A Transaction-Level Model for Blockchain Privacy

François-Xavier Wicht[1](✉), Zhipeng Wang[2], Duc V. Le[3], and Christian Cachin[1]

[1] Institute of Computer Science, University of Bern, Bern, Switzerland
{francois-xavier.wicht,christian.cachin}@unibe.ch
[2] Imperial College London, London, UK
zhipeng.wang20@imperial.ac.uk
[3] Visa Research, Foster City, USA
duc.le@visa.com

Abstract. Considerable work explores blockchain privacy notions. Yet, it usually employs entirely different models and notations, complicating potential comparisons. In this work, we use the Transaction Directed Acyclic Graph (TDAG) and extend it to capture blockchain privacy notions (PDAG). We give consistent definitions for untraceability and unlinkability. Moreover, we specify conditions on a blockchain system to achieve each aforementioned privacy notion. Thus, we can compare the two most prominent privacy-preserving blockchains – Monero and Zcash, in terms of privacy guarantees. Finally, we unify linking heuristics from the literature with our graph notation and review a good portion of research on blockchain privacy.

Keywords: Blockchain privacy · Untraceability · Unlinkability

1 Introduction

Public blockchains are inherently transparent and most cryptocurrencies fail to meet even the basic privacy standards. For instance, each transaction conducted on the two most prominent blockchains, Bitcoin and Ethereum, fully disclose senders, recipients and exchanged amount. Furthermore, identities are quite thinly protected behind pseudonymous addresses, which can easily be compromised by various mechanisms [5,6,22,36]. Once they are, restoring privacy becomes a challenging and expensive endeavor.

Many techniques, add-on solutions, or recommended behaviors exist to increase the base level of privacy. One such behavior is first for users to own multiple addresses and second to change them frequently. Yet, such a technique implies that users' wealth is scattered over several addresses. At some point, if a user wants to operate multiple of her addresses to conduct a single transaction, such a pattern is noticeable [2,27,31]. This linking is based on the assumption

D. V. Le—The main part of the work was conducted while the author was at the University of Bern.

that multiple coins spent together have the same ownership. However, it is possible for various parties to combine their coins and send them to whomever they wish. This is the base principle of the Bitcoin add-on privacy solution *Coinjoin* [25]. A more sophisticated mechanism exists in smart-contract-enabled blockchains such as Ethereum with *Tornado Cash* [33] and related mechanisms. They also allow different users to mix their cryptocurrency together to obscure the origin of the funds. Yet, in that scenario, individuals deposit their coins and receive a note allowing them to withdraw their funds later. Thanks to zero-knowledge proofs, the link between deposits and withdrawals is not revealed, while preventing users from withdrawing funds twice. Unfortunately, the opt-in nature of these tools does not increase the default privacy of the underlying chains. Moreover, there exist multiple ways to decrease the unlinkability properties of add-on privacy tools, as shown by several studies [18,39]. This is also possible because, in such blockchains, no inherent mechanisms exist to hide the origin of transactions. Thus, chain analysis can easily be conducted to trace coins back to their origin.

Privacy-preserving blockchains address this issue by hiding the sender of transactions and providing confidentiality for the amounts transferred. For example, Zcash builds a so-called shielded pool where transactions are anonymized via zero-knowledge proofs [4]. In contrast, Monero expands on the CryptoNote protocol to hide the source of transactions with ring signatures [35]. Zcash and Monero are thus privacy-preserving cryptocurrencies, and they have attracted a lot of attention from the public and from research. However, despite extensive studies, some questions remain open about privacy in blockchain.

– How do the different privacy mechanisms and tools compare?
– What are the relevant structures in a blockchain and the ledger that either protect privacy or break it?
– How do different shapes of privacy manifest themselves in a cryptocurrency using a blockchain?

This work provides an answer to those questions by defining two main privacy notions, *untraceability* and *unlinkability*, using a formal model of the blockchain transaction structure. For that, we build upon a pre-existing model, the "Transaction Directed Acyclic Graph" (TDAG) defined by Cachin et al. [7], and devise the Privacy-preserving transaction DAG (PDAG) to capture additional properties such as the hiding of the transaction's source. This model allows us to contribute to the field in several aspects.

– We provide definitions for untraceability and unlinkability, and thus lift the ambiguity between those terms.
– Also, we give a structural view over blockchain privacy.
– Furthermore, we give conditions on a system for achieving each notion.
– This allows us to compare the different blockchain systems in terms of privacy guarantees.
– Finally, thanks to our notation, we can unify analysis patterns found in the literature that are usually used to link related addresses.

We first recall the building blocks of the TDAG and present the PDAG in Sect. 2. Section 3 tackles the definition of untraceability and unlinkability using PDAG's elements. In the full version of this work [40], we apply our models to add-on privacy solutions, Coinjoin and Tornado Cash, and two privacy-preserving blockchains, namely Monero and Zcash.

2 Privacy-Preserving Transaction Directed Acyclic Graph

This section introduces the *Privacy-Preserving Transaction Directed Acyclic Graph* (PDAG). This model is based upon the Transaction Directed Acyclic Graph (TDAG) described by Cachin et al. [7]. We therefore start by presenting the TDAG, give motivations for a privacy-preserving version, and finally define the PDAG.

2.1 Background: Transaction Directed Acyclic Graph

A *Transaction Directed Acyclic Graph* (TDAG) [7] is a graph $G = (\mathcal{V}, \mathcal{E})$ with two types of vertices, namely states $\mathcal{S}$ and witnesses $\mathcal{W}$, such that $\mathcal{V} = \mathcal{S} \dot{\cup} \mathcal{W}$. The edges, in turn, represent transitions between states and witnesses. In other words, edges $\mathcal{E}$ represent transactional relations between states and witnesses. Edges are partitioned into three different types: consuming, observing, and producing edges, denoted respectively by, $\mathcal{E}_C$, $\mathcal{E}_O$, and $\mathcal{E}_P$.

- **States** ($\bigcirc$). A state $s \in \mathcal{S}$ is the first type of vertex, denoting the output state of a transaction. It refers to an individual asset on the blockchain, a digital coin, a coin controlled by a particular address, or the state of a smart contract. A state is the result of a transaction and can transition to other states through subsequent transactions. The full context of the blockchain consists of all states that exist at a given time. A unique *genesis state*, represents the initial state of the blockchain network. There is only one genesis state, since the blockchain can only be bootstrapped once.
- **Witnesses** ($\square$). A witness $w \in \mathcal{W}$ is the second kind of vertex, representing any data in a transaction for it to be valid according to the blockchain validation rules. Every blockchain transaction contains exactly one witness.
- **Consuming edges**($\bigcirc \longrightarrow \square$). A consuming edge $e \in \mathcal{E}_C$ connects a state s to a witness w and expresses that s is consumed by the transaction involving w. A state can be consumed at most once. After this, no other transaction may consume s. A state that is consumed has been "updated" or "overwritten" by the transaction.
- **Producing edges** ($\square \longrightarrow \bigcirc$). A producing edge $e \in \mathcal{E}_P$ connects a witness w to a state s and expresses that s is created by the transaction corresponding to w.
- **Observing edges** ($\bigcirc \dashrightarrow \square$). An observing edge $e \in \mathcal{E}_O$ connects a state s to a witness w and expresses that s enters into a transaction represented by w, but that it stays available to another transaction for consumption.

