Anonymous Zether: Technical Report

Benjamin E. Diamond
J.P. Morgan / Quorum

Abstract

We describe a cryptographic protocol for *anonymous Zether*, originally proposed in Bünz, Agrawal, Zamani, and Boneh [BAZB]. In particular, we elucidate a "double circulant" approach to anonymous transfers, and 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]. The Zether protocol attains many *private payment* desiderata, combining trustlessness and deniability (as in Monero) while avoiding UTXO accumulation (cf. Monero, as well as Zcash). Zether also introduces the *account-based* model to private cryptocurrencies. More details can be found in [BAZB]; see also Fauzi, Meiklejohn, Mercer and Orlandi [FMMO] for additional discussion.

The manuscript [BAZB] focuses on *basic Zether*, in which account balances and transfer amounts are concealed, but participants' identities are not. In contrast, Appendix D of [BAZB] sketches an *anonymous* extension of Zether, in which a transaction's sender may hide herself and the transaction's recipient in a larger group of parties. The manuscript [BAZB] does not provide an explicit proof protocol for this anonymous extension; we aim to supply one in this technical note.

We briefly sketch our approach. We use *one-out-of-many proofs* to facilitate the transmission of a pair of secret bitstrings, with the aid of which the verifier may obtain the sender's and receiver's ciphertexts, respectively. After subjecting these ciphertexts to the basic Zether protocol, the verifier may then use these bitstrings as *bitmasks*, and in particular, *cicularly rotate* them so as to obtain the remaining ciphertexts (upon which sigma protocols may then be used). The ensuing multi-exponentiations by bitwise rotations in turn constitute exactly a pair of *circular convolutions*, and thus can be made fast (i.e., $\mathcal{O}(N \log N)$ instead of $\mathcal{O}(N^2)$) using ideas related to the fast Fourier transform. In particular, we adapt the *number-theoretic transform* to the setting of elliptic curve points, an innovation which may be of independent interest.

1 Overview

The manuscript of Bünz, Agrawal, Zamani, and Boneh sketches an anonymous transfer mechanism in Appendix D, and in particular provides a binary relation ([BAZB, (8)]). No cryptographic proof protocol for this relation is proposed, though its authors suggest using *one-out-of-many proofs*.

1.1 One-out-of-many proofs

The binary relation [BAZB, (8)] makes use of two secret bitstrings $(s_i, t_i)_{i=0}^{N-1}$, within each of which exactly one element is 1. One-out-of-many proofs (described in Groth and Kohlweiss [GK15] and in Bootle, Cerulli, Chaidos, Ghadafi, Groth, and Petit [BCC$^+$15]) indeed serve to secretly convey bitstrings of this form; in a simple instantiation of these ideas, a multi-exponentiation of some sequence of ElGamal ciphertexts—in which (a blinded version of) such a bitstring resides in the exponent—yields, together with the aid of an accompanying "zeroth-order adjustment" term, a secondary re-encryption of a *single* among these ciphertexts, chosen by the prover. (This is the special case $m = 1$ of [BCC$^+$15, Fig. 5].)

Using this technique, the prover may deliver to the verifier blinded versions of $(s_i)_{i=0}^{N-1}$ and $(t_i)_{i=0}^{N-1}$ (as in [BCC+15, Fig. 4]), and consequently secondary re-encryptions of those (C_i, D), $(C_{Ln,i}, C_{Rn,i})$, and (y_i, g) for which $s_i = 1$ or $t_i = 1$, respectively (see [BAZB, (8)]). These latter ciphertexts may then be directly fed into the original "Σ-Bullets" protocol of basic Zether [BAZB].

It is worth pausing to note the *separation* of the one-out-of-many proof from the basic Zether protocol. In essence, the prover need only deliver the sender's and receiver's ciphertexts to the verifier, while concealing these ciphertexts' indices in the original list; once the verifier has them, she may proceed as in basic Zether. The pair, of course, use the *re-encryptions* in all sigma protocols.

