Blockchain is Watching You: Profiling and Deanonymizing Ethereum Users[*]

Ferenc Béres[1,2], István A. Seres[2], András A. Benczúr[1,3], Mikerah Quintyne-Collins[4]
[1]Institute for Computer Science and Control (SZTAKI), Budapest, Hungary
[2]Eötvös Loránd University, Budapest, Hungary
[3]Széchenyi University, Győr, Hungary
[4]HashCloack Inc., Toronto, Canada

May 29, 2020

Abstract

Ethereum is the largest public blockchain by usage. It applies an account-based model, which is inferior to Bitcoin's unspent transaction output model from a privacy perspective. As the account-based models for blockchains force address reuse, we show how transaction graphs and other quasi-identifiers of users such as time-of-day activity, transaction fees, and transaction graph analysis can be used to reveal some account owners. To the best of our knowledge, we are the first to propose and implement Ethereum user profiling techniques based on user quasi-identifiers.

Due to the privacy shortcomings of the account-based model, recently several privacy-enhancing overlays have been deployed on Ethereum, such as non-custodial, trustless coin mixers and confidential transactions. We assess the strengths and weaknesses of the existing privacy-enhancing solutions and quantitatively assess the privacy guarantees of the Etherum blockchain and ENS. We identify several heuristics as well as profiling and deanonymization techniques against some popular and emerging privacy-enhancing tools.

1 Introduction

The narrative around cryptocurrency privacy provisions has dramatically changed since the inception of Bitcoin [34]. Initially many, especially criminals, thought Bitcoin and other cryptocurrencies provide privacy to hide their illicit business activities [14]. The first extensive study about Bitcoin's privacy provisions was done by Meiklejohn et al [32], in which they provide several powerful heuristics allowing one to cluster Bitcoin addresses. The revelation of Bitcoin's privacy shortcomings spurred the creation and implementation of many privacy-enhancing overlays for Bitcoin [55, 9, 47, 64]. As of today, several Bitcoin wallets, e.g. Wasabi and Samourai wallets, provide privacy-enhancing solutions to their users.

Previous work has focused on assessing the privacy guarantees provided by several UTXO-based (unspent transaction output) cryptocurrencies, such as Bitcoin [2, 32], Monero [13, 33, 7] or Zcash [5, 6, 7, 24, 53].

However, perhaps surprisingly, until today there were no similar studies on account-based cryptocurrency privacy provisions. Therefore in this work, we put forth the problem of studying the privacy guarantees of Ethereum's account-based model. Assessing and understanding the privacy guarantees of cryptocurrencies is essential as the lack of financial privacy is detrimental to most cryptocurrency use cases. Furthermore, there are state-sponsored companies and other entities, e.g. Chainalysis [38], performing large-scale deanonymization tasks on cryptocurrency users.

In contrast to the UTXO-model, many cryptocurrencies apply the account model. In an account-based cryptocurrency, users store their assets not in UTXOs but in accounts. Already in the Bitcoin whitepaper, Nakamoto [34] suggested that "a new key pair should be used for each transaction to keep them from being linked to a common owner." Despite this suggestion, account-based cryptocurrency users tend to use only a handful of addresses for their activities. The account-based model reinforces address-reuse on the protocol level. This behavior practically makes the account-based cryptocurrencies inferior to UTXO-based currencies from a privacy point of view.

Previously, several works have identified the privacy shortcomings of the account-based model, specifically in Ethereum. Those works have proposed trustless coin mixers [31, 49, 51] and confidential transactions [61, 10, 12]. However, until recently, none of these schemes has been deployed on Ethereum. Even today, Ethereum's privacy-enhancing overlays are still in a nascent, immature phase especially in comparison with Bitcoin's well-established coin mixer scene.

Our contributions:

- We identify and apply several quasi-identifiers stemming from address reuse (time-of-day activity, transaction fee, transaction graph), which allow us to profile and deanonymize Ethereum users.
- We establish several heuristics to decrease the privacy

[*]Support from the project 2018-1.2.1-NKP-00008: Exploring the Mathematical Foundations of Artificial Intelligence of the Hungarian Government.

guarantees of non-custodial mixers on Ethereum.

- We describe a version of the Danaan-gift attack [6] applicable in Ethereum.
- We collect and analyze a wide source of Etherum related data, including Ethereum name service, Etherscan blockchain explorer, Tornado Cash mixer contracts, and Twitter.
- We release the collected data as well as our source code for further research[1].

The rest of this paper is organized as follows. In Section 2, we review related work. In Section 3, we give a brief background on Ethereum and its inner workings. In Section 4, we describe our collected data. In Section 5, we overview the literature on quantifying deanonymization methods and propose our evaluation metrics. In Section 6 and 7, we describe our main methods to pair Ethereum accounts that belong to the same user and link Tornado deposits and withdrawals. We describe a variant of the Danaan-gift attack in Section 8. Finally, we conclude our paper in Section 9 by pointing out promising directions for future work.