Those elements constitute the building blocks of a TDAG G, and note that G is bipartite. Additionally, every *transaction* in a TDAG $G = (\mathcal{V}, \mathcal{E})$ is a subgraph denoted by $T_w = (\mathcal{S}' \dot\cup \{w\}, \mathcal{E}')$ composed of a subset of vertices $\mathcal{V}' = \mathcal{S}' \dot\cup \{w\} \subseteq \mathcal{V}$, a subset of edges $\mathcal{E}' \subseteq \mathcal{E}$, a single witness w, a set of input states $\mathcal{S}_I \subseteq \mathcal{S}'$ and a set of output states $\mathcal{S}_O \subseteq \mathcal{S}'$, such that

- Every input state in $\mathcal{S}_I$ is a source (has indegree zero);
- Every output state in $\mathcal{S}_O$ is a sink (has outdegree zero);
- $\mathcal{V}' = \mathcal{S}_I \dot\cup \mathcal{S}_O \dot\cup \{w\} \subseteq \mathcal{V}$;
- Every edge e in $\mathcal{E}'$ is either a consuming edge e_C, or an observing edge e_O and links some input state $s_i \in \mathcal{S}_I$ to w, or it is a producing edge and links w to some output state $s_O \in \mathcal{S}_O$.

In Fig. 1, we depict an example of a transaction T_w. A possible interpretation of T_w is an asset transfer from three states s_0, s_1, s_2 to two different states s_3, s_4. The input states of T_w are being consumed, while the output states are being produced. The produced states are now available for consumption by subsequent transactions. Regarding multiple transactions, the transaction graph must follow certain rules and have a well-defined structure. The graph starts with the genesis state s_g. Regular states are produced exactly once and are consumed at most once. There are otherwise no restrictions on the number of observing edges. This construction allows the TDAG to represent UTXO-based (e.g., Bitcoin), account-based blockchains (e.g., Ethereum), and possibly others. However, we argue in the next subsection that the TDAG lacks certain elements to represent privacy-preserving blockchains (e.g., Monero and Zcash).

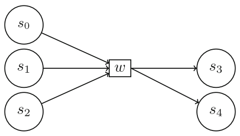

Fig. 1. Illustration of a transaction $T_w = (\mathcal{V}', \mathcal{E}')$, where $\mathcal{V}' = \{s_0, s_1, s_2, w, s_3, s_4\}$ and $\mathcal{E}' = \{(s_0, w), (s_1, w), (s_2, w), (w, s_3), (w, s_4)\}$. The input states $\{s_0, s_1, s_2\}$ are consumed, whereas the output states $\{s_3, s_4\}$ are produced.

2.2 Privacy-Preserving Transactions

This section motivates and defines building blocks to introduce additional components for the TDAG to represent privacy-preserving blockchains. We motivate these additions in regard to conflict-freedom. Cachin et al. [7] give conflict-freedom conditions to a TDAG. Conflicts in a blockchain underlying a cryptocurrency occur in case of double-spending. Such a situation arises if the same unique asset is spent more than once. In UTXO-based blockchains, the system tracks unspent transaction outputs (UTXO) to determine which transactions may be spent. On privacy-preserving chains, the UTXO set is built differently so as not to disclose any information to its users. For example, in Monero, each transaction generates a unique “key image”, and key images used more than once are rejected by the blockchain [21]. A transaction is, therefore, valid if a key image does not conflict with an existing one. Zcash’s shielded equivalent of a UTXO is a “commitment” [17]. A node can, in turn, spend a commitment by revealing a “nullifier”.

Masking. In a TDAG, conflict-freedom holds when every state is produced and consumed at most once, i.e., coins cannot be spent twice, nor can they be duplicated. Now, the same rule applies to privacy-preserving blockchains. Yet, state consumption (coin spending) is not directly visible to the network beyond the sender and receiver. Say, for example, that Alice mixes her input with ten other decoy inputs using a cryptographic mechanism for her to remain anonymous. The edges from these ten decoys are observing edges, since we do not alter them, but only read them. However, Alice’s spent coins are effectively consumed but appear only as being observed for non-involved parties. This introduces a new type of edges, *masking edges* $\mathcal{E}_M$.

- **Masking edges** ($\bigcirc \rightsquigarrow \square$). A masking edge $e \in \mathcal{E}_M$ connects a state to a witness and is a consuming edge that hides behind cryptographic mechanisms. More precisely, it appears as an observing edge to nodes aside from the author, but is a consuming edge. As its name suggests, it masks its actual behavior. It otherwise follows the same rules as consuming edges.

Moreover, such addition brings ambiguity towards observing edges, since an outside observer may not distinguish an observing edge from a masking one. We thus call *ambiguous edges* the union of observing and masking edges, i.e., $\mathcal{E}_O \mathbin{\dot\cup} \mathcal{E}_M$.

Nullifiers. Privacy-preserving blockchains also output additional components at each transaction. These components allow constructing a cryptographic mapping with unspent transactions. This serves to prevent double-spending while hiding the source of a transaction. To be more general, hidden behaviors in the blockchain require users to output a specific value as a *nullifier*. In addition, the exact same behavior must produce the same value. Therefore, this redundancy is visible if users try to benefit from the same asset or service multiple times. The blockchain validation rules will thus forbid one such transaction since two values collide. This implies that each transaction must verify that the new values do not conflict with pre-existing ones. In turn, the total set of generated values cannot be discarded at any point, since it must be read for each transaction to prevent conflicts. We therefore introduce this additional type of state that can only be observed or produced. Depending on the implementation, this component may have different names, e.g., key images for Monero, nullifiers for Zcash, or withdrawal commitments for Tornado Cash [33]. We thus introduce a new type of vertex that we call (without loss of generality) *nullifiers* $\mathcal{N}$.

– **Nullifiers** ($\diamond$). A nullifier $n \in \mathcal{N}$ is a state that encapsulates cryptographic commitments which prevent users from benefiting from the same asset or service multiple times. It may only be produced and observed, but not consumed.