1.2 Even-order circular shifts

More challenging, on the other hand, is the demonstration in zero knowledge of the equalities $\left(C_i^{(1-s_i)\cdot(1-t_i)} = y_i^{(1-s_i)\cdot(1-t_i)\cdot r}\right)_{i=0}^{N-1}$. (These express that ciphertexts other than the sender's and receiver's encrypt zero.) In effect, the challenge is to isolate those C_i, y_i for which neither s_i nor t_i is true, without revealing the corresponding values i.

A naive remediation would commit to an N by N permutation *matrix*, whose status as such could be proven using a straightforward (two-dimensional) adaptation of [BCC+15, Fig. 4]. The first two rows would deliver secondary re-encryptions of those C_i, y_i for which s_i and t_i are true, respectively; the remaining rows would deliver the remaining elements.

We obviate this approach's concomitant quadratic-sized proof using a "double bit masking" technique. The idea is that, provided that we enforce that those respective i for which s_i and t_i are true have *opposite parity* (and that N is even), the *even*-order *circular shifts* of the bitstrings $(s_i)_{i=0}^{N-1}$ and $(t_i)_{i=0}^{N-1}$ collectively exhaust those length-N bitstrings containing only one 1. In other words, the blinded versions of these bitstrings—$(f_i)_{i=0}^{N-1}$ and $(g_i)_{i=0}^{N-1}$, let's say—serve to deliver re-encryptions of the sender's and recipient's (respectively) C_i and y_i; these vectors' circular shifts by nonzero even numbers, then, yield re-encryptions of the remaining C_i and y_i (upon which standard sigma protocols can finally be used). Of course, one still obtains quadratic growth in the number of possible sender–receiver pairs, even after restricting to pairs whose indices have opposite parity.

$$M = \begin{bmatrix} \text{————} \ (f_i)_{i=0}^{N-1} \ \text{————} \\ \text{————} \ (g_i)_{i=0}^{N-1} \ \text{————} \\ \text{——} \ \text{————} \ (f_i)_{i=0}^{N-1} \ \text{—} \\ \text{——} \ \text{————} \ (g_i)_{i=0}^{N-1} \ \text{—} \\ \vdots \\ \text{—} \ (f_i)_{i=0}^{N-1} \ \text{————} \ \text{——} \\ \text{—} \ (g_i)_{i=0}^{N-1} \ \text{————} \ \text{——} \end{bmatrix}$$

Figure 1: A depiction of the permutation matrix implicitly reconstructed by the verifier.

That the indices indeed have opposite parity can be proven using similar ideas. The Hadamard product of the respective *sums* of each blinded bitstring's even circular shifts contains components of at most degree one in the challenge (see [BCC+15, Fig. 4]) if and only if those i for which s_i and t_i (respectively) are true have opposite parities. The verifier may simply check, therefore, whether a multi-exponentiation by this Hadamard product can be eliminated by low-order terms supplied pre-challenge by the prover.

In effect, the prover need only transmit the *first two rows* of some permutation matrix, whose remainder may be straightforwardly reconstructed by the verifier. The resulting proof is of size $\mathcal{O}(N)$.

1.3 Circular convolutions and the number-theoretic transform

The proving and verification *time* of the above protocol, however, remain quadratic, as each circular shift in turn must serve as the vector of exponents in some length-N multi-exponentiation.

Indeed, the process described above may be viewed as the "matrix multiplication" of a fixed vector of curve points by a matrix of field elements, in the latter of which each pair of rows is obtained as two-step rotation of the previous pair. The matrix thus resembles a "circulant matrix" (see Fig. 1), and the matrix multiplication resembles a pair of circular convolutions.

