WabiSabi - Draft v0.3

Ádám Ficsór, Yuval Kogman, István András Seres

October 20, 2020

Abstract

Bitcoin transfers value on a public ledger of transactions anyone can verify. Coin ownership is defined there in terms of public keys. Despite potential use for private transfers, research has shown that users' activity can often be traced in practice. Businesses have been built on dragnet surveillance of Bitcoin users because of this lack of strong privacy, which harms its fungibility, a basic property of functional money.

Although the public nature of this design lacks strong guarantees for privacy, it does not rule it out. A number of methods have been proposed to strengthen privacy. Among these is CoinJoin, which is an approach to structure transactions which adds ambiguity and breaks common assumptions that underlie heuristics used for surveillance. Existing implementations of CoinJoin present a number of limitations which may partly explain the lack of their widespread adoption.

This work introduces *WabiSabi*[1], a new protocol for centrally coordinated CoinJoin implementations utilizing keyed verification anonymous credentials and homomorphic value commitments. This improves earlier approaches which utilize blind signatures in both privacy and flexibility, enabling novel use cases and reduced overhead.

1 Introduction

Bitcoin transactions transfer funds by consuming unspent outputs of previous transactions as inputs to create new outputs. The protocol rules enforced by the network ensure that transactions do not arbitrarily inflate the money supply and that outputs are spent at most once. While some newer cryptocurrencies use more sophisticated approaches to define such rules, in Bitcoin the amounts as well as the specific outputs being spent are broadcast in the clear as part of the transaction. This presents significant challenges to transacting privately[2] as shown already in some of the earliest academic studies of Bitcoin [RH13; RS13; AKR+13; OKH13; MBB13; MPJ+13].

A fundamental requirement for electronic money is double spending prevention, and Bitcoin's main innovation was preventing double spending and illegal inflation without relying on a trusted authority and disintermediating transactions. This is in contrast to online e-cash schemes where a server authorizes transactions, or offline schemes where the identity of the double spender is revealed allowing the authority to intervene after the fact. Bitcoin's relative success suggests that the lack of trusted third parties factored more strongly in its adoption than the comparatively stronger privacy guarantees provided by the (possibly revocable) transaction anonymity traditionally associated with e-cash [DFT+97].

These shortcomings in privacy affect Bitcoin users, both individuals and organizations, leaving casual users especially vulnerable since power to surveil and resist surveillance is unevenly distributed [Rog15]. Even without address reuse, which is pervasive, transactions still reveal some information. This makes clustering of outputs according to heuristics practical [HF16], with wallets of some well known entities generally considered public knowledge. Exchanges complying KYC/AML requirements additionally must obtain and secure information that links transactions to personally identifying information.

The conditions for spending an output are specified in its `scriptPubKey`, typically requiring that the spending transaction be signed by a specific key. The signatures authorizing a transaction usually commit to the transaction in its entirety, making it possible for mutually distrusting parties to jointly create transactions without risking misallocation of funds: participants will only sign a proposed transaction after confirming that their desired outputs are included and the transaction is only valid when all parties have signed.

[1] The Japanese concept *wabi-sabi* is an aesthetic world view which among other things emphasizes acceptance of imperfections, in our case Bitcoin's challenges for privacy, and an appreciation for simplicity and economy.

[2] In this work we restrict the discussion of Bitcoin privacy to that of public ledger transactions, but there are other considerations especially at the network layer. For a more comprehensive discussion see `https://en.bitcoin.it/wiki/Privacy`.

1.1 Chaumian CoinJoin

Chaumian CoinJoin [Miz13; Max13a; FT17] is a privacy enhancing technique that uses the atomicity property of transactions and Chaumian blind signatures [Cha83] to construct collaborative Bitcoin transactions, also known as CoinJoins. Participants connect to a server, known as the coordinator, and submit their inputs and outputs using different anonymity network identities. That alone would provide anonymity but since outputs are unconstrained it's not robust against malicious users who may disrupt the protocol by claiming more than their fair share. To mitigate this the coordinator provides blind signatures representing units of standard denominations in response to submitted inputs. By unblinding and presenting this valid signature, the coordinator is unable to link the signed output to specific inputs but can be still verify that an output registration is authorized.

The use of standard denominations in the resulting CoinJoin transaction obscures the relationship between individual inputs and outputs, making the origins of each output ambiguous. Unfortunately standard denominations limit the use of privacy-enhanced outputs for payments of arbitrary amounts and result in a change output which maintains a link to the non-private input.

1.2 Limitations of Wasabi

In this work, we aim to improve on ZeroLink [FT17] as implemented by Wasabi, the most popular Chaumian CoinJoin implementation for Bitcoin. We identify several privacy shortcomings and inefficiencies of Wasabi CoinJoins. Some metrics comparing Wasabi, Samourai and other apparent CoinJoin transactions are provided. The "Other" category includes JoinMarket[3], but also has an inherent false positive error given these transactions are identified heuristically.

1.2.1 Denominations

Due to the nature of blind signatures, mixed outputs of Wasabi CoinJoins are restricted to fixed set of multiples of a base denomination[4]. This creates friction when sending or receiving arbitrary amounts of Bitcoin, as using fixed denomination generally creates change, both when mixing and when spending mixed outputs.

We define *CoinJoin inefficiency* as the fraction of non-mixed change outputs in a CoinJoin transaction, see fig. 1.

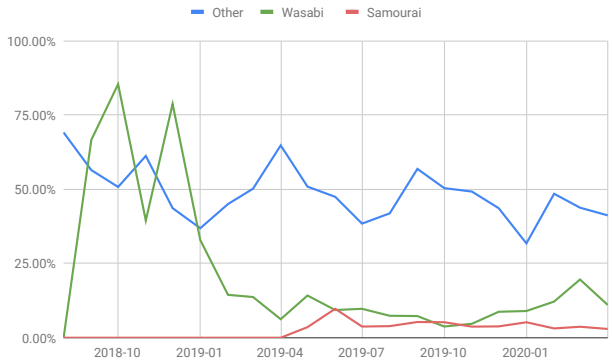


Figure 1: CoinJoin inefficiency of various privacy-focused Bitcoin wallets.

1.2.2 Minimum denomination

In order to participate, a user's combined input amount must be greater or equal to the base denomination.[5] We observe, that considerable portion of CoinJoin inputs are less than this minimum denomination, see fig. 2.

Even when users are able to provide several smaller value inputs with total value greater than the minimum denomination, the coordinator knows those inputs belong to the same user. In an ideal mixing protocol the coordinator should not obtain more information than the already available public ledger data by coordinating

[3] `https://github.com/JoinMarket-Org/joinmarket`

[4] Approximately 0.1₿

[5] The observed base denominations in Wasabi's CoinJoins are usually slightly higher than the announced, agreed upon base denomination. Thus participants sometimes get back slightly more value in the CoinJoins than they put in.

