

WabiSabi: Centrally Coordinated CoinJoins with Variable Amounts

Ádám Ficsór¹, Yuval Kogman¹, Lucas Ontivero¹, and István András Seres²

¹zkSNACKs

²Eötvös Loránd University

Abstract

Bitcoin transfers value on a public ledger of transactions anyone can verify. Coin ownership is defined 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, an approach based on multiparty transactions that can introduce ambiguity and break common assumptions that underlie heuristics used for deanonymization. Existing implementations of CoinJoin have several 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 [Nak09]. 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 privately² 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 controlling supply without relying on a trusted authority, thereby 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 in comparison 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 widespread [GLL18], 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 collect and safeguard 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 particular 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.

¹The 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.

²In 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 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 in the transaction outputs. To mitigate this the coordinator provides blind signatures representing units of standard denominations in response to submitted inputs. Clients unblind and submit the output with valid signature, which allows the coordinator to verify that an output registration is authorized without being able to link it to the corresponding inputs.

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 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 signatures³. 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 SharedCoin⁴ and CashFusion⁵ style transactions and Knapsack [MNF17] mixing. Payments from CoinJoin transactions are possible which enhances sender privacy, as are payments within CoinJoin transactions, improving privacy for both sender and receiver and potentially hiding the payment amount. Both kinds of payments can reduce overall blockspace utilization by reducing the number of intermediate outputs required, especially when payments can be batched without compromising privacy. Additionally, restrictions on the consolidation of inputs can be removed and the minimum amount required can be relaxed without degrading the denial of service guarantees of blind signature based protocols.

The rest of this paper is organized as follows. We present necessary background knowledge about privacy issues of Bitcoin in section 2. In section 3 we provide some preliminaries on our applied cryptographic building blocks. We introduce our system model in section 4. In section 5 we introduce our protocol, WabiSabi, at a high-level, while in section 6 we describe in-depth the cryptographic details of WabiSabi. In section 7 we argue about the security and privacy of WabiSabi. A report on our preliminary implementation is provided in section 8. We review related work in section 9. We outline ongoing and future work in section 10. Finally, we conclude our paper in section 11.

2 Background and Motivation

In this section, we provide an introduction to Bitcoin privacy issues and proposals to enhance transaction privacy. We detail current limitations of Bitcoin privacy-enhancing tools and motivate our work.

2.1 Privacy issues and CoinJoin transactions

Bitcoin is a pseudonymous cryptocurrency [Nak09]: coins are associated with so called addresses, which encode the spending conditions of the coin (their `scriptPubKey`). Bitcoin transactions spend coins as inputs and create new coins output in place of the spent ones. The Bitcoin protocol mandates the rules of a valid transaction (no double-spending, valid signatures, etc.), which is verified by all nodes of the network. Since every transaction is recorded in a public ledger, the flow of money is traceable. Several idioms of use allow one to cluster addresses likely to be controlled by the same user. Many heuristics were devised in previous work [MPJ+13; AKR+13; RH13; RS13]. The most commonly used heuristic to cluster Bitcoin addresses

³Note that the cleartext amounts appearing in the final transaction might still link individual inputs and outputs.

⁴See: <https://github.com/sharedcoin/Sharedcoin>

⁵See: <https://github.com/cashshuffle/spec>

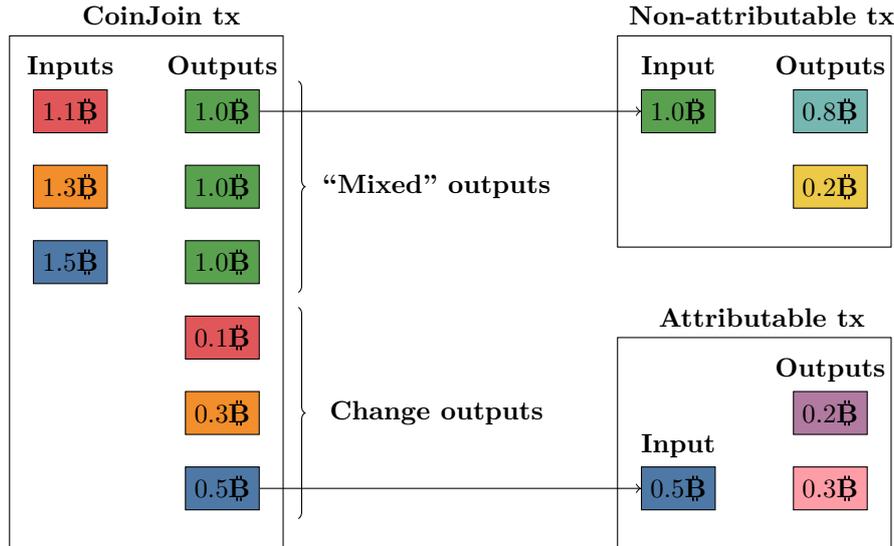


Figure 1: A simple CoinJoin transaction and two post-mix transactions. Privacy-enhanced outputs are colored green. Each of them improved their k -anonymity ($k = 3$). Change outputs have the color of their corresponding inputs. Arrows indicate which transaction output has been spent by the referencing transaction input. The payer of the first post-mix transaction is non-attributable. However, the payer of the second post-mix transaction is easily linkable to the third input of the CoinJoin transaction, since the post-mix transaction spends a change output of the CoinJoin transaction. Coins are only represented by their BTC (₿) value. For simplicity, transaction fees are omitted.

is the *common input ownership heuristic*. It states that in any Bitcoin transaction, all the inputs are likely controlled by the same user. This heuristic already facilitates a fine-grained clustering of Bitcoin addresses, with high accuracy[Nic15].