That Fourier-theoretic techniques may be brought to bear on convolutions of this sort was first noticed apparently by Pollard [Pol71], who introduced an analogue of the fast Fourier transform to finite fields (now often called the *number-theoretic transform*; see [Nus82, §8]). In this procedure, modular-multiplicative roots of unity replace complex roots of unity; as in the complex case, the $\mathcal{O}(N \log N)$-time *Cooley–Tukey* algorithm may be used (see [Nus82, §1.1] for historical notes).

More subtle perhaps is that only the *module* structure of a signal's domain—and not its ring structure—arises throughout its role in the number-theoretic transform. Viewing an elliptic curve $E(\mathbb{F}_p)$ of order q as a module over $\mathbb{F}_q$, then, we may thus number-theoretically transform any "signal" consisting of points on E—even, crucially, when these points' exponents with respect to some fixed generator $G \in E$ are not known (and "field multiplication" cannot be performed). Relatedly, "Hadamard" or "signal–signal" multiplication in the frequency domain (as in [Pol71, (4)]) may be replaced by multi-exponentiation of a frequency-domain *vector* of points by a frequency-domain *exponent* of field elements.

We may thus carry through Pollard's number-theoretic transform to the circular convolution of an elliptic curve point vector by a field vector, and so conduct it in $\mathcal{O}(N \log N)$ time. (We accordingly require that N be a power of 2.) Fortuitously, the alt-bn128 curve used in Ethereum has an order q for which $\mathbb{F}_q$ admits unusually many 2-adic roots of unity; this is not a coincidence, and in fact the curve's creators had FFTs—though not of the module-theoretic variety—in mind [BSCG+13].

2 Specification

We now specify in detail a zero-knowledge proof protocol for the statement $\mathsf{st}_{\mathsf{AnonTransfer}}$ of [BAZB, (8)], 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 $t_{\mathsf{idx}_0} = 1$, respectively. We stipulate 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].

`// Begin Bulletproof [BBB+, §4]`

$\mathcal{P}$ *computes:*

- $\alpha, \beta \leftarrow_\$ \mathbb{Z}_q$
- $\mathbf{a}_L \in \{0,1\}^{2 \cdot n}$ s.t. $\langle \mathbf{a}_{L[:n]}, \mathbf{2}^n \rangle = b^*, \langle \mathbf{a}_{L[n:]}, \mathbf{2}^n \rangle = b'$
- $\mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^{2 \cdot n}$
- $A = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R}$
- $\mathbf{s}_L, \mathbf{s}_R \leftarrow_\$ \mathbb{Z}_q^{2 \cdot n}$
- $S = h^\beta \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R}$
- $\gamma, r_\gamma \leftarrow_\$ \mathbb{Z}_q$
- $H_L = h^\gamma {y_{\mathsf{idx}_0}}^{r_\gamma}$
- $H_R = g^{r_\gamma}$

$\mathcal{P} \rightarrow \mathcal{V} : A, S, H_L, H_R$

$\mathcal{V} : y, z \leftarrow_\$ \mathbb{Z}_q$

$\mathcal{V} \to \mathcal{P} : y, z$

$\mathcal{P}$ *computes:*

$l(X) = (\mathbf{a}_L - z \cdot \mathbf{1}^n) + \mathbf{s}_L \cdot X$

$r(X) = \mathbf{y}^n \circ (\mathbf{a}_R + z \cdot \mathbf{1}^n + \mathbf{s}_R \cdot X) + z^2 \cdot (\mathbf{2}^n \parallel \mathbf{0}^n) + z^3 \cdot (\mathbf{0}^n \parallel \mathbf{2}^n)$

$t(X) = \langle l(X), r(X) \rangle = t_0 + t_1 \cdot X + t_2 \cdot X^2$ `//` 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** $i \in \{1, 2\}$

$\mathcal{P} \to \mathcal{V} : T_1, T_2$

$\mathcal{V} : x \leftarrow\!\!\$\, \mathbb{Z}_q$

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

$\mathcal{P}$ *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` $\mathbb{Z}_q^{2 \cdot n}$`;` $\hat{t} \in \mathbb{Z}_q$