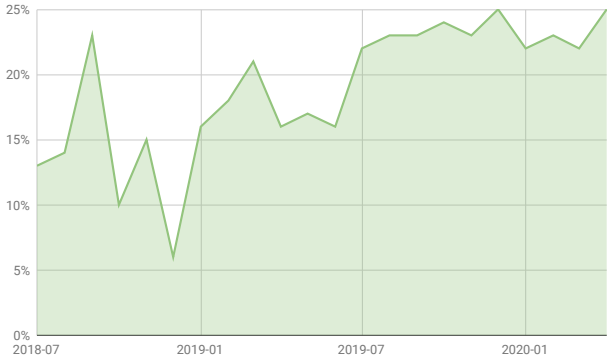


Figure 2: Fraction of inputs with value smaller than the minimum denomination in Wasabi CoinJoin transactions.

the CoinJoin transaction. This information removes many degrees of freedom when assigning non-derived subtransactions [MNF17] or link probability [MT15], reducing intrinsic ambiguity as well as the computational cost when evaluating potential links.

Furthermore if users consolidate coins before the CoinJoin in an additional transaction in order to be able to participate in a CoinJoin, then this link is revealed publicly based on the common input ownership heuristic [MPJ+13].

1.2.3 Varying denominations

Since users pay mining and coordination fees the denominations are gradually reduced between rounds of consecutive CoinJoins in order to make it possible for users to mix several times without providing additional inputs. This introduces a perverse incentive to minimize coordination fees by remixing in quick succession in order, resulting in a smaller anonymity set than with time-staggered remixes.

1.2.4 Block-space efficiency

The rigidity of the current transaction structure, i.e. fixed denominations, constrains users' unspent transaction output set structure as well. These limitations force users to consolidate their coins (see fig. 3) and create additional intermediate outputs with constrained amounts when interspersing CoinJoin transactions with transactions that send or receive value.

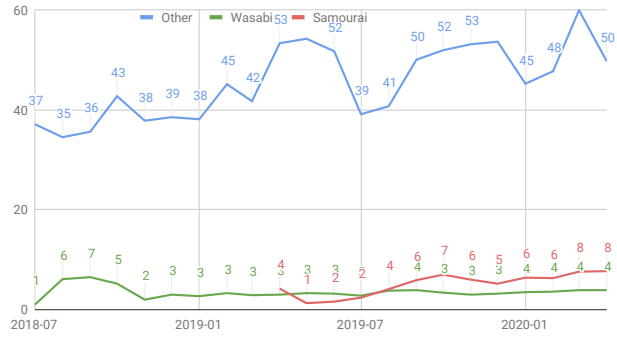


Figure 3: Average number of inputs from the first post-mix transactions in various CoinJoin schemes.[6]

1.2.5 Lack of privacy-enhanced payments

Currently Wasabi supports neither payments from a CoinJoin, nor payments in a CoinJoin. Payments from a CoinJoin would protect sender privacy and improve efficiency by requiring fewer intermediate outputs.

[6]The large figures for other CoinJoin-like transactions suggests false positives in our identification.

Payments within a CoinJoin would protect both sender and receiver privacy, and since they are a form of PayJoin[7] it would also improve privacy by introducing degrees of freedom in the interpretation of CoinJoins.

1.3 Our Contribution

We present WabiSabi, a generalization of Chaumian CoinJoin based on a keyed-verification anonymous credentials (KVAC) scheme [CPZ19]. The use of KVACs replaces blind signatures' standard denominations with homomorphic amount commitments, similar to Confidential Transactions [Max16], where the sum of any participant's outputs does not exceed that of their inputs while hiding the underlying values from the coordinator. In addition to being more flexible this improves privacy compared to blind signatures and standard denominations, since smaller inputs can be combined and change outputs created with the same unlinkability guarantees as those of standard denominations when using blind signatures[8]. WabiSabi builds on a successfully deployed and proven approach, aiming to reduce barriers to further adoption [DM06] by removing restrictions and strengthening unlinkability.

WabiSabi can be instantiated to construct a variety of CoinJoin transaction structures that depart from the standard output denomination convention, as in SharedCoin[9] and CashFusion[10] style transactions and Knapsack [MNF17] mixing. Payments from CoinJoin transactions are possible, as are payments within them, effectively a multiparty PayJoin that trades the steganographic properties for improved privacy from counterparties. Additionally, restrictions on consolidation of inputs can be removed, and there are opportunities for reducing unmixed change and relaxing minimum required denominations, and improved block space efficiency.

The rest of this paper is organized as follows. In section 2 we provide some background on our applied cryptographic building blocks. In section 3 we introduce our protocol, WabiSabi, at a high-level, while in section 4 we describe in-depth the cryptographic details of WabiSabi. In section 5 we argue about the security and privacy of WabiSabi. We review related work in section 6. Finally, we conclude our paper in section 7.

2 Preliminaries

Hereby we give an informal and high-level description of applied cryptographic primitives. In the following the security parameter is denoted as λ.

2.1 Commitment schemes

A commitment scheme allows one to commit to a chosen message while preventing them from changing the message after publishing the commitment. Secure commitments hide the chosen message until they are opened. We assume Pedersen commitments throughout this work.

$\mathsf{Commit}(m, r) \mapsto \mathcal{C}$**:** generate a commitment $\mathcal{C}$ to message m using randomness r.

$\mathsf{OpenCom}(\mathcal{C}, m, r) \mapsto \{True, False\}$**:** verify the correctness of the opening of a commitment by checking $\mathcal{C} \stackrel{?}{=} \mathsf{Commit}(m, r)$. If equality holds the algorithm outputs *True*, otherwise *False*.

2.2 MAC

A message authentication code (MAC) ensures the integrity of a message and consists of the following three probabilistic polynomial-time algorithms:

$\mathsf{GenMACKey}(\lambda) \mapsto \mathsf{sk}$**:** generate a secret key sk for MAC generation and verification.

$\mathsf{MAC}_{\mathsf{sk}}(m) \mapsto t$**:** generate a tag t on a message m using secret key sk.

$\mathsf{VerifyMAC}_{\mathsf{sk}}(m, t) \mapsto \{True, False\}$**:** verify that tag t is valid for message m using secret key sk.

One might intuitively think of a MAC as the symmetric-key counterpart of digital signatures. They both have the same goals and similar security requirements, however a MAC requires a secret rather than public key to verify.