Those two additions lead us to call this model the nullifier-based one, as opposed to the "UTXO-based" or "account-based" model. We formalize the different components in the next subsection.

2.3 Privacy-Preserving Transaction DAG

We here define our model, the *Privacy-preserving Transaction DAG* (PDAG). We begin by formally specifying the PDAG and its elements, and next define a transaction in a PDAG. We use the notation $\mathcal{E} \subseteq X \times Y$ to denote a relation $\mathcal{E}$ between X and Y. Also, the expression $(x, y) \in \mathcal{E}$ can be written as $x\mathcal{E}y$, and we define the set $x\mathcal{E}\star$ as $\{y : x\mathcal{E}y\}$, and write its cardinality as $|x\mathcal{E}\star|$. We speak of the closed neighborhood of a vertex v to mention any adjacent vertex to v, including v itself, and denote it by $N[v]$.

Definition 1 (PDAG). *A* Privacy-preserving Transactional DAG (PDAG) *is a directed unweighted graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{S} \dot\cup \mathcal{N} \dot\cup \mathcal{W}$ are the vertices and $\mathcal{E} = \mathcal{E}_C \dot\cup \mathcal{E}_M \dot\cup \mathcal{E}_O \dot\cup \mathcal{E}_P$ are the edges, where vertices and edges are each partitioned into respective sets. The sets $\mathcal{S}$, $\mathcal{N}$, $\mathcal{W}$, $\mathcal{E}_C$, $\mathcal{E}_M$, $\mathcal{E}_O$, $\mathcal{E}_P$ constitute the components of G. The set $\mathcal{S}$ denotes the states and contains a special state s_g called genesis. The nullifiers $\mathcal{N}$ are states that encapsulate cryptographic commitments. The set $\mathcal{W}$ denotes the witnesses. Edges are partitioned into four subsets, where $\mathcal{E}_C, \mathcal{E}_M, \mathcal{E}_O \subseteq \mathcal{S} \times \mathcal{W}$ and $\mathcal{E}_P \subseteq \mathcal{W} \times \mathcal{S}$. It satisfies the following conditions:*

1. *s_g does not have any producing or observing edges, and it has a single consuming edge: $|\star\mathcal{E}_P s_g| = 0 \wedge |s_g\mathcal{E}_P\star| = 0 \wedge \exists! w \in \mathcal{W} : s_g\mathcal{E}_C w$.*
2. *Every state, except for the genesis state, has exactly one producing edge: $\forall s \in \mathcal{S} \setminus \{s_g\}\, \exists! w \in \mathcal{W} : w\mathcal{E}_P s$.*
3. *Every state, except for the genesis state, may have multiple successors, but at most one among them is connected with a consuming or masking edge: $\forall s \in \mathcal{S} : |s\mathcal{E}_C \star \dot\cup\, s\mathcal{E}_M\star| \leq 1$.*
4. *Nullifiers $n \in \mathcal{N}$ have no consuming edges and no masking edges: $\forall n \in \mathcal{N}\, |n\mathcal{E}_C\star| = 0 \wedge |n\mathcal{E}_M\star| = 0$.*
5. *G is weakly connected.*
6. *G has no cycles.*

Note that for each witness w, there is a corresponding transaction t. We thus define a transaction accordingly. We use the notation $x \prec y$ to say that x comes before y in the topological order of the graph, and call x a predecessor of y.

Definition 2 (Transaction in a PDAG). *Given a PDAG $G = (\mathcal{S} \dot\cup \mathcal{N} \dot\cup \mathcal{W}, \mathcal{E})$ and a witness $w \in \mathcal{W}$, the* transaction with witness *w is the unique subgraph $T_w = (\mathcal{S}' \dot\cup\ \mathcal{N}' \dot\cup \{w\}, \mathcal{E}') \subset G$, where*

– *$\mathcal{S}'$ is the set of states connected to w:*

$$\mathcal{S} = \{s \in \mathcal{S} : s\mathcal{E}_C w \vee s\mathcal{E}_M w \vee s\mathcal{E}_O w \vee w\mathcal{E}_P s\};$$

– *$\mathcal{N}'$ is the subset of all nullifiers produced by predecessors and the ones produced by T_w: $\mathcal{N}' = \{w' \in \mathcal{W}, n \in \mathcal{N} : (w'\mathcal{E}_P n \wedge w' \prec w) \wedge w\mathcal{E}_P n\}$;*
– *$w \in \mathcal{W}$ is the witness of the transaction; and*
– *$\mathcal{E}'$ are the edges with both endpoints in $\mathcal{S}' \dot{\cup} \{w\}$.*

Furthermore, a transaction may be the one of multiple types, depending on the number of its inputs and outputs:

– **SISO.** Single-input, single-output transactions consist of one consuming (or masking) edge that connects an input state to a witness w and of a producing edge that links w to an output state.
– **SIMO.** Single-input, and multiple-output transactions possess producing edges from w to multiple output states.
– **MISO.** Multi-input, single-output transactions have a set of multiple consuming (or masking) and observing edges.
– **MIMO.** Multi-input, multi-output transactions contain multiple consuming (or masking) and observing edges, as well as multiple producing edges.

One special transaction, the initialization transaction, consists of a consuming edge linking the genesis state and a set of producing edges. The next section provides a minimal example that presents different types of transactions.

2.4 Example

We here assume a hypothesized privacy-preserving blockchain that uses an abstract cryptographic mechanism to hide the source of each transaction. This mechanism takes any previously produced state as decoy to the real input. For simplicity, we assume this blockchain does not incur fees for its transactions, and that the first transaction consists of a coin minting process. Each transaction outputs nullifiers to prevent double-spending. Assets are held by public keys (addresses) and to transfer said asset, one needs to prove knowledge of the private key with a signature.

In Fig. 2, we depict the PDAG of a sample execution of such blockchain. The first transaction T_{w_0} consists of the minting of coins to the address of the miner (s_0). The second transaction T_{w_1} represents the transfer from s_0 to four different addresses: s_1, s_2, s_3, s_4. This transaction links s_0 to w_1 with a masking edge, but does not include any decoy inputs. The witness w_1 verifies the validity of the signature and of the produced nullifier n_0. The third transaction T_{w_2} uses three decoy inputs s_2, s_3, s_4 to transfer assets from s_1 to three different addresses: s_5, s_6, s_7. These decoy keys are linked to w_2 with observing edges, since they are only read (and not consumed) by the transaction. The witness w_2 verifies the signature and the conflict-freedom between the previously generated nullifier (n_0) and the newly produced one (n_1). The last transaction T_{w_3} merges coins from two different addresses (s_4, s_7) to a single one (s_8). Transaction T_{w_3} takes five decoy addresses as inputs, s_1, s_2, s_3, s_5, s_6, which are linked with observing edges. The verification of the signatures and the conflict-freedom between the previous nullifiers (n_0, n_1) and the new one (n_2, n_3) take place at the witness w_3.