$\tau_x = \tau_2 \cdot x^2 + \tau_1 \cdot x + z^4 \cdot \gamma$

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

$\mathcal{P} \to \mathcal{V} : \hat{t}, \tau_x, \mu$

`// Begin One-Out-of-Many proof [BCC+15]`

$\mathcal{P}$ *computes:*

$r_P, r_Q, r_U, r_V, r_X, r_Y, \sigma \leftarrow\!\!\$\, \mathbb{Z}_q$

for $j \in \{0, 1\}$ **do** sample $p_{j,1}, \ldots, p_{j,N-1} \leftarrow\!\!\$\, \mathbb{Z}_q$, set $p_{j,0} = -\sum_{i=1}^{N-1} p_{j,i}$

set $(q_{0,0}, \ldots, q_{1,N-1}) = (s_0, \ldots, s_{N-1}, t_0, \ldots, t_{N-1})$

$P = \mathrm{Com}(p_{0,0}, \ldots, p_{1,N-1}; r_P)$

$Q = \mathrm{Com}(q_{0,0}, \ldots, q_{1,N-1}; r_Q)$

$U = \mathrm{Com}((p_{j,i}(1 - 2q_{j,i}))_{j,i=0}^{1,N-1}; r_U)$

$V = \mathrm{Com}(-p_{0,0}^2, \ldots, -p_{1,N-1}^2; r_V)$

$\widetilde{C_{Ln}} = \mathsf{MultiExp}\left((C_{Ln,i})_{i=0}^{N-1}, p_0\right) \cdot \left(g^{-b'} \cdot C_{Ln,\mathsf{idx}_0}\right)^{\sigma}$

$\widetilde{C_{Rn}} = \mathsf{MultiExp}\left((C_{Rn,i})_{i=0}^{N-1}, p_0\right) \cdot (C_{Rn,\mathsf{idx}_0})^{\sigma}$

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

$\widetilde{C_{j,i}} = \mathsf{MultiExp}\left((C_k)_{k=0}^{N-1}, \mathsf{Shift}(p_j, 2 \cdot i)\right) \cdot \left(y^r_{\mathsf{idx}_j - 2 \cdot i}\right)^{\sigma}$

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

end

$\widetilde{D} = D^{\sigma}$

$\widetilde{g} = g^{\sigma}$

$X = \mathrm{Com}\left(\left\{\left(\sum_{k \equiv i \mod 2} p_{0,k}\right) \cdot \left(\sum_{k \equiv i \mod 2} p_{1,k}\right)\right\}_{i \in \{0,1\}}, r_X\right)$

$Y = \mathrm{Com}\left(\left\{\sum_{k \equiv i \mod 2} p_{\mathsf{idx}_i \not\equiv i \mod 2, k}\right\}_{i \in \{0,1\}}, r_Y\right)$ `//` $\mathsf{idx}_i \not\equiv i \mod 2$ `is interpreted as a bit.`

$\mathcal{P} \to \mathcal{V} : P, Q, U, V, X, Y, \widetilde{C_{Ln}}, \widetilde{C_{Rn}}, \left(\widetilde{C_{j,i}}, \widetilde{y_{j,i}}\right)_{j,i=0}^{1,N-1}, \widetilde{D}, \widetilde{g}$

Each tilde-marked term sent by the prover constitutes a zeroth-order adjustment term, with the aid of which the verifier will ultimately derive a secondary re-encryption. The terms E and F facilitate the "opposite parity" proof, which we explain further here.