[7] `https://en.bitcoin.it/wiki/PayJoin`

[8] Note that the cleartext amounts appearing in the final transaction might still link individual inputs and outputs.

[9] `https://github.com/sharedcoin/Sharedcoin`

[10] `https://github.com/cashshuffle/spec`

2.3 Zero-knowledge proofs of knowledge

A very high-level, and hence somewhat imprecise, description of zero-knowledge proofs is provided. This protocol involves a prover and a verifier. A prover wishes to prove that a relation $\mathcal{R}$ holds with respect to a secret input w, called witness, and public input x. Specifically, the prover wants to prove that $(x, w) \in \mathcal{R}$ without revealing anything about w.

$\mathsf{Prove}_{\mathcal{R}}(x, w) \mapsto \pi$**:** Given x and the private witness w the prover generates a proof π. For the specific outputs of Prove we use the notation of [CS97], with the witness and statement: $\pi = \mathrm{PK}\{(w) : (w, x) \in \mathcal{R}\}$.

$\mathsf{Verify}_{\mathcal{R}}(x, \pi) \mapsto \{True, False\}$**:** The verifier is given the proof π and x with which they determine whether the prover knows a secret w such that $(w, x) \in \mathcal{R}$ holds.

3 Overview

In this section we provide a high-level overview of our WabiSabi protocol.

3.1 Phases

A CoinJoin round consists of an Input Registration, an Output Registration and a Transaction Signing phase. To defend against Denial of Service attacks it is important to ensure the inputs of users who do not comply with the protocol are identified so these inputs can be excluded from the following rounds in order to ensure completion of the protocol.

1. While identifying non-compliant inputs during Input Registration phase is trivial, there is no reason to issue penalties at this point.
2. Identifying non-compliant inputs during Output Registration phase is not possible, thus this phase always completes and progresses to the Signing phase.
3. During Signing phase, inputs which are not signed are non-compliant inputs and they shall be issued penalties.

The cryptography in WabiSabi ensures honest participants always agree to sign the final CoinJoin transaction if the coordinator is honest. Anonymous credentials allow the coordinator to verify that amounts of each user's output registrations are funded by input registrations without learning specific relationships between inputs and outputs.

3.2 Credentials

The coordinator issues anonymous credentials which authenticate attributes in response to registration requests. We use keyed-verification anonymous credentials (introduced in [CMZ14]), in particular the scheme from [CPZ19] which supports group attributes (attributes whose value is an element of the underlying group $\mathbb{G}$). A user can then prove possession of a credential in zero knowledge in a subsequent registration request, without the coordinator being able to link it to the registration from which it originates.

In order to facilitate construction of a CoinJoin transaction while protecting the privacy of participants, we instantiate the scheme with a single group attribute M_a which encodes a confidential Bitcoin amount as a Pedersen commitment. These commitments are never opened. Instead, properties of the values they commit to are proven in zero knowledge, allowing the coordinator to validate requests made by honest participants. In ideal circumstances the coordinator would not learn anything beyond what can be learned from the resulting CoinJoin transaction but despite the unlinkability of the credentials timing of requests or connectivity issues may still reveal information about links.

3.3 Registration

To aid intuition we first describe a pair of protocols, where credentials are issued during input registration, and then then presented at output registration. k denotes the number of credentials used in registration requests, and $a_{max} = 2^{51} - 1$ constrains the range of amount values[11]. For better privacy and efficiency these are then generalized into a unified protocol used for both input and output registration, where every registration involves both presentation and issuance of credentials. This protocol is described in detail in section 4.

[11] $\log_2(2099999997690000) \approx 50.9$

In order to maintain privacy clients must isolate registration requests using unique network identities. A single network identity must not expose more than one input or output, or more than one set of requested or presented credentials.

For fault tolerance, request handling should be idempotent, allowing a client to retry a failed request without modification using a fresh network identity or one which was previously used to attempt that request.

3.3.1 Input Registration

1. The user sends k credential requests with accompanying range and sum proofs to the coordinator: $((M_{a_i}, \pi_i^{range})_{i=1}^k, \pi^{sum}, a_{in})$.
2. The coordinator verifies the received proofs. If they are not verified it aborts the protocol, otherwise it issues k MACs on the requested attributes $(\mathsf{MAC}_{\mathsf{sk}}(M_{a_i}), \pi_i^{\text{iparams}})_{i=1}^k$.

Figure 4: Input Registration protocol

The user submits an input of amount a_{in} along with k group attributes, (M_{a_i}). She proves in zero knowledge that the sum of the requested sub-amounts is equal to a_{in} and that the individual amounts are positive integers in the allowed range.

The coordinator verifies the proofs, and issues k MACs on the requested attributes, along with a proof of correct generation of the MAC, as in *Credential Issuance* protocol of [CPZ19].

3.3.2 Output Registration

1. The user sends k randomized commitments, a proof of a valid MAC for the corresponding non-randomized commitments, serial numbers with a proof of their validity, and finally a proof of the sum of the amounts: $((C_{a_i}, \pi_i^{MAC}, S_i, \pi_i^{serial})_{i=1}^k, \pi^{sum}, a_{out})$.
2. The coordinator verifies proofs and registers requested output iff. all proofs are valid and the serial numbers have not been used before.

Figure 5: Output Registration protocol

To register her output the user randomizes the attributes and generates a proof of knowledge of k valid credentials issued by the coordinator.

Additionally, she proves the serial number is valid. These serial numbers are required for double spending protection, and must be correspond but unlinkable to a specific M_a.

Finally, she proves that the sum of her randomized amount attributes C_a matches the requested output amount a_{out}, analogously to input registration.[12]

She submits these proofs, the randomized attributes, and the serial numbers. The coordinator verifies the proofs, and if accepted the output will be included in the transaction.

3.3.3 Unified Registration

In order to increase flexibility in a dynamic setting, where a user may not yet know her desired output allocations during input registration, and to allow setting a small[13] value of k as a protocol level constant to reduce privacy leaks, we can generalize input and output registration into a single unified protocol for use in both phases, which also supports reissuance. For complete definitions see section 4.

The user submits k valid credentials and k credential requests, where the sums of the underlying amount commitments must be balanced (fig. 6).

[12] Note that there is no need for range proofs, since amounts have been previously validated.

[13] Specifically, $2 \leq k \leq 10 \approx \log_2\left(\frac{\texttt{MAX\_STANDARD\_TX\_WEIGHT}-58}{274+124}\right)$ the maximum number of participants, because although $k = 1$ suffices for flexibility it limits parallelism, leaking privacy by temporal fingerprinting. The limit on participant count is because 274 and 124 are the minimum weight units required for a participant with only a single input and output, and 58 is the shared per transaction overhead.