2 Related Work

First results on Ethereum deanonymization [27] attempted to directly apply both on-chain and peer-to-peer (P2P) Bitcoin deanonymization techniques. The starting point of our work is that common deanonymization methods for Bitcoin may not be applicable to Ethereum due to differences in Ethereum's P2P stack and account-based model.

The relevant body of more recent literature takes two different approaches. The first analyzes and clusters Ethereum smart contracts with unsupervised clustering techniques [39]. Kiffer et al. [25] assert a large degree of code reuse which might be problematic in case of vulnerable and buggy contracts.

The second and more relevant branch of literature analyzes and clusters addresses in Ethereum. A crude and initial analysis had been made by Payette et al., who clusters the Ethereum address space into only four different clusters [41]. More interestingly Friedhelm Victor proposes address clustering techniques based on participation in certain airdrops and ICOs [56]. These techniques are indeed powerful, however, they do not generalize well as it assumes participation in certain on-chain events. Our techniques are more general and are applicable to all Ethereum addresses. Victor et al. gave a comprehensive measurement study of Ethereum's ERC-20 token networks, which further facilitates the deanonymization of ERC-20 token holders [57].

A completely different and unique approach is taken by [29], which uses stylometry to deanonymize smart contract authors and their respective accounts. The work had been used to identify scams on Ethereum.

[1]https://github.com/ferencberes/ethereum-privacy

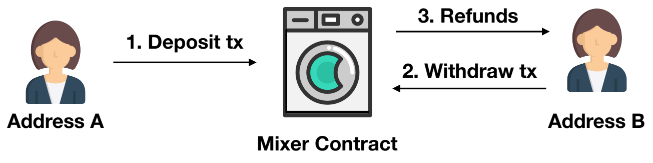

Figure 1: Schematic depiction of non-custodial mixers on Ethereum

3 Background

In this section we provide some background on cryptocurrency privacy-enhancing technologies. We provide more elementary preliminaries on Ethereum and its applied gas mechanism in Appendix A.

3.1 Non-custodial mixers

Coin mixing is a prevalent technique to enhance transaction privacy of cryptocurrency users. Coin mixers may be custodial or non-custodial. In case of custodial mixing, a user wishing to enhance her privacy sends her "tainted" coins to a trusted party, who in return sends back "clean" coins after some timeout. This solution is not satisfactory as the user does not retain ownership of her coins during the course of mixing. Hence, the trusted mixing party might just steal funds, as it already happened with custodial mixers [32].

Motivated by the drawbacks of custodial mixers, recently there have been several proposed non-custodial mixers in the literature [31, 60, 49, 51]. The recurring theme of non-custodial mixers is to replace the trusted mixing party with a publicly verifiable transparent smart contract or with secure multi-party computation (MPC). Non-custodial mixing is a two-step procedure. First, users wishing to mix coins deposit equal amounts of ether or other tokens into a mixer contract from an address $\mathcal{A}$, see Figure 1. After some user-defined time interval, they can withdraw their deposited coins with a withdraw transaction to a fresh address $\mathcal{B}$. In the withdraw transaction, users can prove to the mixer contract that they deposited without revealing which deposit transaction was issued by them by using one of several available cryptographic techniques, including ring signatures [31], verifiable shuffles [49], threshold signatures [51], and zkSNARKs [60].

3.2 Ethereum Name Service

Ethereum Name Service (ENS) is a distributed, open, and extensible naming system based on the Ethereum blockchain. In spirit it is similar to the well-known Domain Name Service (DNS). However, in ENS the registry is implemented in Ethereum smart contracts[2], hence it is resistant to DoS attacks and data tampering. Like DNS, ENS operates on a system of dot-separated hierarchical names called

[2]See: https://docs.ens.domains

Source	Total	At least 5 sent txs	Used as ground truth pairs
Twitter	1364	1260	129
Tornado Cash	2361	1618	*189
Humanity-Dao	695	602	n/a
All	4259	3321	318

Table 1: Number of Ethereum addresses collected from three different sources. *Tornado ground truth pairs are only heuristically identified, see Section 7.1.

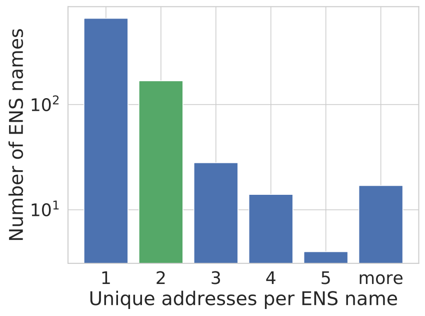

Figure 2: Unique address count of ENS names collected from Twitter. Most of the ENS names in our collection are linked to a single Ethereum address, while some entities use multiple accounts. In Section 6, we use ENS names with exactly two unique addresses **(green)** to measure the performance of different profiling techniques.

domains, with the owner of a domain having full control over subdomains. ENS maps human-readable names like `alice.eth` to machine-readable identifiers such as Ethereum addresses. Therefore, ENS provides a more user-friendly way of transferring assets on Ethereum, where users can use ENS names (`alice.eth`) as recipient addresses instead of the error-prone hexadecimal Ethereum addresses.

