Evaluating User Privacy in Bitcoin

Elli Androulaki[1], Ghassan O. Karame[2], Marc Roeschlin[1],
Tobias Scherer[1], and Srdjan Capkun[1]

[1] ETH Zurich, 8092 Zuerich, Switzerland
elli.androulaki@inf.ethz.ch, romarc@student.ethz.ch,
schereto@student.ethz.ch, capkuns@inf.ethz.ch
[2] NEC Laboratories Europe, 69115 Heidelberg, Germany
ghassan.karame@neclab.eu

Abstract. Bitcoin is quickly emerging as a popular digital payment system. However, in spite of its reliance on pseudonyms, Bitcoin raises a number of privacy concerns due to the fact that all of the transactions that take place are publicly announced in the system.

In this paper, we investigate the privacy provisions in Bitcoin when it is used as a primary currency to support the daily transactions of individuals in a university setting. More specifically, we evaluate the privacy that is provided by Bitcoin *(i)* by analyzing the genuine Bitcoin system and *(ii)* through a simulator that faithfully mimics the use of Bitcoin within a university. In this setting, our results show that the profiles of almost 40% of the users can be, to a large extent, recovered even when users adopt privacy measures recommended by Bitcoin. To the best of our knowledge, this is the first work that comprehensively analyzes, and evaluates the privacy implications of Bitcoin.

1 Introduction

Bitcoin [6] is an emerging digital currency that is currently being integrated across a number of businesses [1] and exchange markets.

Bitcoin is a Proof-of-Work (PoW) based currency that allows users to generate digital coins by performing computations. Bitcoin users execute payments by digitally signing their transactions and are prevented from double-spending their coins (i.e., signing-over the same coin to two different users) through a distributed time-stamping service [6]. This service operates on top of the Bitcoin Peer-to-Peer (P2P) network and ensures that all transactions and their order of execution are available to the public. To strengthen the privacy of its users, Bitcoin users participate in transactions using pseudonyms—referred to as *Bitcoin addresses*. Generally, each user has hundreds of different Bitcoin addresses that are all stored and transparently managed by its client.

In spite of the reliance on pseudonyms, the public timestamping mechanism of Bitcoin raises serious concerns with respect to the privacy of users. In fact, given that Bitcoin transactions basically consist of a chain of digital signatures, the expenditure of individual coins can be publicly tracked [16].

In this work, we evaluate the privacy that is provided by Bitcoin when it is used to support the daily transactions of individuals in a university setting. This is achieved *(i)* by investigating the behavior of Bitcoin client and exploiting its properties, and *(ii)* through a novel simulator that mimics the use of Bitcoin as the primary currency within a university setting. Finally, we discuss possible measures that can be used to enhance the privacy of users in Bitcoin. To the best of our knowledge, this is the first work that analyzes, and evaluates the privacy provisions in Bitcoin. More specifically, our contributions in this paper can be summarized as follows:

- We adapt existing privacy notions to the Bitcoin context and we investigate the privacy-enhancing measures that are used in current Bitcoin implementations.
- We design and implement a simulator that faithfully emulates the functionality of Bitcoin. Our simulator also provides us with "ground truth" information that corresponds to the use of Bitcoin in a university setting.
- We investigate the privacy provisions of Bitcoin in a realistic university setting through our simulator. Our results show that the profiles of 40% of the university users can be constructed using behavior-based clustering techniques with 80% accuracy, even in the case when users manually transfer their Bitcoins among their addresses in an attempt to enhance their privacy.
- We discuss possible measures that can be used by Bitcoin developers to enhance the privacy of users in Bitcoin.

The remainder of this paper is organized as follows. In Section 2, we present a background on Bitcoin. In Section 3, we introduce metrics that we use to measure privacy in Bitcoin. In Section 4, we present our Bitcoin simulator and our evaluation results. In Section 5, we discuss the implications of our findings and we explore possible countermeasures for enhancing privacy in Bitcoin. In Section 6, we overview the related work and we conclude the paper in Section 7.

2 Background on Bitcoin

Bitcoin is a decentralized P2P payment system [6] that relies on PoW. Payments are performed by generating *transactions* that transfer Bitcoin coins (BTCs) between Bitcoin users. Users participate in transactions using pseudonyms—referred to as *Bitcoin addresses*. Generally, each user has hundreds of different Bitcoin addresses that are all stored and managed by its (digital) *wallet*. Each address is mapped through a transformation function to a unique public/private key pair. These keys are used to authorize the transfer of the ownership of BTCs among addresses.

Transactions: Users transfer coins (BTCs) to each other by issuing a transaction. A transaction is formed by digitally signing a hash of the transaction through which a BTC was acquired. Given that in Bitcoin there is one-to-one correspondence between signature public keys and addresses, a transaction taking place between two addresses a_{S} and a_{R} has the following form: $\tau(a_{\mathsf{S}} \rightarrow a_{\mathsf{R}}) = \{\mathrm{source}, B, a_{\mathsf{R}},\ \mathsf{SIG}_{\mathrm{sk}_{a_{\mathsf{S}}}}(\mathrm{source}, B, a_{\mathsf{R}})\}$. Here, $\mathsf{SIG}_{\mathrm{sk}_{a_{\mathsf{S}}}}$ is the signature using the private key $\mathrm{sk}_{a_{\mathsf{S}}}$ that corresponds to the public key associated with the a_{S}, B is the amount of BTCs transferred, and source is a reference to the most recent transaction that a_{S} acquired the B BTCs from. After their creation, Bitcoin transactions are released in the Bitcoin network. Once the validity of τ

is *confirmed* (as described later in this section), a_{R} can subsequently use this transaction as a reference to spend the acquired BTCs. Consequently, Bitcoin transactions form a public record and any user can verify the authenticity of a BTC by checking the chain of signatures of the transactions in which the BTC was involved.

In the case where a_{S} needs to spend a value that exceeds the maximum value of a BTC that it possesses, then its Bitcoin client will automatically combine a number of its BTCs as multiple inputs of the same outgoing transaction. We analyze the impact of "multi-input" transactions on the privacy of users in Bitcoin in Section 4.1. Figure 4 in Appendix A depicts an example of multiple-input transactions.

Shadow Addresses: In the current implementation of Bitcoin, a new address—the "shadow" address [1]—is automatically created and used to collect back the "change" that results from any transaction issued by the user. Besides the reliance on pseudonyms, shadow addresses constitute the only mechanism adopted by Bitcoin to strengthen the privacy of its users.

Confirmation of Transactions: As mentioned before, transactions are broadcasted in the Bitcoin network and are subject to validity checks by users in the system. Valid transactions are included by a special type of users, the *miners*, in Bitcoin *blocks* that are also broadcasted in the network. More specifically, to generate a new block, miners must find a nonce value that, when hashed with additional fields (i.e., the Merkle hash of all valid and received transactions, the hash of the previous block, and a timestamp), results in a value below a given threshold. If such a nonce is found, miners then include it in a block thus allowing any entity to verify the PoW. Upon successful block generation, a miner is granted a number of new BTCs. This provides an incentive for miners to continuously support Bitcoin. Table 2 in Appendix B shows the information included in Bitcoin block number 80,000 as reported in the Bitcoin block explorer [2]. The resulting block is forwarded to all users in the network, who can then check its correctness by verifying the hash computation. If the block is deemed to be "valid"[3], then the users append it to their previously accepted blocks, thus growing the Bitcoin block *chain*. Bitcoin relies on this mechanism to resist double-spending attacks; for malicious users to double-spend a BTC without being detected, they would not only have to redo all the work required to compute the block where that BTC was spent, but also they need to recompute all the subsequent blocks in the chain.

3 Modelling Privacy in Bitcoin

In this section, we introduce our adversarial model and we define a number of metrics that can be used to quantify privacy in Bitcoin.