1. During both input and output registration phases the user submits:
   - k credential requests with accompanying range and sum proofs to the coordinator: $(M_{a_i}, \pi_i^{range})_{i=1}^k$
   - k randomized commitments, proofs of valid credentials issued for the corresponding non-randomized commitments, serial numbers, and proofs of their validity: $(C_{a_i}, \pi_i^{MAC}, S_i, \pi_i^{serial})_{i=1}^k$
   - A balance Δ_a and a proof of its correctness π^{sum}
   - If $\Delta_a \neq 0$, an input or output with value $|\Delta_a|$.
2. The coordinator verifies the received proofs, and that the serial numbers have not been used before, and depending on the current phase, $\Delta_a \geq 0$ (input) or $\Delta_a \leq 0$ (output). If it accepts, it issues k MACs on the requested attributes $(\mathsf{MAC}_{\mathsf{sk}}(M_{a_i}), \pi_i^{\text{iparams}})_{i=1}^k$, and if $\Delta_a \neq 0$, registers the input or output with value $|\Delta_a|$.

Figure 6: Unified Registration protocol

1. During input registration phase the user submits k credential requests: $(M_{a_i}, \pi_i^{null})_{i=1}^k$
2. The coordinator verifies the received proofs. If it accepts, it issues k MACs on the requested attributes $(\mathsf{MAC}_{\mathsf{sk}}(M_{a_i}), \pi_i^{\text{iparams}})_{i=1}^k$.

Figure 7: Credential bootstrapping protocol

To prevent the coordinator from being able to distinguish between initial vs. subsequent input registration requests (which may merge amounts) credential presentation should be mandatory. Initial credentials can be obtained with an auxiliary bootstrapping operation (fig. 7).

3.4 Signing phase

The user fetches the finalized but unsigned transaction from the coordinator. If it contains the outputs she registered she will sign her inputs and submit each signature separately using the network identity used for that input's registration.

3.5 Examples

To illustrate the above protocols, figs. 8a and 8b show how a user might register inputs and outputs when credentials are only presented during the output registration phase and figs. 8c and 8d show the unified protocol, when credentials are both presented and requested in every registration request.

Registration requests are depicted as vertices labeled with Δ_a, a double stroke denoting output registrations. A credential is an edge from the registration in which it was requested to the registration where it was presented, also labeled with the amount. The vertex's label must be equal to the balance of the labels of its incoming edges and its outgoing edges. Note that edges and their labels are only known to the owners of the credentials. For simplicity we omit credentials with zero value.

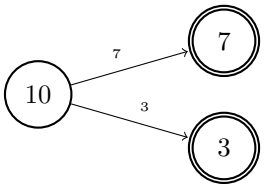
4 WabiSabi Credentials

In this section we provide the details of the unified protocol (section 3.3.3) and its use of the KVAC scheme introduced in [CPZ19]. Following that work our protocol is defined over an Abelian group $\mathbb{G}$ of prime order q, written in multiplicative notation. $\mathsf{HashTo}\mathbb{G} : \{0,1\}^* \mapsto \mathbb{G}$ is a function from strings to group elements, based on a cryptographic hash function[FT12].

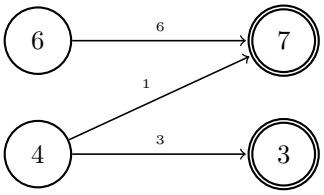
We require the following fixed set of group elements for use as generators with different purposes:

$$\underbrace{G_w, G_{w'}, G_{x_0}, G_{x_1}, G_V}_{\text{MAC and Show}} \qquad \underbrace{G_a}_{\text{attributes}} \qquad \underbrace{G_g, G_h}_{\text{commitments}} \qquad \underbrace{G_s}_{\text{serial numbers}}$$

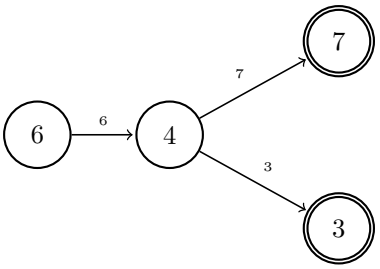
chosen so that nobody knows the discrete logarithms between any pair of them, e.g. $G_h = \mathsf{HashTo}\mathbb{G}(\text{“h”})$.



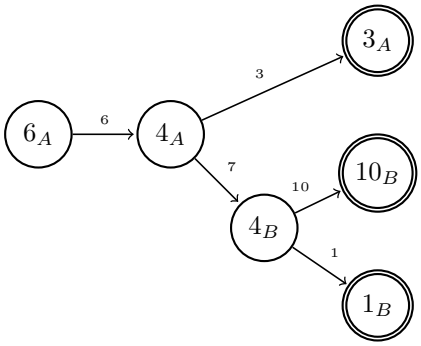
(a) Alice wants to spend an input of amount 10 and create two outputs with amounts 7 and 3 (e.g. a payment and change)



(b) Alice wants to combine her inputs of amounts 6 and 4 and register two outputs as in fig. 8a.



(c) Alice wants to spend two inputs and register two outputs using the unified protocol, which allows her to present the credential from her first input registration when registering her second input to combine the amounts.



(d) Alice wants to pay Bob, who is also participating in the protocol. Alice combines her amounts at input registration and reveals a credential corresponding to the payment amount to Bob. Bob presents this credential in his own input registration. Alice registers a change output, and Bob registers two outputs. Only Alice knows the details of the request in which the credential labeled 7 was issued and only Bob knows where it was presented, but both know the amount.

Figure 8: Credential usage examples

Our notation deviates slightly from [CPZ19], in that we subscript the attribute generators G_{y_i} as G_a instead of using numerical indices, and we require two additional generators G_g and G_h for constructing the attribute M_a as a Pedersen commitment.

As with the generator names, we modify the names of the attribute related components of the secret key $\mathrm{sk} = (w, w', x_0, x_1, y_a) \in_R \mathbb{Z}_q{}^5$ according to our fixed set of group attributes.

The coordinator parameters $iparams = (C_W, I)$ are computed as:

$$C_W = {G_w}^w {G_{w'}}^{w'} \quad I = \frac{G_V}{{G_{x_0}}^{x_0} {G_{x_1}}^{x_1} {G_a}^{y_a}}$$

and published as part of the round metadata and are used by the coordinator to prove correctness of issued MACs, and by the users to prove knowledge of a valid MAC.