CoinJoin transactions aim to contradict to the common ownership heuristic, because multiple users collaboratively create the transaction. There are several types of privacy-enhancing techniques in Bitcoin, e.g. PayJoin⁶, CoinSwap [Bel20]. However, in this work, we solely focus on CoinJoins. In an equal-amount CoinJoin transaction (see fig. 1) each user contributes at least one input and receives at least one output with the same amount as the other users. Assuming the output types are uniform individual outputs differ only in the keys associated with them, which are indistinguishable from uniformly random data. Consequentially it is not possible to link inputs to specific mixed outputs. The privacy of such outputs can be modeled using k -anonymity [Swe02] with amounts and script types as quasi-identifiers, but this analysis becomes rather complicated in a broader context than that of a single transaction in isolation because the graph structure can reveal additional information.

2.1.1 Wasabi’s ZeroLink Protocol

Several CoinJoin protocols have been implemented and many more proposed, with varying privacy and trust models. For an overview see section 9. ZeroLink [FT17] is one such protocol implemented by Wasabi, the most popular Chaumian CoinJoin implementation for Bitcoin. Although the coordinator must be trusted to ensure availability, no trust is required with regards to safeguarding users’ funds or keeping mapping between inputs and equal amount outputs private. The protocol breaks down into three main phases:

Input Registration Users register their inputs along with proofs (digital signatures) that they can spend those inputs. Additionally, users provide their change outputs and blinded outputs to the coordinator. The coordinator blindly signs the requested blinded output.

Output Registration Users unblind their signatures received from the coordinator and register their outputs. We note that input and output registrations are completed using different network identifiers (Tor circuits) to ensure network-level privacy as well.

Signing Users receive the unsigned CoinJoin transaction from the coordinator. If it contains their requested inputs and outputs, users sign the transaction and send back to the coordinator their signature(s). If all signatures are received, the coordinator broadcasts the final, signed CoinJoin transaction.

⁶See: <https://en.bitcoin.it/wiki/PayJoin>

2.2 Limitations of Wasabi

In this work, we aim to improve on the ZeroLink protocol as implemented in Wasabi. We identify several privacy shortcomings and inefficiencies of Wasabi CoinJoins. Specifically, we are interested in quantifying block-space and other inefficiencies stemming from the imposed minimum denomination. These metrics comparing Wasabi, Samourai’s Whirlpool⁷, and other apparent CoinJoin transactions are provided. The “Other” category includes JoinMarket⁸, but also has an inherent false positive error given these transactions are identified heuristically. We collected every equal output CoinJoin-like transactions from the Bitcoin blockchain till 2020 May 22nd. Wasabi and Samourai CoinJoin transactions are trivial to detect. A transaction that has more than 10 equal amount outputs are surely Wasabi CoinJoins, while a Samourai CoinJoin transaction has the same structure, with 5 equal amount outputs of a fixed size, 3 inputs of exactly the same amount, while fees are paid by 2 inputs of a slightly higher amount which are direct descendants of a so called “tx0”, a transaction that prepares outputs for mixing and pays the coordination fees. However, other CoinJoin transactions, e.g. JoinMarket, are more difficult to identify [MB17]. Heuristically, we deem every transaction being in the “Other” CoinJoin transaction category if they simultaneously satisfy the following conditions:

- There are at least two equal amount outputs. We refer to the number of equal amount outputs as the size of the anonymity set.
- The number of distinct public keys of the inputs is at least the size of the anonymity set.
- The number of distinct public keys of the outputs is at least the size of the anonymity set.

The first requirement ensures that transactions in the “Other” category are indeed privacy-enhancing transactions. The second and third requirements filter out batching and exchange deposit/withdraw transactions. However, we acknowledge that this heuristic might incur a few false positives as well as false negatives.

A cross platform tool for the reproducible analysis of Bitcoin privacy-enhancing transactions was developed and released⁹ to facilitate this as well as further research. We report our findings in more detail.

2.2.1 Denominations

Due to the nature of blind signatures, mixed outputs of Wasabi CoinJoins are restricted to a fixed set of multiples of a base denomination¹⁰. 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. Non-mixed change outputs in a CoinJoin are almost always linkable to their corresponding inputs. Therefore, every Bitcoin privacy-enhancing tool should aim to minimize the number of unmixed change outputs.

We define *CoinJoin inefficiency* as the fraction of non-mixed change outputs in a CoinJoin transaction. In fig. 2, we observe the large amount of non-mixed outputs of essentially all Bitcoin privacy-enhancing tools. Samourai has the lowest CoinJoin inefficiency due to the homogeneous transaction structure. We note a sharp decline of CoinJoin inefficiency in Wasabi CoinJoins around January 2019 due to the introduction of multiples of the base denomination.

2.2.2 Minimum denomination

In order to participate in a CoinJoin, a user’s combined input amount must be greater or equal to the base denomination.¹¹ We observe, that considerable portion of CoinJoin inputs are less than this minimum denomination, see fig. 3.

Even when users are able to provide several smaller value inputs with a total value greater than the minimum denomination, the coordinator learns that those inputs belong to the same user. In an ideal mixing protocol, the coordinator should not obtain more information by coordinating the CoinJoin transaction than what the publicly available blockchain data reveals. This information removes many degrees of freedom when assigning non-derived sub-transactions [MNF17] or link probability [MT15], reducing intrinsic ambiguity as well as the computational cost when evaluating potential links.