The *opposite parity condition* $\mathsf{idx}_0 \not\equiv \mathsf{idx}_1 \mod 2$ is equivalent to the linearity (in the challenge w) of the components of $\mathsf{Hadamard}(\mathbf{cycle}_0, \mathbf{cycle}_1)$, where each row $\mathbf{cycle}_j$ consists of the sum of f_j's even-order circular shifts. The prover's transmission (before seeing w) of this product's 0^{th}- and 1^{st}-order parts demonstrates this linearity. As a further optimization, we use only the first two indices of each row $\mathbf{cycle}_j$ (the remaining indices repeat these first two). In this light, the *opposite parity* proof philosophically evokes the original one-out-of-many proof, which essentially demonstrates (among other things) that both f_j and $\mathsf{Hadamard}(f_j, \mathbf{w} - f_j)$ consist of components linear in w. We use the exact same proof of linearity.

We compute the multi-exponentiations described above with the aid of a handful of module-theoretic Fourier transforms. For notational ease, we do not explicate this further.

$\mathcal{V} : w \leftarrow\!\!\$\, \mathbb{Z}_q$
$\mathcal{V} \to \mathcal{P} : w$
$\mathcal{P}$ *computes:*
  for $j \in \{0,1\}, i \in \{0, \ldots, N-1\}$ **do** $f_{j,i} = q_{j,i} \cdot w + p_{j,i}$
  $z_P = r_Q \cdot w + r_P$
  $z_U = r_U \cdot w + r_V$
  $z_X = r_Y \cdot w + r_X$
$\mathcal{P} \to \mathcal{V} : f_{j,1}, \ldots, f_{j,N-1}$ **for** $j \in \{0,1\}, z_P, z_U, z_X$
$\mathcal{V}$:
  for $j \in \{0,1\}$ **do** set $f_{j,0} = w - \sum_{i=1}^{N-1} f_{j,i}$
  $Q^w P \stackrel{?}{=} \mathrm{Com}(f_{0,0}, \ldots, f_{1,N-1}; z_P)$
  $U^w V \stackrel{?}{=} \mathrm{Com}((f_{j,i}(w - f_{j,i}))_{j,i=0}^{1,N-1}; z_U)$
  $Y^w X \stackrel{?}{=} \mathrm{Com}(\{(\sum_{k \equiv i \mod 2} f_{0,k}) \cdot (\sum_{k \equiv i \mod 2} f_{1,k})\}_{i \in \{0,1\}}; z_X)$ `// Opposite parity check`
$\mathcal{P}$ *computes:* `// Prover ‘‘locally anticipates’’ certain among the verifier’s re-encryptions`
  $\overline{C_{Rn}} = (C_{Rn,\mathsf{idx}_0})^{w-\sigma}$
  for $j \in \{0,1\}, i \in \left\{0, \ldots, \frac{N}{2} - 1\right\}$ **do** $\overline{y_{j,i}} = \left(y_{\mathsf{idx}_j - 2i}\right)^{w-\sigma}$
  $\overline{D} = D^{w-\sigma}$
  $\overline{g} = g^{w-\sigma}$
  `// Begin sigma protocol proving`
  $k_r, k_{\mathsf{sk}} \leftarrow\!\!\$\, \mathbb{Z}_q$
  $A_y = \overline{g}^{k_{\mathsf{sk}}}$
  $A_D = \overline{g}^{k_r}$
  $A_u = g_{\mathsf{epoch}}^{k_{\mathsf{sk}}}$
  $A_B = (\overline{y_{0,0}} \cdot \overline{y_{1,0}})^{k_r}$
  $A_t = \left(\overline{D}^{-z^2} \cdot \overline{C_{Rn}}^{z^3} \cdot {H_R}^{w \cdot z^4}\right)^{k_{\mathsf{sk}}}$
  for $j \in \{0,1\}, i \in \left\{1, \ldots, \frac{N}{2} - 1\right\}$ **do** $A_{C_{j,i}} = (\overline{y_{j,i}})^{k_r}$
$\mathcal{P} \to \mathcal{V} : A_y, A_D, A_u, A_B, A_t, \left(A_{C_{j,i}}\right)_{j=0,i=1}^{1,\frac{N}{2}-1}$
$\mathcal{V}$:
  $c \leftarrow\!\!\$\, \mathbb{Z}_q$
$\mathcal{V} \to \mathcal{P} : c$
$\mathcal{P}$ *computes:*
  $s_{\mathsf{sk}} = k_{\mathsf{sk}} + c \cdot \mathsf{sk}$
  $s_r = k_r + c \cdot r$
$\mathcal{P} \to \mathcal{V} : s_{\mathsf{sk}}, s_r$
$\mathcal{V}$:
  `// Begin computation of secondary re-encryptions`
  $\overline{C_{Ln}} = \mathsf{MultiExp}\left((C_{Ln,i})_{i=0}^{N-1}, f_0\right) \cdot (\widetilde{C_{Ln}})^{-1}$
  $\overline{C_{Rn}} = \mathsf{MultiExp}\left((C_{Rn,i})_{i=0}^{N-1}, f_0\right) \cdot (\widetilde{C_{Rn}})^{-1}$
  for $j \in \{0,1\}, i \in \{0, \ldots, \frac{N}{2} - 1\}$ **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
  $\overline{D} = D^w \cdot \widetilde{D}^{-1}$
  $\overline{g} = g^w \cdot \widetilde{g}^{-1}$