4.1 Credential Requests

For each $i \in [1, k]$ the user chooses an amount $a_i \mid 0 \leq a_i < a_{max}$ subject to the constraints of the balance proof (section 4.5). She commits to the amount with randomness $r_i \in_R \mathbb{Z}_q$, and these commitments are the attributes of the requested credentials:

$$M_{a_i} = {G_h}^{r_i} {G_g}^{a_i}$$

For each amount a_i she also computes a range proof which ensures there are no negative values:

$$\pi_i^{range} = \mathrm{PK}\left\{(a_i, r_i) : M_{a_i} = {G_h}^{r_i} {G_g}^{a_i} \wedge 0 \leq a_i < a_{max}\right\}$$

In credential bootstrap requests the range proofs can be replaced with simpler proofs of $a_i = 0$:

$$\pi_i^{null} = \mathrm{PK}\left\{(r_i) : M_{a_i} = {G_h}^{r_i}\right\}$$

We note that if Bulletproofs [BBB+18] are utilized for the range proofs π_i^{range} a combined proof will significantly decrease the communication overhead and that some implementations perform the π^{null} optimization already.

4.2 Credential Issuance

If the coordinator accepts the requests (see sections 4.3 to 4.5), it registers the input or output if one is provided, and for each $i \in [1, k]$ it issues a credential by responding with $(t_i, V_i) \in \mathbb{Z}_q \times \mathbb{G}$, which is the output of $\mathsf{MAC}_{\mathsf{sk}}(M_{a_i})$, where:

$$t_i \in_R \mathbb{Z}_q \qquad U_i = \mathsf{HashTo}\mathbb{G}(t_i) \qquad V_i = {G_w}^{w} {U_i}^{x_0 + x_1 t_i} {M_{a_i}}^{y_a}$$

To rule out tagging of individual users the coordinator must prove knowledge of the secret key, and that (t_i, U_i, V_i) are correct relative to $iparams = (C_W, I)$:

$$\begin{aligned} \pi_i^{iparams} = \mathrm{PK}\{(w, w', x_0, x_1, y_a) : \\ C_W = {G_w}^{w} {G_{w'}}^{w'} \wedge \\ I = \frac{G_V}{{G_{x_0}}^{x_0} {G_{x_1}}^{x_1} {G_a}^{y_a}} \wedge \\ V_i = {G_w}^{w} {U_i}^{x_0 + x_1 t_i} {M_{a_i}}^{y_a}\} \end{aligned}$$

4.3 Credential Presentation

The user chooses k unused credentials issued in prior registration requests, i.e. valid MACs $(t_i, V_i)_{i=1}^k$ on attributes $(M_{a_i})_{i=1}^k$.

For each credential $i \in [1, k]$ she executes the Show protocol described in [CPZ19]:

1. She chooses $z_i \in_R \mathbb{Z}_q$, and computes $z_{0_i} = -t_i z_i (\mathrm{mod} q)$ and the randomized commitments:

$$\begin{aligned} C_{a_i} &= {G_a}^{z_i} M_{a_i} \\ C_{x_{0_i}} &= {G_{x_0}}^{z_i} U_i \\ C_{x_{1_i}} &= {G_{x_1}}^{z_i} {U_i}^{t_i} \\ C_{V_i} &= {G_V}^{z_i} V_i \end{aligned}$$

2. To prove to the coordinator that a credential is valid she computes a proof:

$$\begin{aligned} \pi_i^{MAC} = \mathrm{PK}\{(z_i, z_{0_i}, t_i) : \\ Z_i = I^{z_i} \wedge \\ C_{x_{1_i}} = {C_{x_{0_i}}}^{t_i} {G_{x_0}}^{z_{0_i}} {G_{x_1}}^{z_i}\} \end{aligned}$$

   which implies the following without allowing the coordinator to link π_i^{MAC} to the underlying attributes (M_{a_i}):

$$\mathsf{Verify}((C_{x_{0_i}}, C_{x_{1_i}}, C_{V_i}, C_{a_i}, Z_i), \pi_i^{MAC}) \iff \mathsf{VerifyMAC}_{\mathsf{sk}}(M_{a_i})$$

3. She sends $(C_{x_{0_i}}, C_{x_{1_i}}, C_{V_i}, C_{a_i}, \pi_i^{MAC})$ and the coordinator computes:

$$Z_i = \frac{C_{V_i}}{{G_w}^{w} {C_{x_{0_i}}}^{x_0} {C_{x_{1_i}}}^{x_1} {C_{a_i}}^{y_a}}$$

   using its secret key (independently of the user's derivation), and verifies π_i^{MAC}.

4.4 Double-spending prevention using serial numbers

The user proves that the group element $S_i = {G_s}^{r_i}$, which is used as a serial number, was generated correctly with respect to C_{a_i}:

$$\pi_i^{serial} = \mathrm{PK}\{(z_i, a_i, r_i) : S_i = {G_s}^{r_i} \wedge C_{a_i} = {G_a}^{z_i} {G_h}^{r_i} {G_g}^{a_i}\}$$

The coordinator verifies π_i^{serial} and checks that the S_i has not been used before (allowing for idempotent registration).

Note that since the logical conjunction of π_i^{serial} and π_i^{MAC} is required for each credential, and because these proofs share both public and private inputs it is appropriate to use a single proof for both statements.

4.5 Over-spending prevention by balance proof

The user needs to convince the coordinator that the total amounts redeemed and the requested differ by the public input Δ_a, which she can prove by including the following proof of knowledge:

$$\pi^{sum} = \mathrm{PK}(\{(z, \Delta_r) : B = {G_a}^z {G_h}^{\Delta_r}\})$$

where

$$B = {G_g}^{\Delta_a} \prod_{i=1}^{k} \frac{C_{a_i}}{M'_{a_i}} \qquad z = \sum_{i=1}^{k} z_i \qquad \Delta_r = \sum_{i=1}^{k} r_i - r'_i$$

with r'_i denoting the randomness terms in the $(M'_{a_i})_{i=1}^k$ attributes of the credentials being requested and z_i, r_i denoting the ones in the randomized attributes $(C_{a_i})_{i=1}^k$ of the credentials being presented.

During the input registration phase Δ_a may be positive, in which case an input of amount $a_{in} = \Delta_a$ must be registered with proof of ownership. During the output registration phase Δ_a may be negative, in which case an output of amount $a_{out} = -\Delta_a$ is registered. If $\Delta_a = 0$ credentials are simply reissued, with no input or output registration occurring.

4.6 Unconditional Hiding

Note that S_i is not perfectly hiding because there is exactly one $r_i \in \mathbb{Z}_q$ such that $S_i = {G_s}^{r_i}$. Similarly, randomization by z_i only protects unlinkability of issuance and presentation against a computationally bounded adversary. Null credentials have the same issue, since the amount exponent is known to be zero.