3.1 Adversarial Model

We observe the public log of Bitcoin, denoted by pubLog, within a period of time $\varDelta t$. During this period, $\mathrm{n_U}$ users, $\mathrm{U} = \{u_1, u_2, \ldots, u_{\mathrm{n_U}}\}$, participate in pubLog through a

[3] That is, the block contains correctly formed transactions that have not been previously spent, and has a correct PoW.

set of $\mathrm{n_A}$ addresses: $\mathsf{A} = \{a_1, a_2, \ldots, a_{\mathrm{n_A}}\}$. We assume that within Δt, $\mathrm{n_T}$ transactions have taken place as follows: $\mathrm{T} = \{\tau_1(\mathsf{S}_1 \to \mathsf{R}_1), \ldots, \tau_{\mathrm{n_T}}(\mathsf{S}_{\mathrm{n_T}} \to \mathsf{R}_{\mathrm{n_T}})\}$, where $\tau_i(\mathsf{S}_i \to \mathsf{R}_i)$ denotes a transaction with (unique) ID number i and S_i and R_i denote the sets of senders' addresses and recipients' addresses, respectively.

We assume that the adversary $\mathcal{A}$ is motivated to acquire information about the addresses/transactions pertaining to all or a subset of Bitcoin users. As such, $\mathcal{A}$ does not only have access to pubLog, but is also *part of the Bitcoin system* and can also incur one or more transactions through Bitcoin. Furthermore, we assume that $\mathcal{A}$ can have access to the (public) addresses of some vendors along with (statistical) information such as the pricing of items or the number of their clients within a specified amount of time. We, however, assume that $\mathcal{A}$ is computationally bounded and as such cannot construct ill-formed Bitcoin blocks, double-spend confirmed transactions, or forge signatures, etc.

Throughout this paper, we consider the "privacy" measures adopted by existing Bitcoin clients. Namely, we assume that *(i)* new shadow addresses are used to collect change that results from issued transactions, *(ii)* users own many Bitcoin addresses, and *(iii)* users are encouraged to frequently change their addresses (by transferring some of their BTCs to newly created addresses); this conforms with the current practices adopted in Bitcoin.

3.2 Quantifying Privacy in Bitcoin

In this section, we introduce two different notions of Bitcoin privacy, activity unlinkability and profile indistinguishability and we provide metrics to appropriately quantify these notions.

Activity unlinkability refers to the fact that an adversary $\mathcal{A}$ should not be able to link two different addresses (address unlinkability) or transactions (transaction unlinkability) pertaining to a user of *her choice*. By design, if $\mathcal{A}$ can link two Bitcoin addresses to the same user, then she can also link all the transactions that these addresses participate in. Therefore, we focus our analysis on address unlinkability. On the other hand, *profile indistinguishability* refers to the (in-)ability of $\mathcal{A}$ to reconstruct the profiles of *all the users* that participate in pubLog. Profiles, here, consist of the set of addresses (address-based profiles) or set of transactions (transaction-based profiles) of Bitcoin users. As such, profile indistinguishability property is a stronger privacy notion than the activity unlinkability as it assesses the concealment of the profiles of *all users* in Bitcoin. Unlike the activity unlinkability case, here we also account for transaction-based profiles. This is due to the fact that address-based and transaction-based profiles are not equivalent when it comes to modeling user profiles in Bitcoin[4].

In the following, we provide definitions on both privacy notions, and we rely on these definitions to provide metrics for them. In particular, we define address unlinkability and profiling indistinguishability through the AddUnl and the $\mathrm{ProfInd}$ games, respectively. We quantify these notions by assessing the advantage of an adversary $\mathcal{A}$ in winning these games over an adversary who responds to all game challenges with

[4] An adversary can perform well in address-based profiling but not in transaction-based profiling: she may correctly profile addresses of users that are involved in few transactions and "miss-classify" few addresses who participate in many transactions.

random guesses, $\mathcal{A}^{\mathcal{R}}$. We assume that $\mathcal{A}$ has access to pubLog and that both $\mathcal{A}$ and $\mathcal{A}^{\mathcal{R}}$ have gathered (the same) a-priori knowledge $\mathcal{K}_{\mathcal{A}}$ with respect to correlations of a subset of addresses (i.e., whether these addresses belong to the same user or not).

Activity Unlinkability (Address Unlinkability): We construct the address unlinkability game in Bitcoin, AddUnl, as follows. $\mathcal{A}$ chooses an address $a_0 \in \mathsf{pubLog}$ chosen among the addresses that appear in pubLog and sends it to the challenger $\mathcal{C}$. If the owner of a_0 does not have any other Bitcoin address, then $\mathcal{A}$ wins. Otherwise, the challenger $\mathcal{C}$ randomly chooses a bit $\mathscr{b}$. If $\mathscr{b} = 1$, then $\mathcal{C}$ randomly chooses another address $a_1 \in \mathsf{pubLog}$ such that a_0, a_1 belong to the same user; otherwise, $\mathcal{C}$ randomly chooses a_1 such that the two addresses are owned by different users. The challenger sends $\langle a_0, a_1 \rangle$ to $\mathcal{A}$, who responds with her estimate $\mathscr{b}'$ on whether the two addresses belong to the same user. $\mathcal{A}$ wins the game if she answers correctly, i.e., $\mathscr{b} = \mathscr{b}'$. We say that Bitcoin satisfies address unlinkability if for all p.p.t. adversaries $\mathcal{A}$ and $\forall \langle a_0, a_1 \rangle$, $\mathcal{A}$ has only a negligible advantage over $\mathcal{A}^{\mathcal{R}}$ in winning, i.e., if:

$$\mathrm{Prob}[\mathscr{b}' \leftarrow \mathcal{A}(\mathsf{pubLog}, \mathcal{K}_{\mathcal{A}}) : \mathscr{b} = \mathscr{b}'] - \mathrm{Prob}[\mathscr{b}' \leftarrow \mathcal{A}^{R}(\mathcal{K}_{\mathcal{A}}) : \mathscr{b} = \mathscr{b}'] \leq \varepsilon,$$

where ε is negligible.

Quantifying Address Unlinkability: In Section 4.1, we show that due to inherent properties of the Bitcoin protocol and client, $\mathcal{A}$ can succeed with a considerable probability in winning the above AddUnl game. In what follows, we measure the *degree* to which Bitcoin addresses can be *linked* to the same user.

To do so, we express the estimate of $\mathcal{A}$ through an $\mathrm{n_A} \times \mathrm{n_A}$ matrix, $\mathrm{E_{link}}$, where $\mathrm{E_{link}}[i, j] = \{p_{i,j}\}_{i,j \in [1, \mathsf{n_A}]}$. That is, that for every address a_i, $\mathcal{A}$ assesses the probability $p_{i,j}$ with which a_i is owned by the same user as every other address a_j in pubLog. Note that $\mathrm{E_{link}}$ incorporates $\mathcal{K}_{\mathcal{A}}$, and any additional information that $\mathcal{A}$ could extract from pubLog (e.g., by means of clustering, statistical analysis, etc.). Similar to [15], we quantify the success of $\mathcal{A}$ in the AddUnl game as follows. Let $\mathrm{GT_{link}}$ denote the genuine address association matrix, i.e., $\mathrm{GT_{link}}[i, j] = 1$, if a_i and a_j are of the same user and $\mathrm{GT_{link}}[i, j] = 0$ otherwise for all $i, j \in [1, \mathrm{n_A}]$. For each address a_i we compute the error in $\mathcal{A}$'s estimate, i.e., the distance of $\mathrm{E_{link}}[i, *]$ from the genuine association of a_i with the rest of the addresses in pubLog, $||\mathrm{E_{link}}[i, *] - \mathrm{GT_{link}}[i, *]||$, where $||\cdot||$ denotes norm-L1 of the corresponding row-matrix. Thus, the success of $\mathcal{A}$ in AddUnl, $\mathrm{Succ}_{\mathcal{A}}$, can then be assessed through $\mathcal{A}$'s maximum error: $\max\limits_{\forall a_i \notin \mathcal{K}_{\mathcal{A}}} (||\mathrm{E_{link}}[i, *] - \mathrm{GT_{link}}[i, *]||)$.