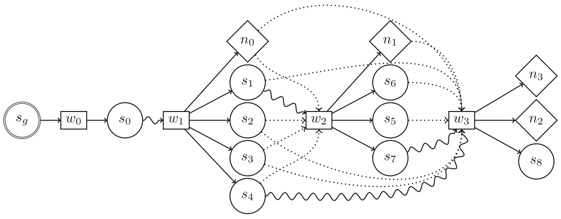

Fig. 2. Illustration of a PDAG with four transactions: T_{w_0}, T_{w_1}, T_{w_2}, and T_{w_3}. Each transaction T_i is the closed neighborhood subgraph $N_G[w_i] \subset G$.

3 Privacy Notions

This section formalizes two privacy notions in blockchain systems with respect to the PDAG, namely untraceability and unlinkability. We first describe the adversarial model, and secondly formalize the different privacy notions.

3.1 Adversarial Model

We use Π to denote the analyzed blockchain system and $\mathcal{A}$ for an adversary. We assume idealized cryptographic primitives and a computationally-unbounded $\mathcal{A}$. However, we assume that the adversary's focus lies on the transaction layer, and that $\mathcal{A}$ operates passively, solely observing and recording the transactions to compromise the privacy guarantees of Π. To establish the concepts of untraceability and unlinkability, we adopt the notion of anonymity sets, as introduced by Chaum [8]. Specifically, we refer to *untraceability sets* and *unlinkability sets*, respectively, which we define and quantify based on the PDAG.

3.2 Untraceability

Untraceability is commonly understood in the blockchain transaction layer as the property that for each transaction, all possible senders are equiprobable [17,24,30,35,41]. In other fields of computer science, this notion is sometimes addressed as *sender anonymity* [34]. According to the PDAG, a sender is an input state linked to a witness with a consuming or with a masking edge. In turn, a sender gets a degree of untraceability if the witness of a transaction has no consuming edge, and if the number of ambiguous edges is greater than one. Otherwise, the sender of the transaction is visible or, readily guessable. We call such a transaction trivially traceable. The input states of a transaction linked with ambiguous edges, thus form the untraceability set of the transaction, and any member of this set is equiprobable to be consumed. For $\mathcal{A}$, it is a matter of lifting the ambiguity behind these edges and to guess which of the states is in fact consumed. We adopt a combinatorial view of this problem and model it as a bipartite graph. This approach is similar to the study of anonymous communications introduced by Edman et al. [11] and later refined by Gierlichs et al. [14]. Given a PDAG $G = (\mathcal{S} \dot\cup \mathcal{W} \dot\cup \mathcal{N}, \mathcal{E})$, one can transform it into another bipartite graph $G^* = (\mathcal{S}^* \dot\cup \mathcal{W}^*, \mathcal{E}^*)$ by pruning vertices and edges. We only include in G^*, witnesses of regular transactions that exchange assets between addresses, and thus exclude mining, minting or any ordering transactions, and also consider input states to these witnesses with their corresponding edges, i.e.,

- $\mathcal{W}^* = \{w^* \in \mathcal{W} \mid w^* \text{ is a regular transaction } \} \subseteq \mathcal{W}$,
- $\mathcal{S}^* = \{s^* \in \mathcal{S} \mid (s^*, w^*) \in \mathcal{S} \times \mathcal{W}^*\} \subseteq \mathcal{S}$, and
- $\mathcal{E}^* = \mathcal{S}^* \times \mathcal{W}^* \subseteq \mathcal{S} \times \mathcal{W}$.

One such bipartite graph composed of three transactions is presented in Fig. 3 with $\mathcal{S}^*$ on the left and $\mathcal{W}^*$ on the right. Each transaction $T^*_{w_i}$ is the closed neighborhood $N_{G^*}[w_i]$ with $1 \leq i \leq 2$. If we consider the single transactions

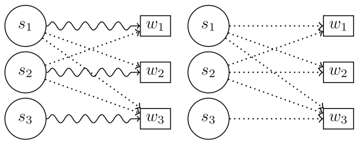

Fig. 3. Three transactions $T^*_{w_1}$, $T^*_{w_2}$, and $T^*_{w_3}$ have a total of three input states s_1, s_2 and s_3. On the left is the graph that represents the ground-truth with masking edges, and on the right, the view of the adversary. The masking edges highlight the maximum matching that $\mathcal{A}$ must guess.

$T^*_{w_1}$, $T^*_{w_2}$, and $T^*_{w_3}$, the traceability sets are $\{s_1, s_2\}$, $\{s_1, s_2\}$, and $\{s_1, s_2, s_3\}$, respectively. The senders of transactions are untraceable among two for w_1 and w_2, and among three for w_3. Furthermore, the masking edges form a maximum matching over the two sets of vertices. In this example, the matching is also perfect since each vertex is covered by the matching. However, this is not necessarily the case, since non-consumed states may also be part of the untraceability set. In addition, this maximum matching is usually not unique. In Fig. 3, we find two different maximum matchings: $M_1 = \{(s_1, w_1), (s_2, w_2), (s_3, w_3)\}$ and $M_2 = \{(s_1, w_2), (s_2, w_1), (s_3, w_3)\}$. Only M_1 highlights the actual senders of transactions in this case. If this matching was unique, an adversary could trivially know each source of transactions irrespective of the incoming degree of their witness. It indeed follows from the graph structure that each witness consumes at least one state, and hence that the state consumption is represented by a maximum matching. Therefore, one way of reducing the untraceability set is to only consider states linked with edges part of a maximum matching. This method has recently been studied by Vijayakumaran [38] and Egger et al. [12] to perform graph-based deanonymization on the CryptoNote protocol. They use the union of maximum matchings known as the Dulmage-Mendelsohn core [10] to prune edges from the bipartite graph and thus to compromise the untraceability of transactions. We here adapt the definition of the core of a bipartite graph G^* to our notation.

Definition 3 (Core [10]). *The* core *of a bipartite graph* $G^* = (\mathcal{S}^* \dot\cup \mathcal{W}^*, \mathcal{E}^*)$*, denoted by* $\mathbf{core}\,(G^*) = (\mathcal{S}^* \dot\cup \mathcal{W}^*, \mathcal{E}')$*, is a subgraph of* G^* *where* $\mathcal{E}' \subseteq \mathcal{E}^*$ *is the union of all maximum matchings of* G^*.