To unconditionally preserve user privacy in the event that the hardness assumption of the discrete logarithm problem in $\mathbb{G}$ is broken we can add an additional randomness term r'_i used with an additional generator G'_h to the amount commitments M_{a_i}, and similarly another randomness term z'_i and generators $G'_a, G'_{x_0}, G'_{x_1}, G'_V$ in order to obtain unconditional unlinkability for the commitments.[14]

5 Security and Privacy

In this section, we discuss the security and privacy guarantees of the WabiSabi credential scheme for construction of CoinJoin transactions. Theft concerns are addressed through Bitcoin's security model, making WabiSabi trustless in that regard. Since most CoinJoins are an overt technique privacy strongly depends on the structure of the transactions themselves. WabiSabi is designed as a general purpose mechanism, so those details are outside of the scope of this work.

The goal of the protocol is to allow a coordinator to provide the service to honest participants, without learning anything about the mapping between registered input and outputs, apart from what is already deducible given the public amounts visible on the Bitcoin blockchain. WabiSabi leverages the unlinkability of anonymous credentials and the hiding property of the amount commitments to minimize privacy leaks when a set of participants utilizes a centralized coordinator to reach agreement about such a transaction.

5.1 Availability

5.1.1 Malicious Coordinator

Being a central point of failure, the coordinator is a trusted party with regards to availability. If competing coordinators charge fees for their services then this is a minimal assumption in practice given the financial incentive.

A malicious coordinator can fully disrupt the protocol by censoring certain inputs either at input registration or during the signing phase. Such denial of service can amplify attacks on privacy, by partitioning users in order to perform set intersection attacks. A malicious coordinator can drop messages causing any user to appear to be non-compliant, and therefore disrupt the protocol arbitrarily, and always learns the requested inputs and outputs, even if a round fails.

5.1.2 Malicious Users

Signatures can only be made after a transaction has been negotiated, and all inputs must provide a valid signature. Consequentially users can always disrupt the protocol during the final phase. Failure to sign is attributable to specific inputs and therefore can be mitigated by the coordinator. This allows the remaining

[14] Assuming the coordinator is not able to attack network level privacy and proofs of knowledge are unconditionally hiding.

honest participants to restart the protocol. Denial of service is not costless because unspent transaction outputs are a limited resource.

A malicious participant still learns all of the input and output registrations[15] including amounts and scripts, so denial of service also presents privacy concerns.

5.2 Unlinkability

A mixing scheme should ensure that any links between registered inputs and outputs are only known to their owners. The unlinkability property in [CPZ19] ensures that presentation of credentials cannot be linked back to their issuance. Since every registration request is only associated with a single input or output, only information that would need be broadcast publicly on the blockchain if the protocol succeeds is revealed directly.

However, there are other means participants or the coordinator could reduce or potentially eliminate the achieved privacy guarantees.

5.2.1 Passive attacks

As in section 3.5 we can consider registration requests as vertices labeled by Δ_a of a graph where credentials are edges labeled by the amount from issuance to presentation. In the unified protocol this is a directed acyclic graph with indegree and outdegree k. The unlinkabilility of credentials and hiding property of the amount commitments obscure the edges and their labels. The serial number and balance proofs enforce a global invariant where the sums of the inwards and outwards edges of every vertex differ by its label.

The coordinator observes registration requests according to a partial order which is an extension of the partial order defined by the DAG, in other words the coordinator knows that parallel requests do not share an edge and that dependent requests must be made sequentially. Clients can mitigate these leaks by minimizing dependencies between requests, scheduling requests randomly during phases of a known duration, and making additional reissuance requests.

5.2.2 Active attacks

Sybil attacks [Dou02] are inherent to mixing schemes, because a transaction between n apparent participants $n-1$ of which are controlled by an attacker will fully link the victim's coins on both sides of the CoinJoin while giving the impression that the victim's privacy has been improved. There is a liquidity requirement for such an attack since participants must provide valid inputs[16], as well as a cost imposed by mining fees.

An attacker attempting to Sybil attack all CoinJoins would need to control some multiple of the combined CoinJoin volume contributed by honest participants, and to successfully partition honest participants to a sufficient degree. In the centrally coordinated setting, fees paid by users can arbitrarily increase the cost of Sybil attacks by other users. However, this does not protect against a malicious coordinator which is only bound by liquidity and mining fees. Furthermore, service fees paid by honest participants may reduce the cost of such an attack or even make it profitable.

A malicious coordinator could also delay processing of requests in order to gain more through timing and ordering leaks. In the worst case the coordinator can attempt to linearize all requests by blocking and unblocking individual users to recover full set of labeled edges. This is possible when $k = 1$ and users have minimal dependencies between their requests and tolerate arbitrary timeouts but issue requests in a timely manner.

A malicious coordinator may also tag users by providing them with different issuer parameters. When registering inputs a proof of ownership must be provided. If signatures are used, by covering the issuer parameters and a unique round identifier these proofs allow other participants to verify that everyone was given the same parameters.

6 Related Work

The Bitcoin privacy-enhancing literature is extensive, with notable published works including: [BNM+14; BOL+14; RMK14; VR15; ZGH+15; RMK17; MNF17; HAB+17]. The deployed solutions so far have been mostly CoinJoin based [Max13a], with various limitations and relying on some simplifying assumptions. This leaves a gap between the privacy technology available to Bitcoin users in the real world and the stronger decentralization or trustlessness properties of schemes that remain unused. We believe that the lack of

[15] If `SIGHASH_ANYONECANPAY` is set in the signature flags the full set of inputs could be kept known only to the coordinator until all signatures have been provided.

[16] See also JoinMarket fidelity bonds: `https://gist.github.com/chris-belcher/18ea0e6acdb885a2bfbdee43dcd6b5af`

practical use and deployment of most of the Bitcoin privacy-enhancing literature is due to shortcomings in one or more of these aspects.

Denomination Some of the proposed and deployed schemes only support fixed amounts. Support for variable amounts typically add complexity, reduces privacy, or both.

Performance Fully decentralized schemes, CoinJoin based or otherwise, typically have higher communication complexity and therefore support fewer participants compared to centralized ones. CoinJoin-based mixing protocols provide mixing in a single Bitcoin transaction. Non-CoinJoin schemes often require more block space, for instance Xim [BOL+14] requires 7 on-chain transactions, TumbleBit [HAB+17] 4 transactions, Blindcoin [VR15] and Mixcoin [BNM+14] both require 2 transactions. Long sequence of on-chain transactions result in longer delays for users.

Infrastructure Several mixing-protocols assume and rely on infrastructure which is not practically available. For example, CoinShuffle assumes a peer-to-peer public bulletin board.