Similarly, we represent the estimate of $\mathcal{A}^{\mathcal{R}}$ in the AddUnl game for all possible pairs of addresses by the $\mathrm{n_A} \times \mathrm{n_A}$ matrix $\mathrm{E^{\mathcal{R}}_{link}}$ which is constructed as follows. $\mathrm{E}^{\mathcal{R}}[i, j] = \pi_{i,j}$ if $\langle a_i, a_j \rangle \in \mathcal{K}_{\mathcal{A}}$, and $\mathrm{E^{\mathcal{R}}_{link}}[i, j] = \rho + (1 - \rho)\frac{1}{2}$ otherwise. Here, $\pi_{i,j}$ represents the probability that addresses a_i a_j correspond to the same user according to $\mathcal{K}_{\mathcal{A}}$, and ρ refers to the fraction of addresses in $\{\mathsf{pubLog}\text{-}\mathcal{K}_{\mathcal{A}}\}$ that cannot be associated to other addresses (i.e., when their owners have only one address)[5]. For pairs of addresses that are not included in $\mathcal{K}_{\mathcal{A}}$, this probability equals to $\rho + (1 - \rho)\frac{1}{2}$.

[5] In our experiments in Section 4, we assume that ρ is negligible since there are at least two addresses per user in Bitcoin (a real address and a "shadow" address).

Given this, we measure the degree of address unlinkability in Bitcoin against $\mathcal{A}$ by measuring address linkability, i.e., by evaluating the additional success that $\mathcal{A}$ can achieve from pubLog, when compared to $\mathcal{A}^{\mathcal{R}}$. We call this advantage $\mathrm{Link}_{\mathcal{A}}^{\mathrm{abs}}$: $\mathrm{Link}_{\mathcal{A}}^{\mathrm{abs}} = \mathrm{Succ}_{\mathcal{A}} - \mathrm{Succ}_{\mathcal{A}^{\mathcal{R}}}$, and its normalized version $\mathrm{Link}_{\mathcal{A}}$: $\mathrm{Link}_{\mathcal{A}} = \frac{\mathrm{Succ}_{\mathcal{A}} - \mathrm{Succ}_{\mathcal{A}^{\mathcal{R}}}}{\mathrm{Succ}_{\mathcal{A}^{\mathcal{R}}}}$.[6]

User Profile Indistinguishability: The profile indistinguishability property is a stronger privacy notion than the activity unlinkability as it refers to the concealment of all Bitcoin user profiles. Here, we require that an adversary is not able to group addresses or transactions corresponding to Bitcoin users correctly. As mentioned before, we account for the indistinguishability of the profiles of users, assuming address-based and transaction-based profiles.

We construct the ProfInd game as follows. Here, the challenger $\mathcal{C}$ sends to $\mathcal{A}$ the number of users $\mathrm{n_U}$ in pubLog-$\mathcal{K}_{\mathcal{A}}$. $\mathcal{A}$ responds with $\mathrm{n_U}$ (non-overlapping) sets of addresses (or transactions) and with $\mathrm{E_{prof}} = \{\mathsf{g}_i\}_{i=1}^{\mathrm{n_U}}$ representing the estimate on the profile of all users in the system. Let $\mathrm{GT_{prof}} = \{\mathrm{gt}_i\}_{i=1}^{\mathrm{n_U}}$ represent the genuine grouping of addresses (or transactions) to users. That is, $\mathrm{GT_{prof}} = \{a_{u_i}\}_{i=1}^{\mathrm{n_U}}$ for address-based profiles, and $\mathrm{GT_{prof}} = \{\tau_{u_i}\}_{i=1}^{\mathrm{n_U}}$, for transaction-based profiles, where a_{u_i} and τ_{u_i} represent the sets of addresses and transactions respectively of user u_i. Clearly, $\mathcal{A}$ wins if she guesses correctly, i.e., if $\mathrm{E_{prof}} \equiv \mathrm{GT_{prof}}$.

We say that a system satisfies the profile indistinguishability property if there is no p.p.t. adversary $\mathcal{A}$ who wins the ProfInd game with better probability than $\mathcal{A}^{\mathcal{R}}$, i.e.:

$$\forall \text{ p.p.t. } \mathcal{A}: \quad \mathrm{Prob}[\mathrm{E_{prof}} \leftarrow \mathcal{A}(\mathsf{pubLog}, \mathrm{n_U}) : \mathrm{E_{prof}} \equiv \mathrm{GT_{prof}}] - \mathrm{Prob}[\mathrm{E}_{\mathrm{prof}}^{\mathcal{R}} \leftarrow \mathcal{A}^{\mathcal{R}}(\mathrm{n_U}) : \mathrm{E}_{\mathrm{prof}}^{\mathcal{R}} \equiv \mathrm{GT_{prof}}] \leq \varepsilon.$$

Quantifying User Profile Indistinguishability: We now proceed to quantify $\mathcal{A}$'s advantage in winning the ProfInd game by measuring the similarity of $\mathcal{A}$'s estimate $\mathrm{E_{prof}}$ from the genuine grouping of profiles $\mathrm{GT_{prof}}$, $\mathrm{Sim}(\mathrm{E_{prof}}, \mathrm{GT_{prof}})$, where the similarity function Sim ranges in $[0, 1]$. As in address unlinkability, we measure profile indistinguishability against $\mathcal{A}$ by measuring the degree of profile distinguishability that $\mathcal{A}$ can achieve, i.e., we assess the advantage of $\mathcal{A}$ in approximating $\mathrm{GT_{prof}}$ over $\mathcal{A}^{\mathcal{R}}$ by $\mathrm{Prof}_{\mathcal{A}} = \mathrm{Sim}(\mathrm{E_{prof}}, \mathrm{GT_{prof}}) - \mathrm{Sim}(\mathrm{E}_{\mathrm{prof}}^{\mathcal{R}}, \mathrm{GT_{prof}})$.[7]

We quantify $\mathrm{Sim}(\mathrm{E_{prof}}, \mathrm{GT_{prof}})$ and $\mathrm{Prof}_{\mathcal{A}}$ by relying on two commonly used entropy based distance metrics, namely: the *Normalized Mutual Information* (NMI) and the *Adjusted Mutual Information*, (AMI). NMI assesses the similarity of two groupings of the same items (in our case, $\mathrm{E_{prof}}$ and $\mathrm{GT_{prof}}$), and takes higher values (1) the more identical the groupings are [19,20]. On the other hand, AMI assesses the advantage of $\mathcal{A}$ in winning the ProfInd game. More specifically, given the two groupings $\mathrm{E_{prof}}$ and $\mathrm{GT_{prof}}$ AMI approaches 0 when $\mathrm{E_{prof}}$ is close to random assignment of addresses/transactions to groups, i.e., $\mathrm{E}_{\mathrm{prof}}^{\mathcal{R}}$, and is 1 when $\mathrm{E_{prof}}$ matches $\mathrm{GT_{prof}}$ [19,20].