91 $\mathcal{V}$:

`// Begin sigma protocol verification`

92 $A_y \stackrel{?}{=} \overline{g}^{s_{\mathsf{sk}}} \cdot (\overline{y_{0,0}})^{-c}$

93 $A_D \stackrel{?}{=} \overline{g}^{s_r} \cdot \overline{D}^{-c}$

94 $A_u \stackrel{?}{=} g_{\mathsf{epoch}}^{s_{\mathsf{sk}}} \cdot u^{-c}$

95 $A_B \stackrel{?}{=} (\overline{y_{0,0}} \cdot \overline{y_{1,0}})^{s_r} \cdot (\overline{C_{0,0}} \cdot \overline{C_{1,0}})^{-c}$

96 $c_{\mathsf{commit}} = \left(\overline{D}^{-z^2} \cdot \overline{C_{Rn}}^{z^3} \cdot \overline{H_R}^{w \cdot z^4}\right)^{s_{\mathsf{sk}}} \cdot \left(\overline{C_{0,0}}^{-z^2} \cdot \overline{C_{Ln}}^{z^3} \cdot \overline{H_L}^{w \cdot z^4}\right)^{-c}$

97 $\delta(y, z) = (z - z^2) \cdot \langle \mathbf{1}^{2 \cdot n}, \mathbf{y}^{2 \cdot n} \rangle - \left(z^2 \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle + z^3 \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle\right)$

98 $A_t \stackrel{?}{=} g^{w \cdot c \cdot (\hat{t} - \delta(y,z))} \cdot h^{w \cdot c \cdot \tau_x} \cdot \left(T_1^x \cdot T_2^{x^2}\right)^{-w \cdot c} \cdot c_{\mathsf{commit}}$

99 **for** $j \in \{0, 1\}, i \in \left\{1, \ldots, \frac{N}{2} - 1\right\}$ **do** $A_{C_{j,i}} \stackrel{?}{=} \overline{y_{j,i}}^{s_r} \cdot \overline{C_{j,i}}^{-c}$