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

⁷See: <https://www.samouraiwallet.com/whirlpool>

⁸See: <https://github.com/JoinMarket-Org/joinmarket>

⁹See: <https://github.com/nopara73/Dumplings>

¹⁰Approximately 0.1 B

¹¹The 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.

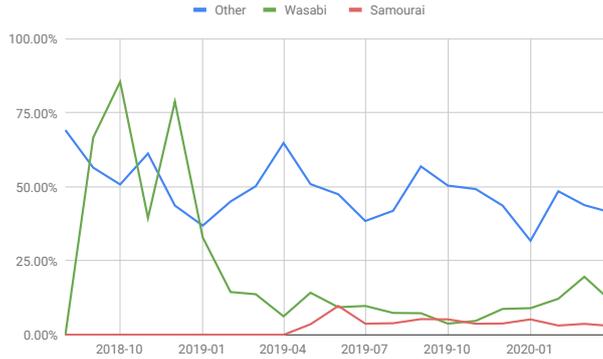


Figure 2: CoinJoin inefficiency of various privacy-focused Bitcoin wallets. We define CoinJoin inefficiency to be the fraction of non-mixed change outputs and the total number of outputs in a CoinJoin transaction. A higher value indicates more block space required to mix a given volume of Bitcoin. All CoinJoin implementations tools produce a significant amount of “toxic change” due to the requirement of mixed outputs being uniform, either directly from CoinJoin transactions or in Samurai’s case from antecedent transactions that prepare outputs for mixing.



Figure 3: Fraction of inputs with value smaller than the minimum denomination in Wasabi CoinJoin transactions.

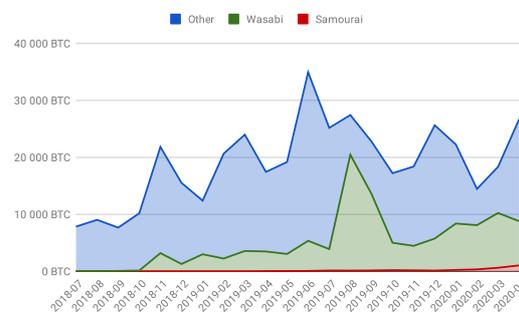


Figure 4: Volume of unmixed coins entering to various CoinJoin schemes. Wasabi is the most popular Chaumian CoinJoin implementation.

2.2.3 Varying denominations

Since users pay mining and coordination fees the base denomination is gradually reduced between rounds of consecutive CoinJoins. This makes it possible for users to mix several times without providing additional inputs. In Wasabi remixed coins are not required to pay coordination fees if they are very close to the base denomination, which 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.

2.2.4 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. 6) and create additional intermediate outputs with constrained amounts when interspersing CoinJoin transactions with transactions that send or receive value.

2.2.5 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. Payments within a CoinJoin would protect both sender and receiver privacy, as well as undermine clustering heuristics since they are analogous to PayJoins. It would also improve privacy by introducing degrees of freedom in the interpretation of CoinJoins.



Figure 5: Average remix count of various Bitcoin privacy-enhancing tools. Remixing increases user privacy at the expense of block-space efficiency.

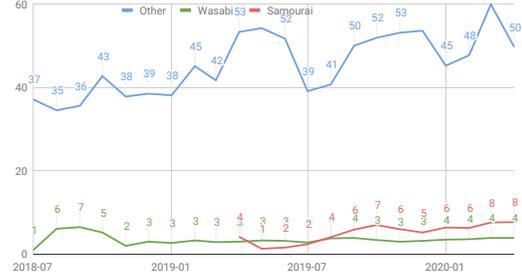


Figure 6: Average number of inputs from the first post-mix transactions in various CoinJoin schemes. The large figures for other CoinJoin-like transactions suggests false positives in our identification.

3 Preliminaries

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

3.1 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.

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

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

3.2 MAC

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

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

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

$\text{VerifyMAC}_{\text{sk}}(m, t) \mapsto \{True, False\}$: 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.

3.3 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 .

$\text{Prove}_{\mathcal{R}}(x, w) \mapsto \pi$: 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: $\pi = \text{PK}\{(w) : (x, w) \in \mathcal{R}\}$.

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

4 System model

In this section, we introduce our system model. We present the participants of our mixing protocol. Subsequently, we describe our communication and threat models along with our high-level goals.

4.1 Participants

There are four components of the WabiSabi system: the users, the coordinator, the anonymity network nodes and the Bitcoin peer-to-peer (P2P) network nodes.

User Numerous users wish to enhance their transaction privacy by collaboratively creating a CoinJoin transaction. Each of them registers at least one input and output in an unlinkable manner.

Coordinator To avoid quadratic communication-complexity in the number of users [RMK14], we apply a central, untrusted coordinator. The coordinator aids users in creating their CoinJoin transaction. The coordinator stores registered inputs and outputs. At last, it serves users with the final CoinJoin transaction for signing. Users sign the final transaction if and only if all of their registered input and outputs are contained in the transaction.

Anonymity network nodes Users need to communicate with the coordinator in an unlinkable fashion. To that end, we apply onion-routing [RSG98]. Users send messages to the coordinator using a path consisting of multiple intermediary anonymity network nodes. Each anonymity network node only knows her preceding and succeeding destination along the path. Therefore, the coordinator cannot link users to a specific request or output registration.

Bitcoin P2P network nodes Typically users do not run Bitcoin full nodes. Therefore, they rely on other full nodes to serve them necessary data from the Bitcoin blockchain. For instance, users invoke full nodes whenever they verify the inclusion of a CoinJoin transaction in the blockchain or query their own balance. Note, that for the sake of privacy, these queries needs to be made unlinkable as well.

