-c}$
$A_u \stackrel{?}{=} g_{\mathsf{epoch}}^{s_{\mathsf{sk}}} \cdot u^{-c}$
$A_B \stackrel{?}{=} (\overline{y_{0,0}} \cdot \overline{y_{1,0}})^{s_r} \cdot \left(\overline{C_{0,0}} \cdot \overline{C_{1,0}}\right)^{-c}$
$\delta(y,z) = (z - z^2) \cdot \langle \mathbf{1}^{2 \cdot n}, \mathbf{y}^{2 \cdot n} \rangle - \left(z^3 \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle + z^4 \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle\right)$
$c_{\mathsf{commit}} = \left((\overline{D} \cdot D')^{z^2} \cdot (\overline{C_{Rn}} \cdot C'_{Rn})^{z^3}\right)^{s_{\mathsf{sk}}} \cdot \left((\overline{C_{0,0}} \cdot C')^{z^2} \cdot (\overline{C_{Ln}} \cdot C'_{Ln})^{z^3}\right)^{-c}$
$g^{w \cdot c \cdot \hat{t}} \cdot h^{w \cdot c \cdot \tau_x} \stackrel{?}{=} g^{w \cdot c \cdot \delta(y,z)} \cdot A_t \cdot {c_{\mathsf{commit}}}^{-1} \cdot \left(T_1^x \cdot T_2^{x^2}\right)^{w \cdot c}$
for all $j \in \{0,1\}, i \in \left\{1, \ldots, \frac{N}{2} - 1\right\}$ **do**
$A_{C_{j,i}} \stackrel{?}{=} \overline{y_{j,i}}^{s_r} \cdot \overline{C_{j,i}}^{-c}$
end for
$A_{\overline{C_{0,0}}} \stackrel{?}{=} g^{s_{v^*}} \cdot \overline{D}^{s_{\mathsf{sk}}} \cdot \overline{C_{0,0}}^{-c}$
$A_{\overline{C_{Ln}}} \stackrel{?}{=} g^{s_{v'}} \cdot \overline{C_{Rn}}^{s_{\mathsf{sk}}} \cdot \overline{C_{Ln}}^{-c}$
$A_{C'} \stackrel{?}{=} h^{s_{\nu^*}} \cdot D'^{s_{\mathsf{sk}}} \cdot C'^{-c}$
$A_{C'_{Ln}} \stackrel{?}{=} h^{s_{\nu'}} \cdot C'^{s_{\mathsf{sk}}}_{Rn} \cdot C'^{-c}_{Ln}$
end $\mathcal{V}$
$\mathbf{h}' = \left(h_1, h_2^{(y^{-1})}, h_3^{(y^{-2})}, \ldots, h_{2 \cdot n}^{(y^{-2 \cdot n + 1})}\right)$ ▷ Complete inner product argument
$Z = A \cdot S^x \cdot \mathbf{g}^{-z} \cdot \mathbf{h}'^{z \cdot \mathbf{y}^{2 \cdot n}} \cdot \mathbf{h}'^{z^2 \cdot (\mathbf{2}^n \| \mathbf{0}^n) + z^3 \cdot (\mathbf{0}^n \| \mathbf{2}^n)}$
$\mathcal{P}$ and $\mathcal{V}$ engage in Protocol 1 of [BBB$^+$] on inputs $(\mathbf{g}, \mathbf{h}', Zh^{-\mu}, \hat{t}; \mathbf{l}, \mathbf{r})$

5 Performance

We describe our implementation of Anonymous Zether. Verification takes place in Solidity contracts. Proving takes place in a JavaScript library, which is in turn invoked by our front-end (also written in JavaScript).

We report performance measurements below. We note that *gas used* includes not just verification itself, but also the relevant account maintenance associated with the Zether Smart Contract; our gas measurements *do incorporate EIP-1108*. The *verification time* we report reflects only the time taken by the local EVM in evaluating a read-only call to the verification contract. Proving time is self-explanatory. Each number next to *Transfer* indicates the size of the anonymity set used (including the actual sender and recipient). Our "Burn" transaction is actually a *partial burn*, in contrast to that of [BAZB]; in other words, we allow a user to withdraw only part of her balance (using a single range proof).

	Prov. Time (ms)	Verif. Time (ms)	Prf. Size (bytes)	Gas Used
Burn	907	83	1,312	2,393,134
Transfer (2)	1,847	118	2,720	4,989,138
Transfer (4)	1,919	139	3,104	6,011,011
Transfer (8)	2,185	170	3,872	8,286,426
Transfer (16)	2,830	241	5,408	13,396,675
Transfer (32)	4,355	402	8,480	24,847,667
Transfer (64)	7,977	742	14,624	50,544,849
Transfer (N)	$\mathcal{O}(N \log N)$	$\mathcal{O}(N \log N)$	$\mathcal{O}(N)$	$\mathcal{O}(N \log N)$

References

[BAZB] Benedikt Bünz, Shashank Agrawal, Mahdi Zamani, and Dan Boneh. Zether: Towards privacy in a smart contract world. Unpublished manuscript.

[BBB+] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. Full version.

[BCC+15] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. Short accountable ring signatures based on ddh. In Günther Pernul, Peter Y A Ryan, and Edgar Weippl, editors, *Computer Security – ESORICS 2015*, volume 9326 of *Lecture Notes in Computer Science*, pages 243–265. Springer International Publishing, 2015.

[BSCG+13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKS for C: Verifying program executions succinctly and in zero knowledge. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, volume 8043 of *Lecture Notes in Computer Science*, pages 90–108. Springer Berlin Heidelberg, 2013.

[FMMO] Prastudy Fauzi, Sarah Meiklejohn, Rebekah Mercer, and Claudio Orlandi. Quisquis: A new design for anonymous cryptocurrencies. Unpublished manuscript.

[GK15] Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, volume 9057 of *Lecture Notes in Computer Science*, pages 253–280. Springer Berlin Heidelberg, 2015.

[Nus82] H. Nussbaumer. *Fast Fourier Transform and Convolution Algorithms*. Springer-Verlag, 1982.

[Pol71] J. M. Pollard. The fast Fourier transform in a finite field. *Mathematics of Computation*, 25(114):365–374, 1971.