As highlighted by previous research [12,38], if the bipartite graph is not equal to its core, then a reduction of the traceability set is possible. In the example of Fig. 3, computing the core, results in the strict subset of edges $\{(s_1, w_1), (s_2, w_2), (s_3, w_3), (s_1, w_2), (s_2, w_1)\}$. In this case, the witness w_3 has only one remaining edge and is thus completely traceable. To formalize our definition of untraceability, we therefore only consider the traceability set of a transaction after reduction of the graph. This leads us to the definition of k-untraceability for a transaction, where k is the cardinality of the traceability set.

Definition 4 (k-untraceable Transaction). *Let* $T^*_w = (\mathcal{S}' \dot\cup \mathcal{N}' \dot\cup \{w\}, \mathcal{E})$ *be a* transaction *such that* $\mathcal{E} = \mathcal{E}'_O \dot\cup \mathcal{E}'_M$ *and* $T^*_w \subseteq \mathbf{core}\,(G^*)$*. We say that* T^*_w *is* k-untraceable *if and only if the set of ambiguous edges,* $\mathcal{E}'_O \dot\cup \mathcal{E}'_M$*, has cardinality* k *with* $k \geq 2$.

When an adversary $\mathcal{A}$ observes a PDAG, the best strategy to identify the source of a transaction is to first consider the trivially traceable transactions. If no such transaction exists, the second-best strategy for $\mathcal{A}$, without further knowledge, is to guess uniformly at random over the edges of transactions with the smallest untraceability set.

In this view, the untraceability guarantees of the PDAG are given by the *weakest* transaction, i.e., the transactions with the least untraceability guarantees. A PDAG does not provide untraceability if any of its transaction is traceable. Otherwise the PDAG's untraceability set is tied to the smallest untraceability set of its transactions. Consequently, for a PDAG to be k-untraceable, all its transactions must at least be k-untraceable.

Definition 5 (k-untraceable PDAG). *A PDAG* G *with the corresponding bipartite graph* G^* *is* k-untraceable *if and only if all transactions* T^*_W *in the core of* G^* *are at least* k*-untraceable.*

3.3 Unlinkability

Whereas untraceability addresses the relation between input states and witnesses, unlinkability is a notion that focuses on the complete set of states in the graph. Unlinkability refers to the inability of the adversary to link two different states or transactions together [2]. Now, if the adversary can link two states, she can also link transactions involving those states. We consequently focus on a more concise definition of unlinkability: for any two states, it is impossible to sufficiently distinguish whether they are related or not. This definition directly echoes the terminology of unlinkability of Pfitzmann and Hansen [34]. We here define a *link relation* to abstract the notion of *linking*.

Definition 6 (Link Relation). *The* link relation L *is an equivalence relation on the states* $\mathcal{S}$*, that is, any two states* $s_i, s_j \in \mathcal{S}$ *are linked whenever* $s_i L s_j$*. We write* $[s]_L$ *the equivalence class of* $\mathcal{S}$ *by* L*, to which* s *belongs, i.e.,* $[s]_L := \{s_i \in \mathcal{S} : sLs_i\}$*.*

Although blockchains function differently, the main concept of unlinkability is the same for various systems. It consists of the adversary trying to link states by grouping them into those that are related. Some states may easily be linked together as they hold the same address, which typically identifies asset holders in blockchains. We therefore introduce the address relation as a refinement of the link relation.

Definition 7 (Address relation). *The* address Relation A *is a refinement of the link relation* L *on the states* $\mathcal{S}$*, that is, any two states* $s_i, s_j \in \mathcal{S}$ *hold the same address whenever* $s_i A s_j$*. We write* $[s]_A$ *the equivalence class of* $\mathcal{S}$ *by* A*, to which* s *belongs, i.e.,* $[s]_A := \{s_i \in \mathcal{S} : sAs_i\}$*.*

However, users are encouraged to use different addresses to break the relation between several transactions. In this case, the linking is no longer trivial, and to our knowledge, there exists no systematic way of tying states together. Research on this subject uses heuristics to associate related states together. Those heuristics are mostly patterns within or across transactions, based on some evidence of shared ownership, that lead to believe that some states may be related [19]. Similarly, we model an adversary that uses an inferred link relation $\tilde{L}$ in her endeavor of compromising unlinkability. The adversary may use a very coarse relation that assumes that any two states are linked, or she may use finer relations such as heuristics presented in the literature or a machine-learning-based classifier. We represent here a few heuristics as an inferred linking relation $\tilde{L}$, and say that a subgraph $G' \subseteq G$ is *vulnerable* to $\tilde{L}$, if said relation applies to G'.

The most studied heuristic is certainly the multi-input heuristic, which assumes that multiple addresses part of a single transaction are linked [31]. We express this heuristic as a linking relation:

$$\forall s_i, s_j \in \star\mathcal{E}_C w : s_i \tilde{L} s_j. \tag{1}$$

Another well-studied heuristic addresses the use of a change-address: when a user sends coins, she might use a newly generated address (change-address) to receive the amount surplus. If a state with a new address is part of an output set of size two, one can assume that such state is linked to at least one input state of the same transaction:

$$\exists s_i \in w\mathcal{E}_P\mathcal{S}_O, \exists s_j \in \star\mathcal{E}_C w : |\mathcal{S}_O| = 2 \land \forall s \in \mathcal{S} : s_i \notin [s]_A \implies s_i \tilde{L} s_j. \tag{2}$$

Note that this condition has different variations in the literature. The above version is stronger with $|\mathcal{S}_O| = 2$ than with $|\mathcal{S}_O| \geq 2$, but the latter is usually more precise. Since the linking relations are expressed under logical statements, we can combine them to express more complex relations. For example, the multi-input heuristic, and the change-address relation can be combined to express that the state with change-address is linked to every input states:

$$\exists s_i \in w\mathcal{E}_P\mathcal{S}_O, \forall s_j \in \star\mathcal{E}_C w : |\mathcal{S}_O| = 2 \wedge \forall s \in \mathcal{S} : s_i \notin [s]_A \implies s_i \tilde{L} s_j. \quad (3)$$

In the full version of the paper [40], we translate around ten heuristics from the literature into inferred linking relations. We model here an adversary that exploits linking relations of the graph to find the different states that belong together. The equivalence class of a state s identified by the adversary is the set $[s]_{\tilde{L}}$, and we argue here that the dichotomy between $[s]_L$ and $[s]_{\tilde{L}}$ is a source of unlinkability.