4.2 Communication model

We assume all messages (onion-routed messages, broadcast transactions, queries about the blockchain) are delivered with a maximum delay under the bounded synchronous communication setting [AW04]. Furthermore, we assume all actors can access and read the current head of the blockchain to verify if transactions are appended to the blockchain. We remark that these are standard assumptions in the blockchain literature [BMT+17].

4.3 Threat model

We assume that the cryptographic primitives (see section 3) are secure. Specifically, we assume that the discrete logarithm is hard in the underlying group, as well as the validity of the random oracle model for hash functions. We further assume that adversaries are computationally bounded and can only corrupt at most $\frac{1}{3}$ of the consensus participants of the blockchain. We assume that users can always read the blockchain state and write to the blockchain. Also, we assume that the adversary can always read all transactions issued, while the transactions are propagating on the P2P network, and afterwards when they are written to the blockchain. We assume that the coordinator remains available throughout the entire mixing protocol. Last but not least, we presume that in case of onion-routed messages for network privacy, at least one intermediary on the source-routed path is not controlled by the adversary, i.e. users can achieve network-level privacy.

4.4 High-level goals

We state informally our desired security and privacy goals.

Availability An ideal CoinJoin protocol should remain secure and should eventually complete successfully even if certain mixing participants are malicious or off-line.

Unlinkability WabiSabi users aim to create outputs in a CoinJoin transaction, that have enhanced transaction privacy than their corresponding inputs.

Theft prevention Mixing participants should not be able to obtain more funds from the CoinJoin transaction than they are rightfully entitled to.

Performance We aim to create a high performance mixing protocol that supports computationally or bandwidth constrained users.

We remark that we consider network-level privacy as given. It can be achieved by using, for example, onion-routing [RSG98] or mix-networks [PHE+17].

5 Protocol overview

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

5.1 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 protocol ensures that if the coordinator is honest the transaction would be valid once signed and honest participants would always agree to sign the final CoinJoin transaction as it would not misallocate their funds. 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.

5.2 Credentials

The coordinator issues anonymous credentials [Bra94; Bra00; BL13] which authenticate attributes in response to registration requests. We use *keyed-verification anonymous credentials* (introduced in [CMZ14]), specifically 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.

5.3 Registration

To aid intuition we first describe a pair of protocols, where credentials are issued during input registration, and 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 values¹². 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 6.

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 (the same network identity should not be associated with more than one input or output).

¹² $\log_2(2099999997690000) \approx 50.9$

5.3.1 Input Registration

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 fall within 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].

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 $(MAC_{sk}(M_{a_i}), \pi_i^{iparams})_{i=1}^k$.

Figure 7: Input Registration protocol

5.3.2 Output Registration

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 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.¹³

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 constructed by the coordinator.

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 8: Output Registration protocol

5.3.3 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 small¹⁴ value of k as a protocol level constant (ensuring that requests are uniform) 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 6.

The user submits k valid credentials and k credential requests, where the sums of the underlying amount commitments must be balanced (fig. 9).

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. 10). A more sophisticated approach would be to prove either knowledge of a valid MAC or that the amount attribute has a value of zero, but because the input registration phase starts synchronously when enough users have indicated their intent to participate this would not actually save a round trip in practice.

¹³Note that there is no need for range proofs since amounts have been previously validated.

¹⁴Specifically, $2 \leq k \leq 10 \approx \log_2 \left(\frac{\text{MAX_STANDARD_TX_WEIGHT} - 58}{274 + 124} \right)$ 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 $(MAC_{sk}(M_{a_i}), \pi_i^{iparams})_{i=1}^k$, and if $\Delta_a \neq 0$, registers the input or output with value $|\Delta_a|$.

Figure 9: 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 $(MAC_{sk}(M_{a_i}), \pi_i^{iparams})_{i=1}^k$.

Figure 10: Credential bootstrapping protocol

5.4 Signing phase

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

5.5 Examples

To illustrate the above protocols, figs. 11a and 11b show how a user might register inputs and outputs with the simplified protocols, where credentials are only requested at input registration and presented at output registration. Figures 11c and 11d show the unified protocol, when credentials are both presented and requested in every request.

Input and output registrations are depicted as vertices labelled by $|\Delta_a|$, with 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 labelled 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.

In fig. 11a Alice first registers an input of amount 10 and requests 2 credentials with amount 7 and 3 in a single input registration request. At the output registration phase, Alice, with a different network level identity registers one output with each.

The flexibility of the credential scheme allows to achieve the same outputs originating from multiple different inputs. For instance, in fig. 11b Alice registers 2 inputs (with different network identities) of value 6 and 4. The input of value 4 is broken up to 2 credentials of value 1 and 3. At output registration Alice then combines her credentials of value 6 and 1 to be able to register an output (with yet another network identity) of value 7. Note that individual credential amounts are not known to the coordinator, which only learns their sum.

In the unified protocol at every step users must request as well as present credentials. Alice begins by obtaining bootstrap credentials (credentials with a 0 amount) in order to be able to make her first input registration. The example in fig. 11c achieves the same results as fig. 11b, but utilizes the credentials differently. At input registration Alice first obtains a credential with value 6, and then presents it in her next input registration for the input valued 4.

The most interesting application of the credential system is presented in fig. 11d. Here Alice begins by registering her inputs as in fig. 11c. Another user, Bob, also obtains bootstrap credentials, but does

not register an input of his own at this point. Alice and Bob can then communicate out of band, with Alice giving Bob the credential valued 7, which he then presents when registering his own input worth 4, effectively receiving a privacy enhanced payment of 7 from Alice. Note that in this last example Alice must ask Bob before signing during the final phase to ensure her funds were allocated as intended, and Bob must trust Alice to provide him with a valid credential, and cannot verify its validity without presenting it to the coordinator¹⁵.