[6] We say that Bitcoin satisfies μ-address unlinkability if $\forall$ p.p.t. algorithms $\mathcal{A}$ and the corresponding $\mathcal{A}^{\mathcal{R}}$: $\mathrm{Prob}[\mathrm{E_{link}} \leftarrow \mathcal{A}(\mathsf{pubLog}, \mathcal{K}_{\mathcal{A}}), \mathrm{E}_{\mathrm{link}}^{\mathcal{R}} \leftarrow \mathcal{A}^{\mathcal{R}}(\mathcal{K}_{\mathcal{A}}) : \mathrm{Link}_{\mathcal{A}} \geq 1 - \mu] \leq \varepsilon$.

[7] We say that a system offers μ-profile indistinguishability, and we write μ-ProfInd, if $\forall$ p.p.t. $\mathcal{A}$: $\mathrm{Prob}[\mathrm{E_{prof}} \leftarrow \mathcal{A}(\mathsf{pubLog}, \mathrm{n_U}), \mathrm{E}^{\mathcal{R}} \leftarrow \mathcal{A}^{\mathcal{R}}(\mathrm{n_U}) : \mathrm{Prof}_{\mathcal{A}} \geq 1 - \mu] \leq \varepsilon$.

Assuming address-based profiles, NMI and AMI are computed as follows:

$$\mathrm{NMI} = \frac{\mathcal{I}(\mathrm{E}_{\mathrm{prof}}, \mathrm{GT}_{\mathrm{prof}})}{\max(\mathrm{H}(\mathrm{E}_{\mathrm{prof}}), \mathrm{H}(\mathrm{GT}_{\mathrm{prof}}))}, \quad \mathrm{AMI} = \frac{\mathcal{I}(\mathrm{E}_{\mathrm{prof}}, \mathrm{GT}_{\mathrm{prof}}) - \mathcal{E}}{\max(\mathrm{H}(\mathrm{E}_{\mathrm{prof}}), \mathrm{H}(\mathrm{GT}_{\mathrm{prof}})) - \mathcal{E}},$$

where:

$\mathcal{I}(\mathrm{E}_{\mathrm{prof}}, \mathrm{GT}_{\mathrm{prof}}) = \sum_{i=1}^{\mathrm{n_U}} \sum_{j=1}^{\mathrm{n_U}} \frac{n_{(i,j)}}{\mathrm{n_A}} \log(\frac{n_{(i,j)} \cdot \mathrm{n_A}}{n_{(i,*)} n_{(*,j)}}),$

$\mathrm{H}(\mathrm{E}_{\mathrm{prof}}) = -\sum_{i=1}^{\mathrm{n_U}} \frac{n_{(i,*)}}{\mathrm{n_A}} \log(\frac{n_{(i,*)}}{\mathrm{n_A}}), \quad \mathrm{H}(\mathrm{GT}_{\mathrm{prof}}) = -\sum_{j=1}^{\mathrm{n_U}} \frac{n_{(*,j)}}{\mathrm{n_A}} \log(\frac{n_{(*,j)}}{\mathrm{n_A}}),$

$\mathcal{E} = \sum_{i=1}^{\mathrm{n_U}} \sum_{j=1}^{\mathrm{n_U}} \sum_{n \in \mathcal{M}} \frac{n}{\mathrm{n_A}} \log(\frac{\mathrm{n_A} n}{n_{(i,*)} n_{(*,j)}}) \frac{n_{(i,*)}! n_{(*,j)}! (\mathrm{n_A} - n_{(i,*)})! (\mathrm{n_A} - n_{(*,j)})!}{\mathrm{n_A}! (n_{(i,*)} - n)! (n_{(*,j)} - n)! (\mathrm{n_A} - n_{(i,*)} - n_{(*,j)} - n)!}.$

Here, $\mathrm{n_A}$ is the number of Bitcoin addresses, $n_{(i,j)}$ is the number of u_i's addresses, which are assigned to group g_j, $n_{(i,*)}$ and $n_{(*,j)}$ are the number of addresses of u_i and g_j respectively. $\mathcal{E}$ reflects the *expected* mutual information between $\mathrm{GT}_{\mathrm{prof}}$ and a random grouping of addresses ($\mathrm{E}_{\mathrm{prof}}^{\mathcal{R}}$). Also, $\mathcal{M} = [\max(n_{(i,*)} + n_{(*,j)} - \mathrm{n_A}, 0), \min(n_{(i,*)}, n_{*,j})]$. Similar calculations can be derived to compute NMI and AMI for transaction-based profiles.

Remark 1. Although NMI and AMI represent sufficiently well the success of $\mathcal{A}$ in profiling all the users in the system, they do not measure the success of the adversary on the profiling of a particular user u_i. In Section 4.4, we measure the success of $\mathcal{A}$ in profiling u_i by assessing the maximum similarity of the addresses (or transactions) of each user u_i with each adversarial cluster g_j i.e., $\max_{\forall j}(\mathrm{Sim}(a_{u_i}, \mathrm{g}_j))$.

4 Evaluating Privacy in Bitcoin

In what follows, we show how the adversary, given pubLog, can gather some knowledge about Bitcoin users by exploiting the properties of existing Bitcoin client implementations. We then evaluate, by means of our Bitcoin simulator, the success of the adversary in the aforementioned privacy games given this knowledge.

4.1 Exploiting Existing Bitcoin Client Implementations

Current Bitcoin client implementations enable $\mathcal{A}$ to link a fraction of Bitcoin addresses that belong to the same user.

Heuristic I—Multi-input Transactions: As mentioned earlier, multi-input transactions occur when u wishes to perform a payment, and the payment amount exceeds the value of each of the available BTCs in u's wallet. In fact, existing Bitcoin clients choose a set of BTCs from u's wallet (such that their aggregate value matches the payment) and perform the payment through multi-input transactions. It is therefore straightforward to conclude that if these BTCs are owned by different addresses, then the input addresses belong to the same user. Note that, currently, Bitcoin clients do not provide support for different users to participate in a single transaction; to achieve this, users would have to modify the Bitcoin client implementation themselves.

Heuristic II—"Shadow" Addresses: As mentioned earlier, Bitcoin generates a new address, the "shadow" address [1], on which each sender can collect back the "change".

This mechanism suggests a distinguisher for shadow addresses. Namely, in the case when a Bitcoin transaction has two output addresses, a_{R_n}, a_{R_o}, such that a_{R_n} is a new address (i.e., an address that has never appeared in pubLog before), and a_{R_o} corresponds to an old address (an address that has appeared previously in pubLog), we can safely assume that a_{R_n} constitutes a shadow address for a_i. Note that, in the current Bitcoin implementation, users rarely issue transactions to two different users.

Evaluating Heuristics I and II: In what follows, we evaluate the implications of these heuristics on the user-privacy in Bitcoin. Given the absence of recent statistical data on the number of Bitcoin users, we rely on an estimate of the number of Bitcoin users performed in September 2011; at that time, Bitcoin users amounted to 60,000 users [1]. We then created a C++ parser that parses the first 140,000 blocks (till August 2011) in the Bitcoin block explorer [2].

Our C++ parser extracts all the addresses in each block and categorizes them in clusters of Generic Addresses, GAs, given the two aforementioned heuristics. The parser then outputs a list of addresses organized in different GAs. Our results show that there were 1,632,648 unique addresses in the first 140,000 blocks. Given Heuristic I, we could classify these addresses into 1,069,699 distinct GAs. Given Heuristic II, this number decreases to 693,051 GAs; this corresponds to grouping approximately 58% of Bitcoin addresses with an average of 11.55 addresses per GA. This clearly shows that $\mathcal{A}$'s advantage in the AddUnl game is considerable given Heuristics I and II.