As an example, consider transaction T_w in Fig. 4. We assume here that all states have a different address, that s_2 and s_3 are linked to s_4, and s_1 is linked to neither s_2, s_3, nor s_4. If an adversary exploits relation (1) to identify the states linked together, she will obtain $\{s_1, s_2, s_3\}$. In this scenario, s_2 and s_3 are accurately associated, but s_1 is wrongfully linked to s_2 and s_3. Although linking s_2 and s_3 together reduces unlinkability, assigning s_1 to the same equivalence class has the opposite effect. The unlinkability set is therefore decreasing with any addition of a linked state into the correct equivalence class, but increasing with additions of unlinked states. In the scenario of Fig. 4, s_1 is singly unlinkable among three and s_2, s_3 are both unlinkable among three, according to linking relation (1). The best situation for a state s in terms of unlinkability is clearly

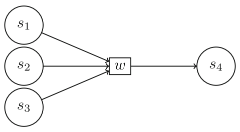

Fig. 4. Example of a transaction $T_w = (\mathcal{V}', \mathcal{E}')$ with a set of vertices $\mathcal{V}' = \{s_1, s_2, s_3, w, s_4\}$ and a set of edges $\mathcal{E}' = \{(s_1, w), (s_2, w), (s_3, w), (w, s_4)\}$.

to be singly unlinkable in the wrong equivalence class, i.e., $s \in [s']_{\tilde{L}} \setminus [s']_L$ with $s' \neq s$. Conversely, the worst case scenario is when the equivalence class built by the adversary, $[s]_{\tilde{L}}$, accurately reflects the linking between states, i.e., $[s]_{\tilde{L}} \subseteq [s]_L$.

Between those two edge cases, s still has some degree of unlinkability. This degree is directly related to the difference between the equivalence classes $[s]_{\tilde{L}}$ and $[s]_L$. In Fig. 5, we depict this situation, in which the outer, striped area represents the unlinked states. In contrast, the inner, gray area represents the set of states accurately linked by $\tilde{L}$. This set, de facto, contains the equivalent class $[s]_A$, since A is a refinement of L and states with the same address are trivially linkable for $\mathcal{A}$. Yet, the size of $[s]_{\tilde{L}} \setminus [s]_L$ is not quite representative of the degree of unlinkability of s. As an example, consider the two cases where:

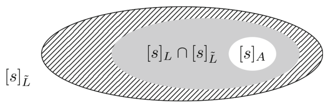

Fig. 5. Illustration of the unlinkability problem: the striped area represents the degree of unlinkability for state s and the gray area is the set of accurately linked states, which contains the set of states with the same address. (Color figure online)

1. $|[s]_{\tilde{L}}| = 3$ and $|[s]_{\tilde{L}} \cap [s]_L| = 2$ and
2. $|[s]_{\tilde{L}}| = 100$ and $|[s]_{\tilde{L}} \cap [s]_L| = 99$.

is 1. However, in the first case, an adversary has a chance of $\binom{2}{2}/\binom{3}{2} = \frac{1}{3}$ of guessing a linked pair, whereas in the second case, this chance is increased to $\binom{99}{2}/\binom{100}{2} \approx 0.98$. Reciprocally, unlinkability is the ratio of pairs a state s can mix into. We define the *unlinkability score* accordingly.

Definition 8 (Unlinkability Score). *The unlinkability score u of a state s according to an inferred linking relation $\tilde{L}$ is the ratio of the number of unlinked pairs to all possible pairs, i.e., $u = \binom{|[s]_{\tilde{L}} \setminus [s]_L|}{2} / \binom{|[s]_{\tilde{L}}|}{2}$.*

The unlinkability score u ranges from 0 to 1, where 0 denotes linkable states, and 1 represents singly-unlinkable states. We define u-unlinkability for a state with $u > 0$.

Definition 9 (u-unlinkable State). *A state s of a PDAG G is u-unlinkable according to an inferred linking relation $\tilde{L}$ if and only if exploiting said relation on G, results in an equivalence class $[s]_{\tilde{L}}$ with an unlinkability score $u > 0$.*

When observing a PDAG, the first best strategy for $\mathcal{A}$ is to link states with the same address together, i.e., $\tilde{L} = A$. In this case, for any state s, $[s]_{\tilde{L}}$ is subset of $[s]_L$, and thus the unlinkability score of all states $s' \in [s]_{\tilde{L}}$ is 0. If $[s]_A = \{s\}$, the adversary must rely on non-trivial inferred linking relations $\tilde{L}$. These relations may nonetheless yield an unlinkability score of 0 depending on their accuracy and the topology of the PDAG. For instance, in the example of Fig. 4, if s_1 is linked to s_2 and s_3, applying relation (1), would result in an unlinkability score of 0. Consequently, even if the second strategy is not preferable over the second, the second one may still jeopardize unlinkability completely. PDAG's unlinkability is thus tied to the smallest unlinkability score. We here define unlinkability for a PDAG according to its least unlinkable state.

Definition 10 (u-unlinkable PDAG). *A PDAG G is u-unlinkable according to an inferred linking relation $\tilde{L}$ if and only if all states in $\mathcal{S}$ are at least u-unlinkable according to $\tilde{L}$.*

As said before, inferred relation $\tilde{L}$ directly refers to heuristics from the literature, and the level of unlinkability of states and ultimately, the one of the PDAG is tied to the accuracy of this inference. However, one could imagine exploiting the different heuristics by involving unlinked states in a transaction vulnerable to $\tilde{L}$. This is, in a way, the principle of Coinjoin that counteracts the multi-input heuristic assumption, but we could also leverage the other heuristics and increase privacy up to their unlinkability score. In a way, the maximum unlinkability score of an inferred relation $\tilde{L}$ can be interpreted as the *maximal extractable privacy* (MEP). In the full version of the paper [40], we apply our definitions to various systems and measure their privacy guarantees. Meanwhile, we propose some extensions to our model in the following section.

4 Extensions

During the course of this paper, we adopted a deterministic view to describe and formalize the different privacy notions. Yet, the probabilistic standpoint might offer more flexibility and be more potent in reflecting extra information that the adversary gets from other sources (e.g., analysis from other layers, leaked databases or other attacks). Furthermore, it is probably more akin to capture machine learning-based models from the literature [20,29,30]. We here give a few leads to adapt our analysis to a probabilistic one by addressing each notion separately.

Untraceability. In Sect. 3, we describe the untraceability problem using a bipartite graph $G^* = (\mathcal{S}^* \dot{\cup} \mathcal{W}^*, \mathcal{E}^*)$. As part of the extension to a probabilistic model, we could assign weights to $\mathcal{E}^*$. Let $\omega : \mathcal{E}^* \rightarrow [0,1] \subseteq \mathbb{R}$ be a map that assigns a real number between 0 and 1 to each edge. The weight $\omega(e)$ of edge $e = (s, w)$ represents the probability that the underlying state s is consumed in the transaction involving witness w. This means that the sum of weights for edges of a single witness sums up to 1. The adversary is therefore able to assign a probability distribution based on some information (e.g., time or value-based correlations) or an inference of a machine learning model. $\mathcal{A}$ can then output a guess according to the weights of the edges.