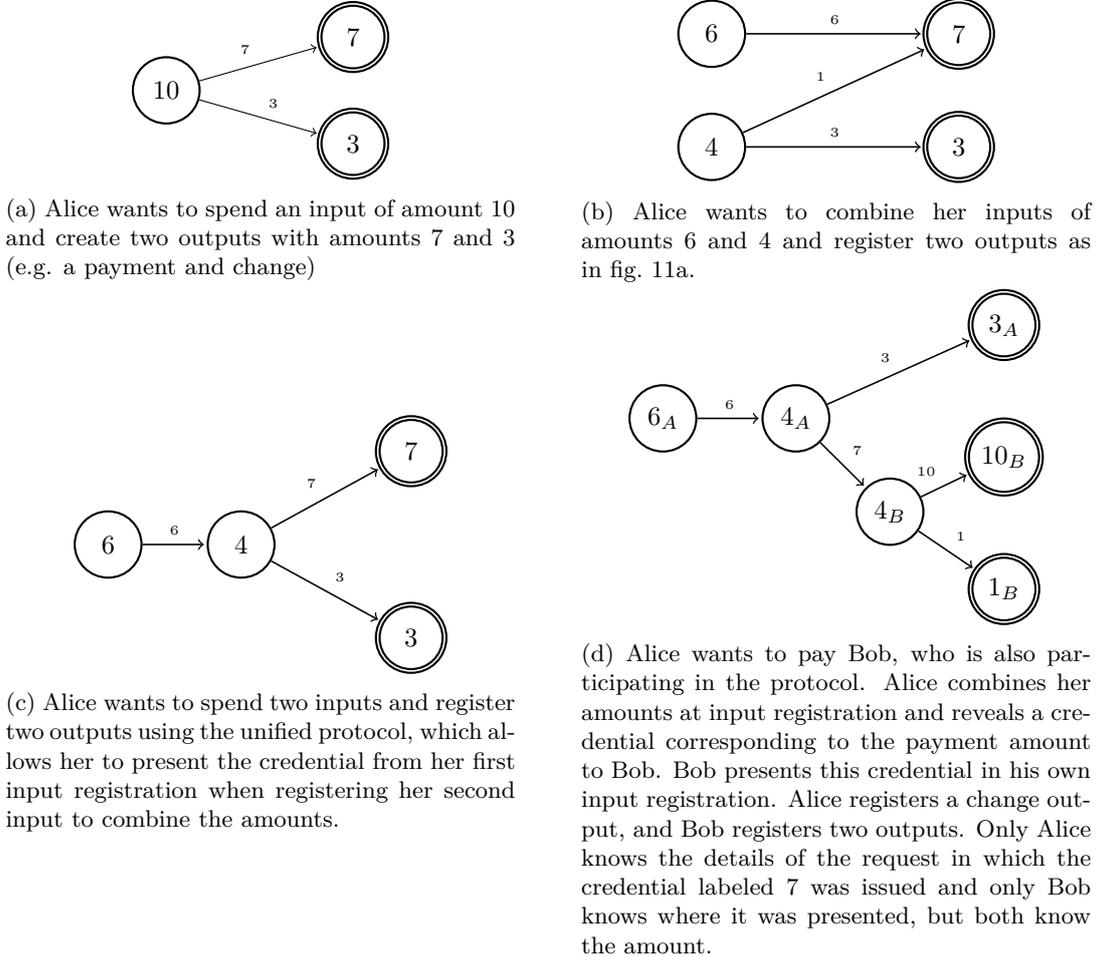


Figure 11: Credential usage examples

6 WabiSabi Credentials

In this section we provide the details of the unified protocol (section 5.3.3) and its use of the KVIC 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. $\text{HashTo}\mathbb{G} : \{0, 1\}^* \mapsto \mathbb{G}$ 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:

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

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

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 $\text{sk} = (w, w', x_0, x_1, y_a) \in_R \mathbb{Z}_q^5$.

¹⁵For publicly verifiable credentials the scheme in [BL13] can be substituted for [CPZ19] with relatively minor changes to the proofs.

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.

6.1 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 6.5). 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 proof that the amounts are in the allowed range, which ensures that finite field arithmetic will not overflow and the values will behave like positive integers when added or subtracted:

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

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

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

6.2 Credential Issuance

If the coordinator accepts the requests (see sections 6.3 to 6.5), 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 $\text{MAC}_{sk}(M_{a_i})$, where:

$$t_i \in_R \mathbb{Z}_q \quad U_i = \text{HashToG}(t_i) \quad V_i = G_w^w U_i^{x_0 + x_1 t_i} M_{a_i}^{y_a}$$

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 $iparams = (C_W, I)$:

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

6.3 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 $z_{0_i} = -t_i z_i \pmod{q}$ 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:

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

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

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

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} .

6.4 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} :

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

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

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.

6.5 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:

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

where

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

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.

6.6 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 it is possible to 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 $G'_a, G'_{x_0}, G'_{x_1}, G'_V$. Assuming the coordinator is not able to attack network level privacy and proofs of knowledge are unconditionally hiding this would ensure unconditional unlinkability.

7 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 CoinJoin is 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, and further research into its safe application is ongoing.

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.

7.1 Availability

7.1.1 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. The coordinator can also drop messages causing any user to appear to be non-compliant, and therefore disrupt the protocol arbitrarily.