4.2 Behavior-Based Analysis

Besides exploiting current Bitcoin implementations, $\mathcal{A}$ could also make use of behavior-based clustering techniques, such as K-Means (KMC), and the Hierarchical Agglomerative Clustering (HAC) algorithms. Let U be the set of users populating Bitcoin and $(\mathsf{GA}_1, \ldots, \mathsf{GA}_{\mathrm{n_{GA}}})$ denote the GAs that $\mathcal{A}$ has obtained by applying the two aforementioned heuristics on pubLog, respectively. Given this, the goal of $\mathcal{A}$ is to output a group of clusters of addresses $\mathrm{E_{prof}} = \{\mathsf{g}_1, \ldots, \mathsf{g}_{n_\mathsf{U}}\}$ such that $\mathrm{E_{prof}}$ best approximates U. Since each GA is owned by exactly one user, the estimate on the assignment of each GA_i can be modeled by a variable z_i such that $z_i = k$, if and only if, GA_i belongs to g_k.

In fact, HAC assumes that initially each GA represents a separate cluster ($\{z_i = i\}_{i=1}^{\mathrm{n_{GA}}}$) and computes similarity values for each pair of clusters. Clusters with higher similarity value are combined into a single cluster and cluster-to-cluster similarity values are re-computed. The process continues until the number of created clusters equals the number of users $\mathrm{n_U}$. KMC is then initialized using the output of HAC and assumes that each user is represented by the center of each cluster. The algorithm iterates assignments of GAs to clusters and aims at minimizing the overall distance of GAs to the center of the cluster they have been assigned to. The centers of the clusters and the GA-to-cluster distances are re-computed in each round.

In our implementation (cf. Section 4.3), we represent each transaction that appears within a GA using: *(i)* the time at which the transaction took place, *(ii)* the indexes of the different GAs that appear within the transaction (as senders or recipients), and *(iii)* the values of the BTCs spent by the transaction. Let τ_x denote the set of transactions of GA_x. The degree of similarity between GA_i and GA_j, denoted by $\mathsf{Sim}^{\mathsf{hac}}(\mathsf{GA}_i, \mathsf{GA}_j)$, is then represented by the cosine similarity of lists τ_i and τ_j, i.e., $\mathsf{Sim}^{\mathsf{hac}}(\mathsf{GA}_i, \mathsf{GA}_j) = \frac{\sum_{\forall \tau \in \tau_i \cap \tau_j} (f_{(\tau,i)} \cdot f_{(\tau,j)})}{\|\tau_i\| \cdot \|\tau_j\|}$, where $f_{(\tau,i)}$, $f_{(\tau,j)}$ are the occurrences of item τ in lists τ_i and τ_j respectively, and $\|X\|$ denotes the norm II of vector X. Given this, the resulting distance metric in KMC is $\mathsf{Dist}^{\mathsf{kmc}}(\mathsf{GA}_i, \mathsf{g}_k) = \frac{2}{1+\mathsf{Sim}^{\mathsf{hac}}(\mathsf{GA}_i, \mathsf{g}_k)} - 1$.

Our implementation also accounts for constraints that are posed by real-world settings. Namely, since users cannot be physically located in two different places at the same time, they cannot participate in two different (physical) exchanges of goods at the same time. To account for this case (cf. Section 4.4), we apply different weighting for similarity of GAs who participate in transactions concurrently.

4.3 Simulating the use of Bitcoin in a University

To evaluate the success of $\mathcal{A}$ in the AddUnl and the $\mathrm{ProfInd}$ games, we simulate a realistic case of using Bitcoin in the Department of Computer Science in ETH Zurich. Here, we assume that the shops located around the university also accept BTCs as a currency. Given the lack of details and statistics about the current use of Bitcoin, this was one of the few "workable" uses of Bitcoin that we could try to accurately model and through which we could evaluate the advantage of $\mathcal{A}$ in the system.

Experimental Setup: To evaluate the privacy implications of using Bitcoin in a university environment, we devised the setup shown in Figure 1. Our Bitcoin simulator takes an XML configurations file as input and outputs: *(i)* a log that details the events that were simulated, the "ground truth", as well as *(ii)* the resulting simulated public Bitcoin log, pubLog. The XML configurations file contains all the necessary parameters to run the simulator. These include the number of users, the number of miners, the simulation time, the difficulty in block generation, as well as usage configurations for creating user profiles and Bitcoin sellers/buyers. Further details about our simulator can be found in Appendix D. As shown in Figure 1, the outputs of our simulator are used to evaluate $\mathcal{A}$'s success. In fact, once the simulations terminate, a Perl-based parser uses the simulated Bitcoin block as input and pre-classifies the simulated addresses into GAs according to Heuristics I and II. The resulting "pre-filtered output" is then fed into our clustering algorithms, the HAC and the KMC algorithms (both implemented in C). The output of these algorithms is then compared using another Perl-based script with the "ground truth" generated by the simulator in order to compute the success of $\mathcal{A}$ in the AddUnl and the $\mathrm{ProfInd}$ games.

We tuned our simulator to match a real-world scenario that reflects the actual behavior of the staff and student members of a Computer Science Department of a university in the Fall 2012 semester. In our setting, we consider a variable number of users, 5.2% of which are "Professors", 42.0% are "Staff" and the remaining 52.8% are "Students". We consider a total of 6 events, each having several options: lunching/dining

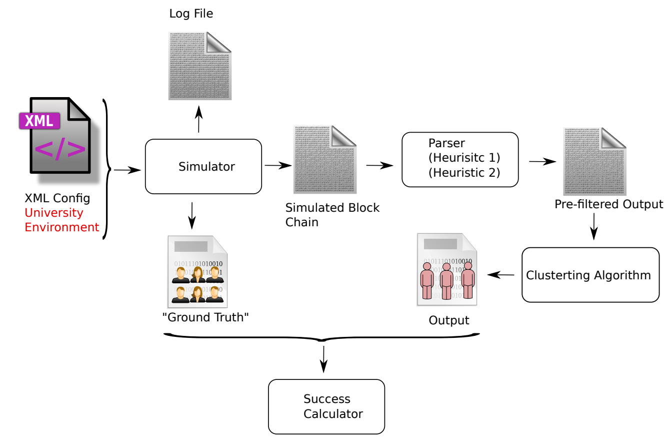


Fig. 1. Experimental setup used throughout our simulations. The outputs of our Bitcoin simulator are pre-filtered according to Heuristics I and II and then fed as input to our clustering algorithm. The clustering result is then compared with the "ground truth" that is emulated by our simulator.

(12 options), buying groceries (2 options), buying from vending machines (4 options), online shopping (5 options), purchasing books (2 options), and performing barters with other users, totalling 25 different Bitcoin vendors present in our system. For each user, we assign a probability that the user undergoes each of the possible options of each event. These probabilities are assigned according to the "category" of the user; that is, if the user is a "Professor", then it is more likely that he/she would eat lunches at more expensive restaurants, when compared to the case where the user falls in the "Student" category. For each event, we specify in the XML configuration the following parameters: the frequency of the event, and the price range per option of the event. Note that each option is assigned a rating that would reflect its popularity. The probability of performing an option is interpolated from the frequency of occurrence of the event per week, and from the rating of the option. To ensure a large variety of profiles in our user base, we specify in the XML configuration a minimum and maximum value for the frequency, rating and price fields. These bounds depend on the category of the user, the event and option in question (c.f. Appendix C). At the start of our experiments, users originally have few (< 10) Bitcoin addresses; as they issue new transactions, new (shadow) addresses are created in their wallets. In the XML file, we also model the behavior of "privacy-aware" users. We assume that these users create new Bitcoin addresses in their wallets and send some of their BTCs from their old to their new addresses.