Unlinkability. Although the inferred linking relation $\tilde{L}$ can already abstract a probabilistic classifier, we may go further by modifying this relation into a fuzzy relation μ such that $\mu : \mathcal{S} \times \mathcal{S} \rightarrow [0,1] \subseteq \mathbb{R}$. This relation gives a membership indicator between two states s_i and s_j, where 0 means that s_i and s_j are not related whereas 1 means they are. This would allow one state s to belong to two distinct clusters or to measure uncertainty in the clustering output.

5 Related Work

Regarding untraceability, Möser et al. [30] conduct an empirical analysis in the Monero blockchain. They identify two phenomena that jeopardize the untraceability guarantees of Monero. Möser [30] notice that 0-mixin transactions not only imply full traceability of such users, but also pose a privacy risk for other users that include poorly mixed outputs as mixins in subsequent transactions. Fortunately, since October 2018, the mixins input number is set globally to eleven [28], consequently alleviating this issue. Yet, Möser et al. [30] make a second observation regarding the sampling distribution of mixins inputs. The distribution of the sampling algorithm to choose the different mixins does not match the distribution of the coins spent in the transaction. The sampling distribution is uniform over the set of available transaction outputs, irrespective of the age of coins, whereas new coins are more likely to be spent directly. Old mixin coins can thus be considered decoys, reducing the true spender's anonymity set. Following this study, the Monero community also addressed this issue with a better sampling algorithm [13].

As for unlinkability, various works devise empirical behavioral patterns in transactions – called heuristics – to link addresses together. Nakamoto [31] describes the most widely studied heuristic in the original Bitcoin white paper. This heuristic allows linking to the same entity, multiple addresses used as input to the same transaction [31]. Studies find it to be relatively accurate [2] and able to reveal hidden clusters of addresses [27]. The Coinjoin mechanism mitigates the threat posed by this heuristic by having different users join their coins into a single transaction to blur the links between several of their addresses. Yet, Goldfeder et al. [15] devise a counter-heuristic to detect such a technique by considering multi-input-multi-output transactions as part of a Coinjoin mixing. Victor [37] devises heuristics for specific patterns in the Ethereum and Wang et al. [39] for cases where users utilize mixers to break links between their transactions. Finally, Kappos et al. [19] expose heuristics specifically for Zcash. Our work is also close to any literature review of deanonymization heuristics such as the systematization of knowledge of [9]. Furthermore, Meiklejohn and Orlandi [26] devise a metric similar to the unlinkability score for Coinjoin. We yet believe our framework and metric to be more generally applicable to any scheme and adversarial classifier.

Concerning graph-based analysis of privacy, Ober et al. [32] conduct an empirical study of crucial properties of the Bitcoin transaction graph, such as unlinkability and coin dormancy. They use the multi-input heuristic mentioned above to merge addresses that belong to the same entity. As a result, they discover an impressive amount of public keys that belong together while only considering a small subset of the full blockchain history. Androulaki et al. [2] carry out a similar study through a simulator that mimics the user of Bitcoin within a university. In this setting, the results of the heuristics can be compared to the ground truth. It turns out that almost 40% of user profiles can be recovered even though recommended privacy measures are applied. Also, Atzei et al. [3] develop a model to formally prove the fulfillment of some properties in the Bitcoin blockchain [3]. Conversely, Amarasinghe et al. [1] employ a common, universal framework to characterize the various aspects of anonymity provided by different blockchain implementations [1].

Finally, our work is also related to research on privacy notions in anonymous communication networks. Kuhn et al. [23] provide an in-depth analysis on the matter [23]. They study privacy notions in anonymous communication and their relations and devise a formal hierarchy to classify each of these properties. Moreover, Henry et al. [16] aim at formalizing anonymous blacklisting systems.

6 Conclusion

This work has tackled the definition of two main blockchain privacy notions: untraceability and unlinkability. To do so, we first extended the TDAG [7] to capture privacy-preserving blockchains (PDAG) and, secondly, gave consistent definitions to these notions according to PDAG's elements. This allowed us to model and compare blockchain implementations and unify literature results in the full version of this work [40]. During the course of this work, we also have discovered that the PDAG gives intuitive definitions to each notion and proposes a way to reason about them.

Acknowledgment. We thank the anonymous reviewers for helpful suggestions and feedback. Special thanks go to Jayamine Alupotha and Mathieu Gestin for interesting discussions about blockchain privacy and valuable comments. We are also grateful to David Lehnherr for fruitful advice and comments. Duc V. Le has been supported by a grant from Protocol Labs to the University of Bern.

References

1. Amarasinghe, N., Boyen, X., McKague, M.: The complex shape of anonymity in cryptocurrencies: case studies from a systematic approach. In: Borisov, N., Diaz, C. (eds.) FC 2021. LNCS, vol. 12674, pp. 205–225. Springer, Heidelberg (2021). https://doi.org/10.1007/978-3-662-64322-8_10
2. Androulaki, E., Karame, G.O., Roeschlin, M., Scherer, T., Capkun, S.: Evaluating user privacy in bitcoin. In: Sadeghi, A.-R. (ed.) FC 2013. LNCS, vol. 7859, pp. 34–51. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39884-1_4
3. Atzei, N., Bartoletti, M., Lande, S., Zunino, R.: A formal model of bitcoin transactions. In: Meiklejohn, S., Sako, K. (eds.) FC 2018. LNCS, vol. 10957, pp. 541–560. Springer, Heidelberg (2018). https://doi.org/10.1007/978-3-662-58387-6_29
4. Ben-Sasson, E., et al.: ZeroCash: decentralized anonymous payments from Bitcoin. In: IEEE Symposium on Security and Privacy, pp. 459–474. IEEE Computer Society (2014)
5. Biryukov, A., Khovratovich, D., Pustogarov, I.: Deanonymisation of clients in bitcoin P2P network. In: CCS, pp. 15–29. ACM (2014)
6. Biryukov, A., Pustogarov, I.: Bitcoin over tor isn't a good idea. In: IEEE Symposium on Security and Privacy, pp. 122–134. IEEE Computer Society (2015)
7. Cachin, C., De Caro, A., Moreno-Sanchez, P., Tackmann, B., Vukolić, M.: The transaction graph for modeling blockchain semantics. Cryptoeconomic Systems **1**(1) (2021). https://doi.org/10.21428/58320208.a12c57e6, preliminary version appears as Cryptology ePrint Archive, Report 2017/1070, 2017
8. Chaum, D.: The dining cryptographers problem: Unconditional sender and recipient untraceability. J. Cryptol. **1**(1), 65–75 (1988)