7.1.2 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, allowing the remaining honest participants to restart the protocol and ensuring that a valid transaction can be output after a finite number of attempts. Denial of service is not costless because unspent transaction outputs are a limited resource.

7.2 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 the presentation of credentials cannot be linked back to their issuance. Since every registration request is only associated with a single input or output, the only information that would need to be broadcast publicly on the blockchain if the protocol succeeds is revealed directly.

However, because the protocol may be repeated several times before resulting in a valid transaction, and because the transaction data is still overt the unlinkability of credentials is not enough to guarantee privacy, and there are other means participants or the coordinator could attack it.

7.2.1 Passive attacks

We can model registration requests as vertices of a directed acyclic graph labeled by Δ_a and credentials as edges labelled by the amount in the attribute, connecting the registration request where a credential was issued to where it was presented, much like the graph depicted in section 5.5. Apart from the bootstrap requests which have indegree 0 (no credentials are presented) and final output requests which have outdegree 0 (credentials must be requested but not used since there are no subsequent registrations), this graph is k -regular. 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, and so long as honest participants are able to make their registrations it will be balanced (both vertex and edge labels will sum to 0).

The unlinkability of credentials and hiding property of the amount commitments obscure the edges and their labels from the coordinator and other participants. However, the coordinator observes registration requests according to a partial order which is an extension of the partial order defined by the transitive completion of the graph. In other words, the coordinator knows that parallel requests do not share an edge and that dependent requests must be made sequentially. Information about the order and timing of requests as well as the inherently overt transaction data relayed in the registration requests can therefore constrain the graph topology allowing the coordinator to draw inferences about the edges despite not being able to observe the edge set directly.

Clients can mitigate these leaks by minimizing dependencies between requests and scheduling requests randomly during the registration phases. Additional reissuance requests can be made by clients, including clients which do not actively participate in the transaction (cover traffic can be created using only the bootstrap credentials).

7.2.2 Active attacks

Sybil attacks [Dou02] are an inherent threat to privacy in 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 inputs¹⁶, as well as a cost imposed by mining fees.

¹⁶See also JoinMarket fidelity bonds: <https://gist.github.com/chris-belcher/18ea0e6acdb885a2bfbdee43dcd6b5af>

An attacker attempting to Sybil attack all CoinJoins would need to control some multiple of the combined Bitcoin 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 may 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.

A malicious coordinator could also delay the processing of requests in order to learn more through timing and ordering leaks. In the worst case, the coordinator can attempt to linearize all requests by delaying individual to recover the full set of labelled 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.

Similarly the coordinator may delay information such as the set of ownership proofs or the final unsigned transaction. In the case of the latter, this can be used to learn about links between inputs. This is because a signature can only be made after the details of the transaction are known. If the unsigned was only known to one user but multiple inputs have provided signatures, it follows that those inputs are owned by the same user.

Since the coordinator must be trusted with regards to denial of service a more practical variant of this attack would involve more subtle delays followed by sabotaging multiple successive rounds during the signing phase in order to learn of correlations between registrations while maintaining deniability.

More generally denial of service can amplify attacks on unlinkability, as it can be used to perform intersection attacks. The coordinator always learns the requested inputs and outputs, even if a round fails, and is able to partition users. A malicious participant will also learn all the input and output registrations if they wait until the signing phase to defect¹⁷.

7.3 Theft prevention

Since the output of the protocol is a single Bitcoin transaction theft is prevented by only signing the transaction if the outputs are as expected, so the protocol inherits theft resistance directly from Bitcoin's security model.

A malicious user could claim more funds than she registered if and only if she is able to forge the coordinator's MAC on some credential. This is certainly not possible unless credentials can be forged under chosen message attacks, but even if that were the case this would only result in denial of service, as the honest users would refuse to sign such a transaction.

7.4 Security Proofs

The above outlined security and privacy guarantees directly follow from the underlying KVAC scheme of [CPZ19]. The unlinkability of WabiSabi credentials follows from unlinkability and the denial of service resistance of the WabiSabi protocol follows from unforgeability. We refer the to [CPZ19] for formal security arguments.

8 Performance

In this section we discuss the efficiency of our protocol. Because the protocol can only complete successfully when all participants are both honest and available, and due to the underlying anonymous overlay network it is desirable to operate with as little communication complexity as possible. Reducing communication costs can also make the protocol more robust against traffic analysis by a global passive adversary or targeted censorship by an active one. Computational complexity is mainly a consideration for the coordinator it must process the requests of all users, which follow a bursty pattern.

8.1 Theoretical performance

The following table estimates communication overheads for the protocol assuming simple Σ -protocol for linear relations [BS20, 19.5.3 pp. 747-8, 20.4.1 pp. 792]. These proofs are amenable to compression [BBB+18;

¹⁷If `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.

AC20] and could be reduced to a size that is logarithmic in the bit width and the number of credentials per request.

	Request						Response	
	M_a	π^{range}	C_a	S	$(\pi^{serial} \wedge \pi^{MAC})$	π^{sum}	MAC	$\pi^{iparams}$
$\#\mathbb{G}$	k	$k(2n+1)$	k	k	$4k$	1	k	$3k$
$\#\mathbb{Z}_q$	—	$k(3n+1)$	—	—	$5k$	2	k	$5k$

Table 1: Communication overhead of the WabiSabi protocol. Each round trip corresponds to a single registered input and k denotes the number of requested credentials. The bit-width of the range proofs is denoted as n , and for bootstrap requests its value is zero.