Trustlessness Some protocols have stronger trust assumptions with respect to theft, for example Mixcoin and Blindcoin[17], whereas other protocols utilize Bitcoin's scripting capabilities to secure user funds.

Since Chaumian CoinJoins have been successfully deployed despite some limitations (see section 1.2), it appears that centrally coordinated CoinJoins strike a desirable balance in the Bitcoin privacy design space. It is non-custodial, has low messaging complexity, and results in a single transaction minimizing delays and network fees.

In tables 1 and 2, n denotes the number of participants. Overt transactions are apparent and fingerprintable on the blockchain, but still provide privacy within the transaction. Disjoint transactions do not link individual users' inputs and outputs on the blockchain. Covert transactions are not fingerprintable as privacy enhancing transactions.

				Privacy against	
	Messages	Denominations	Coordination	Server	Participants
Knapsack [MNF17]	-	variable	-	-	-[a]
CoinShuffle [RMK14]	$\mathcal{O}(n^2)$	fixed	decentralized	n.a.	✓[b]
CoinShuffle++ [RMK17]	$\mathcal{O}(n)$	fixed	decentralized	n.a.	✓[b]
CashFusion	$\mathcal{O}(n)$	variable	centralized	✓[c]	✓[d]
JoinMarket	$\mathcal{O}(n)$	variable	by taker[e]	n.a.[e]	✓
Chaumian CoinJoin	$\mathcal{O}(n)$	fixed	centralized	✓	✓
This Work	$\mathcal{O}(n)$	variable	centralized	✓	✓

[a] Transaction structure guarantees minimal ambiguity in sub-transaction assignments
[b] Requires only 2 honest participants.
[c] Assuming server does not participate as a verifier.
[d] Transaction ambiguity guarantees are probablistic, with additional computational difficulty under assumptions.
[e] Transactions are initiated and coordinated by users buying mixing offers on a decentralized markeplace.

Table 1: CoinJoin based protocols, no theft risk and requiring a single overt transaction.

Our application of [CPZ19] has similarities to Danake [Val20], another recent application of KVACs which has additional considerations for longer lived tokens. It uses the credentials of [CMZ14], hiding the amounts using blind issuance with ElGamal encrypted attribute values.

7 Conclusion

In this work, we introduced WabiSabi, an application of the keyed-verification anonymous credentials scheme proposed in [CPZ19] to Bitcoin mixing. WabiSabi builds on Chaumian CoinJoins, adding support for arbitrarily variable amounts. The goal of this extension of previous work is to make CoinJoins more practical by enabling new use cases and removing existing restrictions of deployed solutions. We note that the credential scheme can be applied in centrally coordinated settings which are not necessarily based on CoinJoins for a wider range of theft and availability threat models.

[17] Both protocols have provisions for accountability of the mixes, mitigating theft concerns

						Privacy against		
	Msg.	Txn.	Denom.	Theft	Coord.	Srv.	Part.	Blockchain
Custodial	-	-	-	✗	cent.	TTP	TTP	-
Mixcoin [BNM+14]	$2n^a$	$2n^a$	var.[b]	TTP[c]	cent.	TTP	TTP	disjoint
Blindcoin [VR15]	$4n$	$2n$	fix.	TTP[c]	cent.	✓	✓	disjoint
TumbleBit [HAB+17]	$12n$	$4n$	fix.	✓	cent.	✓	✓	overt[d]
CoinSwap[Max13b]	$\mathcal{O}(1)^e$	4^e	var.	✓	p2p	n.a.	✗	disjoint
Xim [BOL+14]	n.a.	7^e	var.	✓	p2p[f]	n.a.	✗	disjoint
CoinParty [ZGH+15]	$?\mathcal{O}(n \log n)?$	2n	fix.	✓[g]	dec.	n.a.	✓[h]	covert
CoinSwap[Bel20]	$\mathcal{O}(n)$	$\mathcal{O}(n)$	var.	✓	taker	n.a.	✓	covert

[a] Per mixnet stage
[b] Fixed chunk size recommended for privacy, but technically not required.
[c] Provides server accountability
[d] TumbleBit provides k-anonymity per multi-transaction epoch.
[e] $n = 2$
[f] Advertisements are made on the blockchain
[g] Assuming $\frac{1}{3}$ of participants are honest.
[h] Assuming at least 2 participants are honest.

Table 2: Non-CoinJoin based protocols.

8 Acknowledgements

We would like to acknowledge the inputs and invaluable contributions of ZmnSCPxj, Yahia Chiheb, Thaddeus Dryja, Adam Gibson, Dan Gould, Ethan Heilman, Max Hillebrand, Aviv Milner, Jonas Nick, Lucas Ontivero, Tim Ruffing, Ruben Somsen and Greg Zaverucha to this paper.

References

[AKR+13] Elli Androulaki, Ghassan O. Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. "Evaluating User Privacy in Bitcoin." In: *Financial Cryptography and Data Security*. Ed. by Ahmad-Reza Sadeghi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 34–51. ISBN: 978-3-642-39884-1. DOI: 10.1007/978-3-642-39884-1_4.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. "Bulletproofs: Short proofs for confidential transactions and more." In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2018, pp. 315–334. DOI: 10.1109/SP.2018.00020.

[Bel20] Chris Belcher. *Design for a CoinSwap Implementation for Massively Improving Bitcoin Privacy and Fungibility*. 2020. URL: https://gist.github.com/chris-belcher/9144bd57a91c194e332fb5ca371d096 (visited on 07/01/2020).

[BNM+14] Joseph Bonneau, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A Kroll, and Edward W Felten. "Mixcoin: Anonymity for bitcoin with accountable mixes." In: *International Conference on Financial Cryptography and Data Security*. Springer. 2014, pp. 486–504.

[BOL+14] George Bissias, A Pinar Ozisik, Brian N Levine, and Marc Liberatore. "Sybil-resistant mixing for bitcoin." In: *Proceedings of the 13th Workshop on Privacy in the Electronic Society*. 2014, pp. 149–158. DOI: 10.1145/2665943.2665955.

[Cha83] David Chaum. "Blind signatures for untraceable payments." In: *Advances in cryptology*. Springer. 1983, pp. 199–203. DOI: 10.1007/978-1-4757-0602-4_18.

[CMZ14] Melissa Chase, Sarah Meiklejohn, and Greg Zaverucha. "Algebraic MACs and Keyed-Verification Anonymous Credentials." In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. CCS '14. Scottsdale, Arizona, USA: Association for Computing Machinery, 2014, pp. 1205–1216. ISBN: 9781450329576. DOI: 10.1145/2660267.2660328. URL: https://eprint.iacr.org/2013/516.