9. Deuber, D., Ronge, V., Rückert, C.: SoK: assumptions underlying cryptocurrency deanonymizations. Proc. Priv. Enhancing Technol. **2022**(3), 670–691 (2022)
10. Dulmage, A.L., Mendelsohn, N.S.: Coverings of bipartite graphs. Can. J. Math. **10**, 517–534 (1958). https://doi.org/10.4153/CJM-1958-052-0
11. Edman, M., Sivrikaya, F., Yener, B.: A combinatorial approach to measuring anonymity. In: ISI, pp. 356–363. IEEE (2007)
12. Egger, C., Lai, R.W.F., Ronge, V., Woo, I.K.Y., Yin, H.H.F.: On defeating graph analysis of anonymous transactions. Proc. Priv. Enhancing Technol. **2022**(3), 538–557 (2022)
13. Ehrenhofer, J.: Response to "an empirical analysis of traceability in the Monero blockchain", Version 2 (2018). https://www.getmonero.org/2018/03/29/response-to-an-empirical-analysis-of-traceability.html
14. Gierlichs, B., Troncoso, C., Díaz, C., Preneel, B., Verbauwhede, I.: Revisiting a combinatorial approach toward measuring anonymity. In: WPES, pp. 111–116. ACM (2008)
15. Goldfeder, S., Kalodner, H.A., Reisman, D., Narayanan, A.: When the cookie meets the blockchain: privacy risks of web payments via cryptocurrencies. Proc. Priv. Enhancing Technol. **2018**(4), 179–199 (2018)
16. Henry, R., Goldberg, I.: Formalizing anonymous blacklisting systems. In: IEEE Symposium on Security and Privacy, pp. 81–95. IEEE Computer Society (2011)
17. Hopwood, D., Bow, S., Hornby, T., Wilcox, N.: Zcash protocol specification (2021). https://zips.z.cash/protocol/protocol.pdf
18. Kalodner, H.A., Möser, M., Lee, K., Goldfeder, S., Plattner, M., Chator, A., Narayanan, A.: Blocksci: design and applications of a blockchain analysis platform. In: USENIX Security Symposium, pp. 2721–2738. USENIX Association (2020)
19. Kappos, G., Yousaf, H., Maller, M., Meiklejohn, S.: An empirical analysis of anonymity in Zcash. In: USENIX Security Symposium, pp. 463–477. USENIX Association (2018)
20. Kappos, G., Yousaf, H., Stütz, R., Rollet, S., Haslhofer, B., Meiklejohn, S.: How to peel a million: validating and expanding Bitcoin clusters. In: USENIX Security Symposium, pp. 2207–2223. USENIX Association (2022)
21. Koe, Alonso, K.M., Noether, S.: Zero to Monero. Online, 2nd edn. (2020). https://web.getmonero.org/library/Zero-to-Monero-2-0-0.pdf
22. Koshy, P., Koshy, D., McDaniel, P.: An analysis of anonymity in bitcoin using P2P network traffic. In: Christin, N., Safavi-Naini, R. (eds.) FC 2014. LNCS, vol. 8437, pp. 469–485. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45472-5_30
23. Kuhn, C., Beck, M., Schiffner, S., Jorswieck, E.A., Strufe, T.: On privacy notions in anonymous communication. Proc. Priv. Enhancing Technol. **2019**(2), 105–125 (2019)
24. Kumar, A., Fischer, C., Tople, S., Saxena, P.: A traceability analysis of Monero's blockchain. In: Foley, S.N., Gollmann, D., Snekkenes, E. (eds.) ESORICS 2017. LNCS, vol. 10493, pp. 153–173. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66399-9_9
25. Maxwell, G.: CoinJoin: Bitcoin privacy for the real world (2013). https://bitcointalk.org/index.php?topic=279249.0
26. Meiklejohn, S., Orlandi, C.: Privacy-enhancing overlays in bitcoin. In: Brenner, M., Christin, N., Johnson, B., Rohloff, K. (eds.) FC 2015. LNCS, vol. 8976, pp. 127–141. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48051-9_10

27. Meiklejohn, S., et al.: A fistful of Bitcoins: Characterizing payments among men with no names. In: login USENIX Magazine, vol. 38, no. 6 (2013)
28. Monero Project: Monero 0.13.0 "Beryllium Bullet" Release (2018). https://www.getmonero.org//2018/10/11/monero-0.13.0-released.html
29. Möser, M., Narayanan, A.: Resurrecting address clustering in Bitcoin. In: Eyal, I., Garay, J. (eds.) Financial Cryptography. LNCS, vol. 13411, pp. 386–403. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-18283-9_19
30. Möser, M., et al.: An empirical analysis of traceability in the monero blockchain. Proc. Priv. Enhancing Technol. **2018**(3), 143–163 (2018)
31. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008). https://bitcoin.org/bitcoin.pdf
32. Ober, M., Katzenbeisser, S., Hamacher, K.: Structure and anonymity of the Bitcoin transaction graph. Future Internet **5**(2), 237–250 (2013)
33. Pertsev, A., Semenov, R., Storm, R.: Tornado cash privacy solution (2019)
34. Pfitzmann, A., Köhntopp, M.: Anonymity, unobservability, and pseudonymity — A proposal for terminology. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, pp. 1–9. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44702-4_1
35. van Saberhagen, N.: CryptoNote (2013). https://bytecoin.org/old/whitepaper.pdf
36. Venkatakrishnan, S.B., Fanti, G., Viswanath, P.: Dandelion: redesigning the bitcoin network for anonymity. Proc. ACM Meas. Anal. Comput. Syst. **1**(1), 22:1–22:34 (2017)
37. Victor, F.: Address clustering heuristics for Ethereum. In: Bonneau, J., Heninger, N. (eds.) FC 2020. LNCS, vol. 12059, pp. 617–633. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51280-4_33
38. Vijayakumaran, S.: Analysis of cryptonote transaction graphs using the dulmage-mendelsohn decomposition. In: AFT. LIPIcs, vol. 282, pp. 28:1–28:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2023)
39. Wang, Z., et al.: On how zero-knowledge proof blockchain mixers improve, and worsen user privacy. In: WWW, pp. 2022–2032. ACM (2023)
40. Wicht, F.X., Wang, Z., Le, D.V., Cachin, C.: A transaction-level model for blockchain privacy. Cryptology ePrint Archive, Paper 2023/1902 (2023). https://eprint.iacr.org/2023/1902, https://eprint.iacr.org/2023/1902
41. Yu, Z., Au, M.H., Yu, J., Yang, R., Xu, Q., Lau, W.F.: New empirical traceability analysis of CryptoNote-style blockchains. In: Goldberg, I., Moore, T. (eds.) FC 2019. LNCS, vol. 11598, pp. 133–149. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32101-7_9