We record a few explanatory observations. The Σ-protocols presented here rely on the fact that $(\overline{C_{0,0}}, \overline{D})$ and $(\overline{C_{1,0}}, \overline{D})$ give secondary re-encryptions of (C_{idx_0}, D) and (C_{idx_1}, D) respectively, while $(\overline{C_{j,i}}, \overline{D})$ for $j \in \{0, 1\}$ and $i \in \{1, \ldots, \frac{N}{2} - 1\}$ give re-encryptions of the remaining (C_i, D) (all with randomness $r \cdot (w - \sigma)$); similarly, $(\overline{y_{0,0}}, \overline{g})$ and $(\overline{y_{1,0}}, \overline{g})$ give secondary re-encryptions of (y_{idx_0}, g) and (y_{idx_1}, g) respectively, while $(\overline{y_{j,i}}, \overline{g})$ for $j \in \{0, 1\}$ and $i \in \{1, \ldots, \frac{N}{2} - 1\}$ give re-encryptions of the remaining (y_i, g) (all with randomness $w - \sigma$). Finally, $(\overline{C_{Ln}}, \overline{C_{Rn}})$ gives a re-encryption of $(C_{Ln,\mathsf{idx}_0}, C_{Rn,\mathsf{idx}_0})$, with randomness $r' \cdot (w - \sigma)$ (we denote by r' the randomness of $(C_{Ln,\mathsf{idx}_0}, C_{Rn,\mathsf{idx}_0})$). The values of idx_0 and idx_1 of course are not revealed.

We comment further on the lines 96 and 98. The steps involving H_L, H_R, and γ mitigate an attack against basic Zether, which neglected to add 0^{th}-order randomness to τ_x in line 27—and hence to blind the "Pedersen commitment" $A_t \cdot (c_{\mathsf{commit}})^{-1} = g^{w \cdot c \cdot (\hat{t} - \delta(y,z))} \cdot h^{w \cdot c \cdot \tau_x} \cdot \left(T_1^x \cdot T_2^{x^2}\right)^{-w \cdot c}$. Indeed, absent our corrective measure, both sides of this equality yield (if the prover is honest) the quantity $g^{w \cdot c \cdot (b^* \cdot z^2 + b' \cdot z^3)}$, which in turn allows b' and b^* to be brute-forced by a malicious verifier using at worst 2^{64} work (and vastly less, if these values are "small" or "round").

Our treatment adds the randomness $\gamma \cdot z^4$ to τ_x and an extra term to c_{commit}, so that both sides of the above equality instead yield $\left(g^{b^* \cdot z^2 + b' \cdot z^3} \cdot h^{\gamma \cdot z^4}\right)^{w \cdot c}$—a quantity which information-theoretically hides $g^{b^* \cdot z^2 + b' \cdot z^3}$. Crucially, the Pedersen randomness h^γ—ElGamal encrypted under y_{idx_0} within (H_L, H_R)—never appears alone, and can arise only together with $g^{b^* \cdot z^2 + b' \cdot z^3}$ (via honest execution of line 98).

Our correction adds two curve points to the proof size, and necessitates a few extra multiplications for both the prover and the verifier. We reiterate that this issue is independent of—that is, prior to—the anonymous extension.

`// Complete inner product argument`

100 $\mathbf{h}' = \left(h_1, h_2^{(y^{-1})}, h_3^{(y^{-2})}, \ldots, h_{2 \cdot n}^{(y^{-2 \cdot n + 1})}\right)$

101 $Z = A \cdot S^x \cdot \mathbf{g}^{-z} \cdot \mathbf{h}'^{z \cdot \mathbf{y}^{2 \cdot n}} \cdot \mathbf{h}'^{z^2 \cdot (\mathbf{2}^n \| \mathbf{0}^n) + z^3 \cdot (\mathbf{0}^n \| \mathbf{2}^n)}$

102 $\mathcal{P}$ and $\mathcal{V}$ engage in Protocol 1 of [BBB+] on inputs $(\mathbf{g}, \mathbf{h}', Zh^{-\mu}, \hat{t}; \mathbf{l}, \mathbf{r})$

We remark finally upon our choice of parameters in the protocols [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 cf. [BCC+15, p. 13], which mentions the use of $m = \mathcal{O}(1)$). We adopt the unusual choice $n = N, m = 2$, in view of the fact that the prover must transmit correction terms not just once, but $\mathcal{O}(N)$ times (once for each re-encryption). Our choice, therefore, allows for proofs of $\mathcal{O}(N)$ size; the conventional approach would necessitate $\mathcal{O}(N \log N)$-sized proofs (and $\mathcal{O}(N \log^2 N)$ prover time).