8.2 Concrete performance

A proof of concept implementation of the credential system has been added to the open source Wasabi wallet¹⁸. This implementation relied on `NBitcoin.Secp256k1`¹⁹ a port of `libsecp256k1`²⁰. Due to limitations of the .NET runtime only 32-bit limb field elements are supported. All proofs were made non-interactive using the strong variant of the Fiat-Shamir transform [FS87; BPW12; Ham17].

Performance was measured on an Intel Core i7-7500U CPU 2.70GHz (Kaby Lake) using BenchmarkDotNet version 0.12.1 on Ubuntu 20.04 with .NET Core version 5.0.101. We ran a unified registration request cycle including bootstrapping and parameter generation with $k = 2$ and 51 bit range proofs²¹, and observed a 1.278 seconds mean time with a standard deviation of 0.103 seconds in 93 iterations.

9 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. 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 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 Blindcoin²², whereas other protocols utilize Bitcoin’s scripting capabilities to secure user funds.

Additional related work on Bitcoin privacy that is based on longer lived payment channels [GM17; TMM19] is less directly comparable and have been omitted. Some of the listed approaches, namely TumbleBit [HAB+17], BlindCoin [VR15] and custodial services can be used more similarly, but here we consider their use in mixes.

In tables 2 and 3, n denotes the number of participants. The number of transactions is not necessarily comparable because different protocols have significantly different transaction structures. Cost in terms of network fees is better provided in terms of the block weight, in terms of the minimum the number of outputs created and destroyed, with any change outputs incurring additional constant overhead. Overall wait times

¹⁸<https://github.com/zkSNACKs/WalletWasabi>

¹⁹<https://github.com/MetacoSA/NBitcoin/tree/master/NBitcoin.Secp256k1>

²⁰<https://github.com/bitcoin-core/secp256k1/pull/486>

²¹See: <https://github.com/zkSNACKs/WasabiBenchmark>

²²Both protocols have provisions for accountability of the mixes, mitigating theft concerns

are also bounded by the minimum number of phases which require transactions to be confirmed in a block. Furthermore, some protocols like Chris Belcher’s recent CoinSwap proposal [Bel20] are designed to provide robust privacy with a single iteration of the protocol, whereas others such as Mixcoin [BNM+14] require multiple mixes to achieve privacy. Secondly, the various protocols are qualitatively different, so comparing overall cost for a desired level of privacy is difficult. 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.

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

*Transaction structure guarantees minimal ambiguity in sub-transaction assignments

†Requires only 2 honest participants.

‡Assuming server does not participate as a verifier.

§Transaction ambiguity guarantees are probabilistic, with additional computational difficulty under assumptions.

¶Transactions are initiated and coordinated by users buying mixing offers on a decentralized marketplace.

Table 2: CoinJoin variants. Inherently theft resistant, one overt transaction with $\mathcal{O}(n)$ weight required.

	Msg.	Txn.	Wgt.	Conf.	Coord.	Den.	Theft Prev.	Privacy against		
								Srv.	Part.	Chain
Custodial	-	-	-	2	cent.	-	✗	TTP	TTP	-
CoinJoin [Max13a]	-	1	n	1	-	-	✓	-	-	overt*
CoinSwap [Max13b]	$\mathcal{O}(1)$ †	$2n$ †	$2n$ †	2	p2p	var.	✓	n.a.	✗	disjoint
Xim [BOL+14]	n.a.	$7n$ †	$7n$ †	7	p2p‡	var.	✓	n.a.	✗	disjoint
Mixcoin [BNM+14]	$2n$ §	$2n$ §	$2n$ §	2 §	cent.	var.¶	TTP	TTP	TTP	disjoint
Blindcoin [VR15]	$4n$	$2n$	$2n$	2	cent.	fix.	TTP	✓	✓	disjoint
CoinParty [ZGH+15]	$\mathcal{O}(nm)$ **	$2n$	$2n$	2	dec.**	fix.	TTP††	✓‡‡	✓	covert
TumbleBit [HAB+17]	$12n$	$4n$	$4n$	2	cent.	fix.	✓	✓§§	✓	disjoint
CoinSwap[Bel20]	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	2	taker	var.	✓	n.a.	✓	covert

*excluding PayJoin transactions, which may be covert

†Both CoinSwap and Xim fix $n = 2$

‡Advertisements are made on the blockchain

§Number of confirmations per mixnet stage

¶Fixed chunk size recommended for privacy, but technically not required.

|| Provides server accountability

**CoinParty involves n users and m mixing peers, which are considered servers in this comparison

†† Assuming $\frac{2}{3}$ of mixing peers (servers) are honest.

‡‡ Assuming at least 2 users are honest mixing peers learn input and output sets but no links.

§§ In its classic tumbler mode TumbleBit provides k -anonymity per multi-transaction epoch.

Table 3: Broader comparison of Bitcoin privacy techniques.

Since Chaumian CoinJoins have been successfully deployed despite some limitations (see section 2.2), it appears that centrally coordinated CoinJoins achieve a conservative 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.

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.

10 Future Directions

Fully addressing the limitations discussed in section 2.2 is the subject of ongoing work instantiating the credential scheme introduced in this work with a concrete protocol and a specific transaction structure. Due to the transparency of Bitcoin transactions the transaction structure further study is required in order to make use of the additional flexibility without compromising privacy.

The scope of such inquiries also extends to usability and user interface questions. For instance, making payments from CoinJoin transactions may benefit not just in costs but also in privacy by scheduling and batching payments, a technique which has mostly been applied to high volume wallets such as exchanges but not to wallets designed for end users. Payments made within a CoinJoin by transferring credentials (as in fig. 11d) present even more of a challenge for usability both due to the interactivity requirements and because of the departure from familiar mechanisms such as Bitcoin addresses.