4.4 Experimental Results

Throughout our experiments, we emulated two different scenarios for each simulation round. In the first scenario denoted by "Partial Knowledge", we assume that $\mathcal{A}$ is aware of the location/service of all Bitcoin vendors and as such can distinguish whether a

		100 (50%)	200 (0%)	200 (50%)	200 (100%)	400 (50%)
$\mathbf{Link}_{\mathcal{A}}$		0.91 ±0.01	0.90 ±0.01	0.91 ±0.01	0.92 ±0.01	0.93 ±0.01
$\mathbf{Prof}^{a}_{\mathcal{A}}$	NMI	0.76 ± 0.01	0.87 ± 0.01	0.79 ± 0.01	0.70 ± 0.01	0.80 ± 0.01
	AMI	0.75 ± 0.01	0.86 ± 0.01	0.77 ± 0.01	0.68 ± 0.01	0.77 ± 0.01
$\mathbf{Prof}^{\tau}_{\mathcal{A}}$	NMI	0.68 ± 0.01	0.73 ± 0.02	0.70 ± 0.01	0.65 ± 0.01	0.72 ± 0.01
	AMI	0.67 ± 0.01	0.72 ± 0.01	0.69 ± 0.01	0.63 ± 0.01	0.70 ± 0.01

(a) Results in the "Partial Knowledge" scenario.

		100 (50%)	200 (0%)	200 (50%)	200 (100%)	400 (50%)
$\mathbf{Link}_{\mathcal{A}}$		0.90 ±0.01	0.90 ±0.01	0.91 ±0.01	0.92 ±0.01	0.93 ±0.01
$\mathbf{Prof}^{a}_{\mathcal{A}}$	NMI	0.79 ± 0.01	0.89 ± 0.01	0.79 ± 0.01	0.71 ± 0.02	0.80 ± 0.01
	AMI	0.78 ± 0.02	0.88 ± 0.01	0.78 ± 0.02	0.69 ± 0.02	0.78 ± 0.01
$\mathbf{Prof}^{\tau}_{\mathcal{A}}$	NMI	0.69 ± 0.01	0.73 ± 0.03	0.69 ± 0.03	0.65 ± 0.01	0.72 ± 0.01
	AMI	0.68 ± 0.01	0.72 ± 0.01	0.68 ± 0.03	0.63 ± 0.01	0.70 ± 0.01

(b) Results in the "No Knowledge" scenario.

Table 1. Behavior-based clustering results in the "Partial Knowledge" and "No Knowledge" scenarios. A column entitled X (Y%) denotes an experiment featuring X users among which $Y\%$ are privacy-aware. Each data point in our plots is averaged over five rounds of experiments; we also present the corresponding 95% confidence intervals (shown after the "$\pm$" sign).

transaction was performed in exchange of a physical good. In this case, we include the vendors's addresses in the prior knowledge of $\mathcal{A}$ when computing $\mathrm{Adv}_{\mathrm{unl}}$; we also assume that $\mathcal{A}$ can tune the clustering algorithm to take into account that the same user performing this transaction cannot appear in other transactions that takes place at the same time. This case emulates the realistic setting where $\mathcal{A}$ can extract a subset of the addresses owned by geographically co-located Bitcoin users/vendors from the overall public Bitcoin log; for example, $\mathcal{A}$ can extract from the Bitcoin log all the addresses that interact with a known address of a vendor located within the university environment. In the second scenario denoted by "No Knowledge", we consider the case where $\mathcal{A}$ does not know the location or service of the vendors, and as such does not have any prior knowledge, but assumes that up to 10% of the transactions are performed in exchange of goods delivered over the Internet.

Given this setup, we evaluate the metrics $\mathrm{Link}_{\mathcal{A}}$, $\mathrm{Prof}^{a}_{\mathcal{A}}$ (for address-based profiles), and $\mathrm{Prof}^{\tau}_{\mathcal{A}}$ (for transaction-based profiles) with respect to *(i)* the fraction of "privacy-aware" users and *(ii)* the number of users $\mathrm{n_U}$. By privacy-aware users, we refer to users that manually generate new Bitcoin addresses (following a configuration in the XML file) to enhance their privacy in the system. Table 1 depicts our findings. Our results show that both the "Partial Knowledge" and the "No Knowledge" configurations exhibited comparable results.

In the first round of experiments, we evaluate the success of $\mathcal{A}$ with respect to the fraction of "privacy-aware" users. More specifically, we run our clustering and privacy evaluation algorithm in a setting featuring 200 users, among which $0\%, 50\%$, and 100% of the users are privacy-aware. Table 1 shows $\mathrm{Link}_{\mathcal{A}}$, $\mathrm{Prof}^{a}_{\mathcal{A}}$, and $\mathrm{Prof}^{\tau}_{\mathcal{A}}$ with respect to the fractions of privacy-aware users. Here, we use a normalized version on $\mathrm{Link}_{\mathcal{A}}$.

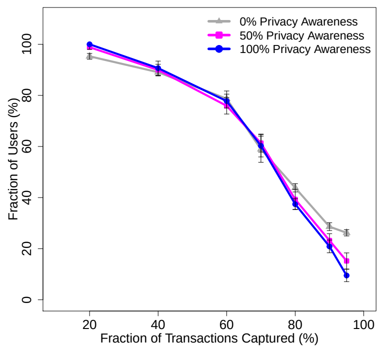


Fig. 2. Fraction of transactions captured by our clustering algorithms in the "No Knowledge" case.

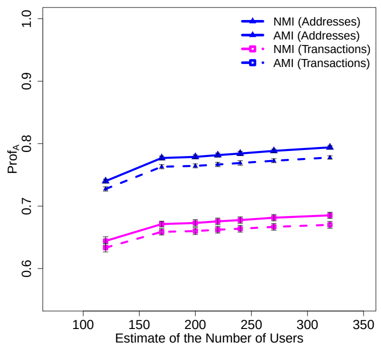


Fig. 3. The case where $\mathcal{A}$ cannot accurately estimate $\mathrm{n_U}$. We assume the 'Partial Knowledge" case where $\mathrm{n_U} = 200$.

In both configurations, the advantage of $\mathcal{A}$ in AddUnl game is only negligible affected by the fraction of the privacy aware users in the system. More specifically, we can see that our clustering algorithms outperform $\mathcal{A}^{\mathcal{R}}$ by almost 90%. On the contrary, $\mathrm{Prof}^{a}_{\mathcal{A}}$ and $\mathrm{Prof}^{\tau}_{\mathcal{A}}$ show a better dependency on the fraction of privacy aware users. When none of users in the system are privacy-aware, the performance of our clustering algorithms is high. In particular, in both configurations $\mathrm{Prof}_{\mathcal{A}^{a}}$ (NMI and AMI for addresses) range within $0.87 - 0.89$, while $\mathrm{Prof}^{\tau}_{\mathcal{A}}$ (NMI and AMI for transactions) are 0.73. However, as the fraction of the privacy aware users increases, the performance of $\mathcal{A}$ drops and results in $\mathrm{Prof}^{\tau}\ \mathcal{A}$ and $\mathrm{Prof}^{a}_{\mathcal{A}}$ of 0.70 and 0.63 respectively. This mischief can be explained by the fact that privacy aware users add noise to the Bitcoin log. However, the fact that AMI values remain consistently far from 0 and close to 1 indicates that $\mathcal{A}$ performs much better than $\mathcal{A}^{\mathcal{R}}$ and that the estimate chosen by $\mathcal{A}$ is close to the genuine assignment of users to clusters.

Figure 2 depicts the overall fraction of user profiles (measured by means of the similarity of transactions appearing in a user's wallet and the corresponding cluster as discussed in Section 3.2) that are captured by $\mathcal{A}$. Our results show that in the case when $\mathrm{n_U} = 200$ users with 0% privacy awareness, almost 42% of the users have their preferences captured with 80% accuracy. In the case featuring 100% privacy-aware users, this fraction drops to 35% of the users whose profile was correctly clustered with an accuracy of at least 80%. We therefore conclude that the privacy of users in Bitcoin can be compromised, even if users manually create new addresses in order to enhance their privacy in the system.