We are not currently able to obtain $\mathcal{O}(\log N)$-sized proofs or $\mathcal{O}(N)$ prover–verifier time, although [BAZB] appears to suggest that such asymptotics are possible. We leave these tasks open for future work.

3 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 (measurements *do not* 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). *Partial burns* are explained below.

	Prov. Time (ms)	Verif. Time (ms)	Prf. Size (bytes)	Gas Used
Partial Burn	919	65	1,248	8,035,208
Transfer (2)	1,867	98	2,464	16,449,979
Transfer (4)	1,956	107	2,848	19,545,466
Transfer (8)	2,170	135	3,616	26,645,114
Transfer (16)	2,873	211	5,152	43,115,569
Transfer (32)	4,418	371	8,224	80,290,954
Transfer (64)	7,963	706	14,368	164,124,492
Transfer (N)	$\mathcal{O}(N \log N)$	$\mathcal{O}(N \log N)$	$\mathcal{O}(N)$	$\mathcal{O}(N \log N)$

4 Miscellaneous

We describe further miscellaneous adjustments we have made to the Zether protocol.

4.1 Partial burns

The burn transaction of Zether [BAZB] entails that a user's entire balance be wholly and instantly decrypted and withdrawn. One might wish, however, to *partially* burn her balance, or, in other words, to withdraw some portion of her balance from the contract without revealing the remaining balance. We facilitate *partial burns* of this sort using a simplification of the (non-anonymous) transfer statement of page 14—in which no recipient $\overline{y}$ need be specified, the "transfer" amount b^* is revealed publicly, and no randomness is used. In essence, one proves that her remaining balance conceals a non-negative number b' and that she knows her own secret key.

4.2 Rollovers *on demand* in confidential / "basic" Zether

We conclude with remarks on *basic Zether*, in which the identities of transactors are not concealed (but cf. below). While both the *basic* and *anonymous* versions of [BAZB] employ a "synchronized" paradigm, we suggest in basic Zether the elimination of epochs and the availability of rollovers "on demand".

As noted in [BAZB], in the basic setting asynchronous modifications to acc do not stand to jeopardize the in-progress transactions of other unwitting, honest provers. (This is what makes possible the immediate debit of transfers and withdrawals from acc, and hence the system's simpler replay protection mechanism. Deposits in basic Zether can also be credited immediately.) For identical reasons, asynchronous *rollovers* too don't stand to jeopardize others' transactions, and may be permitted freely.

More subtle is whether on-demand (or *purely* on-demand) rollovers could leak information about users' private balances. For example, given a client utility known to roll over just when its user's requested transfer amount exceeds her available (but not her *combined available and pending*) balance, the presence of a rollover—followed immediately, say, by a transfer—could reveal precisely this information about the client's user's balance.

Of course, this information doesn't mean much, especially after sufficiently many transfers have taken place. We deem this possible information leakage sufficiently negligible so as to be outweighed by the convenience of on-demand rollovers. We include this feature in our implementation of basic Zether.

4.3 "Hybrid" Zether in the basic paradigm

We further observe that the various simplifications attending basic Zether—including, in particular, those described above—necessitate only that the *sender* of each transfer reveal herself. Accordingly, we propose a "hybrid" Zether protocol in which the sender, but not the recipient, of each transaction is public. We use a simplified "single circulant" technique to obscure the actual recipient's C_i and y_i, and to generate re-encryptions of the false recipients' such elements for use in proof-of-exponent sigma protocols. In particular, epochs can remain absent.

This hybrid version is useful in cases in which the anonymity of the sender is *not* important, and maximal throughput is desired.

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.