[CPZ19] Melissa Chase, Trevor Perrin, and Greg Zaverucha. *The Signal Private Group System and Anonymous Credentials Supporting Efficient Verifiable Encryption*. Cryptology ePrint Archive, Report 2019/1416. 2019. URL: https://eprint.iacr.org/2019/1416.

[CS97] Jan Camenisch and Markus Stadler. "Proof systems for general statements about discrete logarithms." In: *Technical Report/ETH Zurich, Department of Computer Science* 260 (1997). DOI: 10.3929/ethz-a-006651937.

[DFT+97] George Davida, Yair Frankel, Yiannis Tsiounis, and Moti Yung. "Anonymity control in e-cash systems." In: *International Conference on Financial Cryptography*. Springer. 1997, pp. 1–16. DOI: 10.1007/3-540-63594-7_63.

[DM06] Roger Dingledine and Nick Mathewson. "Anonymity Loves Company: Usability and the Network Effect." In: *WEIS*. 2006. URL: https://www.freehaven.net/anonbib/cache/oreilly-usability.pdf (visited on 07/01/2020).

[Dou02] John R. Douceur. "The Sybil Attack." In: *Peer-to-Peer Systems*. Ed. by Peter Druschel, Frans Kaashoek, and Antony Rowstron. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 251–260. ISBN: 978-3-540-45748-0. DOI: 10.1007/3-540-45748-8_24.

[FT12] Pierre-Alain Fouque and Mehdi Tibouchi. "Indifferentiable Hashing to Barreto–Naehrig Curves." In: *Progress in Cryptology – LATINCRYPT 2012*. Ed. by Alejandro Hevia and Gregory Neven. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 1–17. ISBN: 978-3-642-33481-8. DOI: 10.1007/978-3-642-33481-8_1.

[FT17] Adam Ficsor and TDevD. *ZeroLink: The Bitcoin Fungibility Framework*. 2017. URL: https://github.com/nopara73/ZeroLink (visited on 05/01/2020).

[HAB+17] Ethan Heilman, Leen Alshenibr, Foteini Baldimtsi, Alessandra Scafuro, and Sharon Goldberg. "Tumblebit: An untrusted bitcoin-compatible anonymous payment hub." In: *Network and Distributed System Security Symposium*. 2017. DOI: 10.14722/ndss.2017.23086.

[HF16] Martin Harrigan and Christoph Fretter. "The unreasonable effectiveness of address clustering." In: *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE. 2016, pp. 368–373. DOI: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0071.

[Max13a] Greg Maxwell. *CoinJoin: Bitcoin privacy for the real world*. 2013. URL: https://bitcointalk.org/index.php?topic=279249.0 (visited on 05/01/2020).

[Max13b] Greg Maxwell. *CoinSwap: Transaction graph disjoint trustless trading*. 2013. URL: https://bitcointalk.org/index.php?topic=321228.0 (visited on 07/01/2020).

[Max16] Greg Maxwell. *Confidential Transactions*. 2016. URL: https://web.archive.org/web/20200502151159/https://people.xiph.org/~greg/confidential_values.txt (visited on 05/16/2020).

[MBB13] Malte Möser, Rainer Böhme, and Dominic Breuker. "An inquiry into money laundering tools in the Bitcoin ecosystem." In: *2013 APWG eCrime Researchers Summit*. Ieee. 2013, pp. 1–14.

[Miz13] Alex Mizrahi. *coin mixing using Chaum's blind signatures*. 2013. URL: https://bitcointalk.org/index.php?topic=150681.0 (visited on 05/01/2020).

[MNF17] Felix Konstantin Maurer, Till Neudecker, and Martin Florian. "Anonymous CoinJoin transactions with arbitrary values." In: *2017 IEEE Trustcom/BigDataSE/ICESS*. IEEE. 2017, pp. 522–529. DOI: 10.1109/Trustcom/BigDataSE/ICESS.2017.280.

[MPJ+13] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. "A Fistful of Bitcoins: Characterizing Payments among Men with No Names." In: *Proceedings of the 2013 Conference on Internet Measurement Conference*. IMC '13. Barcelona, Spain: Association for Computing Machinery, 2013, pp. 127–140. ISBN: 9781450319539. DOI: 10.1145/2504730.2504747.

[MT15] Laurent MT. *Bitcoin Transactions & Privacy*. (see also https://code.samourai.io/oxt/boltzmann/). 2015. URL: https://gist.github.com/LaurentMT/d361bca6dc52868573a2 (visited on 07/01/2020).

[OKH13] Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. "Structure and anonymity of the bitcoin transaction graph." In: *Future internet* 5.2 (2013), pp. 237–250. DOI: 10.3390/fi5020237.

[RH13] Fergal Reid and Martin Harrigan. "An analysis of anonymity in the bitcoin system." In: *Security and privacy in social networks*. Springer, 2013, pp. 197–223. DOI: 10.1007/978-1-4614-4139-7_10.

[RMK14] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. "Coinshuffle: Practical decentralized coin mixing for bitcoin." In: *European Symposium on Research in Computer Security*. Springer. 2014, pp. 345–364. DOI: 10.1007/978-3-319-11212-1_20.

[RMK17] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. "P2P Mixing and Unlinkable Bitcoin Transactions." In: *NDSS*. 2017, pp. 1–15. DOI: 10.14722/ndss.2017.23415.

[Rog15] Phillip Rogaway. *The Moral Character of Cryptographic Work*. Cryptology ePrint Archive, Report 2015/1162. 2015. URL: https://eprint.iacr.org/2015/1162.

[RS13] Dorit Ron and Adi Shamir. "Quantitative Analysis of the Full Bitcoin Transaction Graph." In: *Financial Cryptography and Data Security*. Ed. by Ahmad-Reza Sadeghi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 6–24. ISBN: 978-3-642-39884-1. DOI: 10.1007/978-3-642-39884-1_2.

[Val20] Henry de Valence. *Danake*. 2020. URL: https://lightweight.money (visited on 06/10/2020).

[VR15] Luke Valenta and Brendan Rowan. "Blindcoin: Blinded, accountable mixes for bitcoin." In: *International Conference on Financial Cryptography and Data Security*. Springer. 2015, pp. 112–126. DOI: 10.1007/978-3-662-48051-9_9.

[ZGH+15] Jan Henrik Ziegeldorf, Fred Grossmann, Martin Henze, Nicolas Inden, and Klaus Wehrle. "CoinParty: Secure Multi-Party Mixing of Bitcoins." In: *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*. CODASPY '15. San Antonio, Texas, USA: Association for Computing Machinery, 2015, pp. 75–86. ISBN: 9781450331913. DOI: 10.1145/2699026.2699100.