Furthermore, our results show that $\mathcal{A}$'s advantage over $\mathcal{A}^{\mathcal{R}}$ is not significantly affected by the number of participant users in the case of address unlinkability. $\mathrm{Prof}^{a}_{\mathcal{A}}$ and $\mathrm{Prof}^{\tau}_{\mathcal{A}}$ increase from 0.76 and 0.68 to 0.80 and 0.72 respectively as the number of users increases from 100 to 400. This is mostly because when the number of users increases, the assignments of addresses (or transactions) into groups of users ($\mathcal{A}^{\mathcal{R}}$) performs worst. In Figure 3, we evaluate the case where $\mathcal{A}$ does not have an accurate estimate of the number of users in the university setting. Our findings show that even if $\mathcal{A}$'s estimate of the number of users is not accurate, the privacy of a considerable fraction of users is still compromised.

5 Discussion

So far, our analysis focused on the current implementation of Bitcoin when used in a university setting. Note that our results provide rather an upper bound to privacy in the studied university setting than an accurate assessment of privacy in Bitcoin in generic settings. In what follows, we analyze the implications of our findings in generic implementations/uses of Bitcoin.

Evading Heuristic I: We start by showing that Heuristic I cannot be easily evaded in any future implementation of Bitcoin without compromising the basic operation of Bitcoin. Indeed, the combination of multiple inputs ensures that coins with "large" values can be recreated from existing smaller BTCs; this prevents the value of coins from being continuously deprecated following every issued transaction until the value of these coins reaches the minimum amount. At that point in time, the only way for Bitcoin users to issue transactions without combining their previous coins is to perform multiple transactions with single-input, one coin at a time. This clearly does not solve the problem since this process can still be tracked by the adversary (transactions will be linked by time). Another alternative would be for Bitcoin developers to provide support for different users to transparently participate in a single transaction. While this would increase the use-cases where Bitcoin finds applicability (e.g., performing contracts [1]), the collaborative construction of transactions by different users is unlikely to be predominately used in the network.

Evading Heuristic II: Similarly, it is easy to see that evading Heuristic II can only decrease the privacy of Bitcoin users. That is, if "shadow" addresses were not utilized, and the change of coins is simply put back in the sender's address then users' activities can be traced in an easier way. We point out that Heuristic II results in the dispersion of the coins among several addresses of the users. This makes the privacy leakage due to Heuristic I even more considerable. One possible way to "evade" Heuristic II would be for Bitcoin users to *(i)* first divide the coins according to the required payment in one transaction and *(ii)* then make the payment with zero change at a random point later in time. Another possible solution to "harden" the reliance on Heuristic II would be for Bitcoin to support Mutli-Input Multi-Output (MIMO) transactions.

Implications to Generic Uses of Bitcoin: We argue that our findings are not specific to the studied university setting and apply to other generic-uses of Bitcoin. More specifically, we believe that the adversary can, to a large extent, extract from the public Bitcoin log a small set of addresses that correspond to geographically co-located users; the adversary can subsequently run our clustering algorithms on the extracted set of addresses. For instance, if Bitcoin were to be used across shops, then the adversary can extract all the addresses that interacted with a specific set of Bitcoin vendors that are located within a specific geographic region. The larger is the number of addresses of (physical) vendors that the adversary knows, the more complete is the view of the adversary of the geographic sub-network.

The reliance on third-party trusted entities (e.g., Bitcoin banks, Bitcoin Anonymizers, FlexCoin [3]) emerges as one of the few workable solutions to increase the privacy of Bitcoin clients. These entities can hide the direct relationship between the inputs and outputs of a transaction within a sufficiently large anonymity set. However, this solution comes at odds with the main intuition behind the complete decentralization in Bitcoin.

6 Related Work

In [5], Elias investigates the legal aspects of privacy in Bitcoin. In [7], Babaioff *et al.* address the lack of incentives for Bitcoin users to include recently announced transactions in a block, while in [4], Syed *et al.* propose a user-friendly technique for managing Bitcoin wallets. In [14], Karame *et al.* thoroughly investigate double-spending attacks in Bitcoin and show that double-spending fast payments in Bitcoin can be performed in spite of the measures recommended by Bitcoin developers. Clark *et al.* [11] propose the use of the Bitcoin PoW to construct verifiable commitment schemes. Reid and Harrigan [16] analyze the flow Bitcoin transactions in a small part of Bitcoin log, and show that external information, i.e., publicly announced addresses, can be used to link identities and organizations to some transactions.

ECash [8–10] and anonymous credit cards were the first attempts to define privacy-preserving transactions. Privacy in ECash consists of user anonymity and transaction unlinkability; by relying on a set of cryptographic primitives ECash ensures that payments pertaining to the same user cannot be linked to each other or to the payer, provided that the latter does not misbehave. In [15], Pfitzmann *et al.* define unlinkability and privacy in pseudonymous systems. Dwork [13] defined differential privacy and quantified the information leakage from the query access of individuals. In Section 3.2, we adapt Dwork's generic differential privacy notion to our Bitcoin privacy notions. Finally, in [18], Shokri *et al.* quantify location privacy by assessing the error of the adversarial estimate from the ground truth. In [12, 17] the authors further introduce entropy-based metrics to assess the communication privacy in anonymous networks.

7 Conclusion

In this paper, we evaluated the privacy provisions in Bitcoin when it is used as a primary currency to support the daily transactions of individuals in a university setting.

Our findings show that the current measures adopted by Bitcoin are not enough to protect the privacy of users if Bitcoin were to be used as a digital currency in a university setting. More specifically, we rely on a simulator that mimics the use of Bitcoin in a realistic university setting. Our results show that if Bitcoin is used as a digital currency to support the daily transactions of users in a typical university environment, then behavior-based clustering techniques can unveil, to a large extent, the profiles of 40% of Bitcoin users, even if these users try to enhance their privacy by manually creating new addresses. Finally, we discussed a number of solutions that could be integrated by Bitcoin developers to enhance the privacy of users.

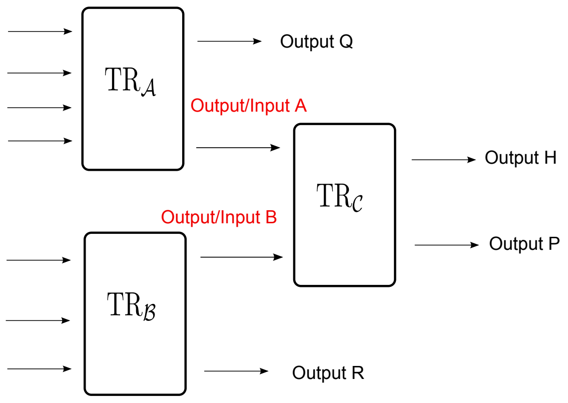
Acknowledgements

The authors would like to acknowledge the reviewers for their valuable comments and feedback.