11 Conclusion

In this work, we introduced WabiSabi, an application of the keyed-verification anonymous credentials scheme proposed in [CPZ19] to centrally coordinating CoinJoin transactions. WabiSabi builds on Chaumian CoinJoins, adding support for arbitrarily variable amounts. The goal of extending previous work in this way is as an enabler, removing restrictions imposed on currently deployed solutions and opening up new use cases. We note that the credential scheme can also be applied in centrally coordinated settings which are not necessarily based on CoinJoins for a wider range of theft and availability threat models.

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, Tim Ruffing, Ruben Somsen and Greg Zaverucha to this paper.

References

- [AC20] Thomas Attema and Ronald Cramer. “Compressed Σ -Protocol Theory and Practical Application to Plug & Play Secure Algorithmics.” In: *IACR Cryptol. ePrint Arch.* (2020). URL: <https://eprint.iacr.org/2020/152>.
- [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.
- [AW04] Hagit Attiya and Jennifer Welch. *Distributed computing: fundamentals, simulations, and advanced topics*. Vol. 19. John Wiley & Sons, 2004.
- [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).
- [BL13] Foteini Baldimtsi and Anna Lysyanskaya. “Anonymous Credentials Light.” In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. CCS ’13. Berlin, Germany: Association for Computing Machinery, 2013, pp. 1087–1098. ISBN: 9781450324779. DOI: 10.1145/2508859.2516687.
- [BMT+17] Christian Badertscher, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. “Bitcoin as a transaction ledger: A composable treatment.” In: *Annual International Cryptology Conference*. Springer, 2017, pp. 324–356.
- [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.
- [BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. “How Not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios.” In: *Advances in Cryptology – ASIACRYPT 2012*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 626–643. ISBN: 978-3-642-34961-4. DOI: 10.1007/978-3-642-34961-4_38.
- [Bra00] Stefan Brands. *Rethinking public key infrastructures and digital certificates: building in privacy*. Mit Press, 2000.
- [Bra94] Stefan Brands. “Untraceable Off-line Cash in Wallet with Observers.” In: *Advances in Cryptology – CRYPTO’ 93*. Ed. by Douglas R. Stinson. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 302–318. ISBN: 978-3-540-48329-8. DOI: 0.1007/3-540-48329-2_26.
- [BS20] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. 2020. URL: <http://cryptobook.us/>.
- [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.
- [FS87] Amos Fiat and Adi Shamir. “How To Prove Yourself: Practical Solutions to Identification and Signature Problems.” In: *Advances in Cryptology – CRYPTO’ 86*. Ed. by Andrew M. Odlyzko. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 186–194. ISBN: 978-3-540-47721-1. DOI: 10.1007/3-540-47721-7_12.
- [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).
- [GLL18] A. Gaihre, Y. Luo, and H. Liu. “Do Bitcoin Users Really Care About Anonymity? An Analysis of the Bitcoin Transaction Graph.” In: *2018 IEEE International Conference on Big Data (Big Data)*. 2018, pp. 1198–1207. DOI: 10.1109/BigData.2018.8622442.
- [GM17] Matthew Green and Ian Miers. “Bolt: Anonymous Payment Channels for Decentralized Currencies.” In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. New York, NY, USA: Association for Computing Machinery, 2017, pp. 473–489. ISBN: 9781450349468. DOI: 10.1145/3133956.3134093.
- [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.

- [Ham17] Mike Hamburg. “The STROBE protocol framework.” In: *IACR Cryptol. ePrint Arch.* (2017). URL: <https://eprint.iacr.org/2017/003>.
- [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/CBDCCom/IoP/SmartWorld)*. IEEE, 2016, pp. 368–373. DOI: 10.1109/UIC-ATC-ScalCom-CBDCCom-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).
- [MB17] M. Möser and R. Böhme. “Anonymous Alone? Measuring Bitcoin’s Second-Generation Anonymization Techniques.” In: *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*. 2017, pp. 32–41. DOI: 10.1109/EuroSPW.2017.48.
- [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/ICSS*. IEEE, 2017, pp. 522–529. DOI: 10.1109/Trustcom/BigDataSE/ICSS.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).
- [Nak09] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. 2009. DOI: 10.1.1.221.9986. URL: <https://bitcoin.org/bitcoin.pdf>.
- [Nic15] Jonas David Nick. “Data-driven de-anonymization in Bitcoin.” MA thesis. ETH-Zürich, 2015.
- [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.
- [PHE+17] Ania M Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. “The loopix anonymity system.” In: *26th {USENIX} Security Symposium ({USENIX} Security 17)*. 2017, pp. 1199–1216.
- [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.

- [RSG98] Michael G Reed, Paul F Syverson, and David M Goldschlag. “Anonymous connections and onion routing.” In: *IEEE Journal on Selected areas in Communications* 16.4 (1998), pp. 482–494.
- [Swe02] Latanya Sweeney. “K-Anonymity: A Model for Protecting Privacy.” In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570. ISSN: 0218-4885. DOI: 10.1142/S0218488502001648.
- [TMM19] Erkan Tairi, Pedro Moreno-Sanchez, and Matteo Maffei. “A2L: Anonymous Atomic Locks for Scalability and Interoperability in Payment Channel Hubs.” In: (2019). URL: <https://eprint.iacr.org/2019/589>.
- [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. “Coin-Party: 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.