References

1. Bitcoin – Wikipedia, Available from `https://en.bitcoin.it/wiki`.
2. Bitcoin Block Explorer, Available from `http://blockexplorer.com/`.
3. Flexcoin –The Bitcoin Bank, Available from `http://www.flexcoin.com/`.
4. Bitcoin Gateway, A Peer-to-peer Bitcoin Vault and Payment Network, 2011. Available from `http://arimaa.com/bitcoin/`.
5. Bitcoin: Tempering the Digital Ring of Gyges or Implausible Pecuniary Privacy, 2011. Available from `http://ssrn.com/abstract=1937769ordoi:10.2139/ssrn.1937769`.
6. Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System.
7. M. Babaioff, S. Dobzinski, S. Oren, and A. Zohar. On Bitcoin and Red Balloons. *CoRR*, 2011.
8. S. Brands. Electronic Cash on the Internet. In *Proceedings of the Symposium on the Network and Distributed System Security*, 1995.
9. J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact E-Cash. In *Proceedings of Advances in Cryptology - EUROCRYPT*, 2005.
10. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Proceedings on Advances in Cryptology - CRYPTO*, 1990. `http://dl.acm.org/citation.cfm?id=88314.88969`.
11. J. Clark and A. Essex. (Short Paper) CommitCoin: Carbon Dating Commitments with Bitcoin. In *Proceedings of Financial Cryptography and Data Security*, 2012.
12. C. Diaz, S. Seys, J. Claessens, and B. Preneel. Towards measuring anonymity. In R. Dingledine and P. Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*, April 2002.
13. C. Dwork. Differential privacy: a survey of results. In *Proceedings of the 5th international conference on Theory and applications of models of computation*, TAMC'08, 2008.
14. G. Karame, E. Androulaki, and S. Capkun. Double-Spending Fast Payments in Bitcoin. In *Proceedings of ACM CCS*, 2012.
15. A. Pfitzmann and M. Hansen. Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management ï£¡ A Consolidated Proposal for Terminology. *Fachterminologie Datenschutz und Datensicherheit*, pages 111–144, 2008.
16. F. Reid and M. Harrigan. An Analysis of Anonymity in the Bitcoin System. *CoRR*, 2011. `http://www.bibsonomy.org/bibtex/257d6640d03ae4a5668ef8b32656461eb/dblp`.
17. A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In R. Dingledine and P. Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.
18. R. Shokri, G. Theodorakopoulos, J. L. Boudec, and J. P. Hubaux. Quantifying location privacy. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2011.
19. N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *26th Annual International Conference on Machine Learning (ICML)*, 2009.
20. N. X. Vinh, J. Epps, and J. Bailey. Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. *Journal of Machine Learning Research*, 2010.

A Multi-Input Transactions



$\mathrm{TR}_{\mathcal{A}}$
Output Q
Output/Input A
$\mathrm{TR}_{\mathcal{C}}$
Output H
Output P
Output/Input B
$\mathrm{TR}_{\mathcal{B}}$
Output R

Fig. 4. Example of referencing input/output in transactions.

B Bitcoin Block Explorer

Hash: 000000000043a8c0fd1d6f726790caa2a406010d19efd2780db27bdbbd93baf6
Previous block: 00000000001937917bd2caba204bb1aa530ec1de9d0f6736e5d85d96da9c8bba
Next block: 00000000000036312a44ab7711afa46f475913fbd9727cf508ed4af3bc933d16
Time: 2010-09-16 05:03:47
Difficulty: 712.884864
Transactions: 2
textbfMerkle root: 8fb300e3fdb6f30a4c67233b997f99fdd518b968b9a3fd65857bfe78b2600719
Nonce: 1462756097

Input/Previous Output	**Source & Amount**	**Recipient & Amount**
N/A	Generation: 50 + 0 total fees	Generation: 50 + 0 total fees
f5d8ee39a430...:0	1JBSCVF6VM6QjFZyTnbpLjoCJ...: 50	16ro3Jptwo4asSevZnsRX6vf..: 50

Table 2. Example Block of Bitcoin. The block contains 2 transactions, one of which awards the miner with 50 BTCs.

C Example Configuration File

Listing 1 Example of XML configuration parameters for a "lunch" event with 12 options corresponding to the profile of a "Professor".

```
<!– lunch, eventid="0" refers to event with id="0" from above –>
<ProfileEvent eventid="0" minFreqPerWeek="5" maxFreqPerWeek="5" >
<Store storeid="0" maxPref="1" minPref="0" maxPrice="10.0" minPrice="8.0" />
<Store storeid="1" maxPref="1" minPref="0" maxPrice="13.0" minPrice="10.0" />
<Store storeid="2" maxPref="1" minPref="0" maxPrice="15.0" minPrice="13.0" />
<Store storeid="3" maxPref="4" minPref="2" maxPrice="25.0" minPrice="15" />
<Store storeid="4" maxPref="2" minPref="0" maxPrice="20.0" minPrice="15.0" />
<Store storeid="5" maxPref="2" minPref="0" maxPrice="17.0" minPrice="12.0" />
<Store storeid="6" maxPref="0.5" minPref="0" maxPrice="20.0" minPrice="8.0" />
<Store storeid="7" maxPref="0.5" minPref="0" maxPrice="10.0" minPrice="7.5" />
<Store storeid="8" maxPref="4" minPref="2" maxPrice="25.0" minPrice="15" />
<Store storeid="9" maxPref="0.5" minPref="0" maxPrice="25.0" minPrice="10" />
<Store storeid="10" maxPref="0.5" minPref="0" maxPrice="10.0" minPrice="5.0" />
<Store storeid="11" maxPref="3" minPref="1" maxPrice="12.0" minPrice="9.0" />
</ProfileEvent>
```

D Bitcoin Simulator

Our simulator is round-based; in each simulation round (defined as a "weekly timestepping" interval), events are added to a priority queue with a probability dictated by the configuration file. These events correspond to one of the following operations:

– *Issue a new transaction:* Users might issue new Bitcoin transactions whose time, value, beneficiary and purpose stem from the XML configurations file. The process of transaction issuance in our simulator fully mimics its counterpart in the genuine Bitcoin system.
– *Generate a new Bitcoin address:* Here, in addition to the automatically generated addresses in Bitcoin (cf. Section 3), "privacy-aware" users might decide to generate a number of new addresses to further "obfuscate" their usage of Bitcoin.

Conforming with the current use of Bitcoin, only few users in our setting were miners (i.e., mining is currently mostly performed by dedicated mining pools).

Our Bitcoin simulator abstracts away network delays, congestion, jitter, etc.. We also assume that all transactions in the system are well-formed and we do not model transaction fees that are incurred in the network. While malformed transactions and double-spending attempts [14] can be indeed witnessed in the genuine Bitcoin system, we believe that malicious behavior in Bitcoin is orthogonal to our privacy investigation—which explains the reason why we did not model such a misbehavior in our simulator. Moreover, our simulator relies on a variant greedy algorithm that closely approximates the genuine algorithm used in Bitcoin. Note that while the distribution of generated blocks in our simulator matches that in Bitcoin [14][8], we increased the average time between the generation of successive blocks in the simulator to better cope with simulated network dynamics[9].

[8] It was shown in [14] that the block generation in Bitcoin follows a shifted geometric distribution with parameter 0.19.

[9] Throughout our experiments, we considered that new blocks were generated every 20 minutes on average.

E Captured Transactions w.r.t. $\mathrm{n_U}$

Figure 5 shows the impact of the number of users $\mathrm{n_U}$ on the performance of our profiling algorithms.

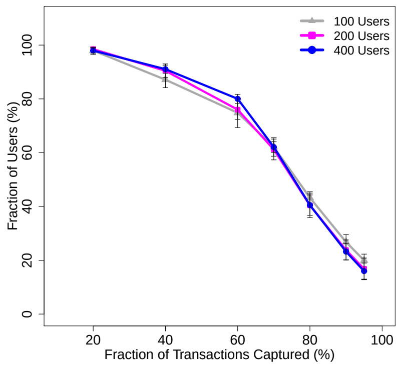


Fig. 5. Fraction of captured transactions with respect to $\mathrm{n_U}$. Here, we consider the "No Knowledge" case where 50% of the users are privacy-aware.

Our results show that the fraction of users whose transactions are correctly captured by our algorithms is not considerably affected by the number of users in the system. For instance, the fraction of users whose profiles were captured with an accuracy of at least 80% is approximately 40% when $\mathrm{n_U} = 100, 200, 400$.

Note that this conforms with our previous observations in Section 4.4; as shown in Figure 3, $\mathrm{Prof}_{\mathcal{A}}$ is only slightly affected as $\mathrm{n_U}$ increases.