4 Data collection

We collected addresses presumably belonging to regular users and not automatic (trader or exchange) bots from the following publicly available data sources. **Twitter:** By using the Twitter API, we were able to collect 890 ENS names included in Twitter profiles, and discover the connected Ethereum addresses, see Figure 2. **Humanity DAO:**[3] A human registry of Ethereum users, which can include a Twitter handle in addition to the Ethereum address. **Tornado Cash mixer contracts:** We collected all Ethereum addresses that issued or received transactions from Tornado Cash mixers. Table 1 shows the total number of addresses collected from each data source as well as addresses with at least 5 sent transactions. We note that there are overlaps between the three address groups.

[3] See: https://www.humanitydao.org/humans

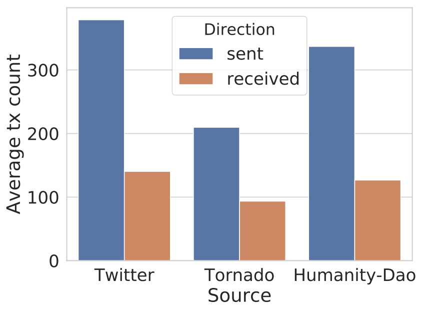

Figure 3: Average number of transactions sent or received by the addresses of each data source. Tornado accounts have less transactions as the service has only recently been launched.

By using the Etherscan blockchain explorer API, we collected 1,155,188 transactions sent or received by the addresses in our collection. The final transaction graph contains 159,339 addresses. The transactions span from 2015-07-30 till 2020-04-04. The distribution of the number of transactions sent by each Ethereum address follows a power-law distribution. Figure 3 shows the average number of transactions sent and received in the three data sources. Addresses collected from Twitter and Humanity DAO have similar behavior, while Tornado accounts have fewer transactions since Tornado Cash has only recently been launched.

Finally, using the Etherscan Label Word Cloud, we manually collected service category labels (e.g. exchange, gambling, stablecoins) related to the most popular addresses in our data set. We summarize the fraction of ENS names in our collection that interacted with the given services in Figure 4. We observed that the publicly revealed ENS names already expose sensitive activities such as gambling and adult services. Therefore, users should avoid sensitive activities on addresses easily linkable to their public identities, such as ENS name or their Twitter handle.

5 Evaluation measures

In this paper, we propose deanonymization methods for pairing Etherum accounts of the same user (Section 6), Tornado deposits and withdrawals (Section 7), and fingerprinting accounts (Section 8). To establish an appropriate measure for evaluating our methods, we face the diversity and complexity of estimates of the adversary's success to breach privacy. In the literature, the adversary's output takes the form of a posterior probability distribution, see the survey [58].

The simplest metrics consider the success rate of a deanonymizing adversary. Metrics such as accuracy, coverage, fraction of correctly identified nodes [3, 37, 35] are applicable only when the attack has the potential to exactly identify a significant part of the network.

Exact identification is an overly ambitious goal in our experiments, which aim to use very limited public information to rank candidate pairs and quantify the leaked information

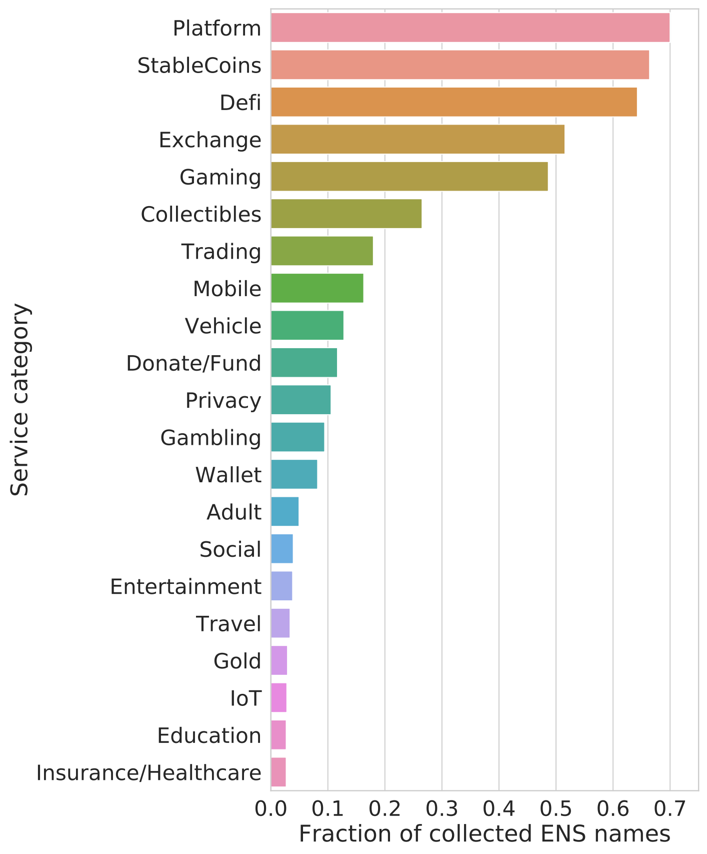

Figure 4: Fraction of ENS names (collected from Twitter) that interacted with the given service topics. Popular services within the categories are shown in Figure 5.

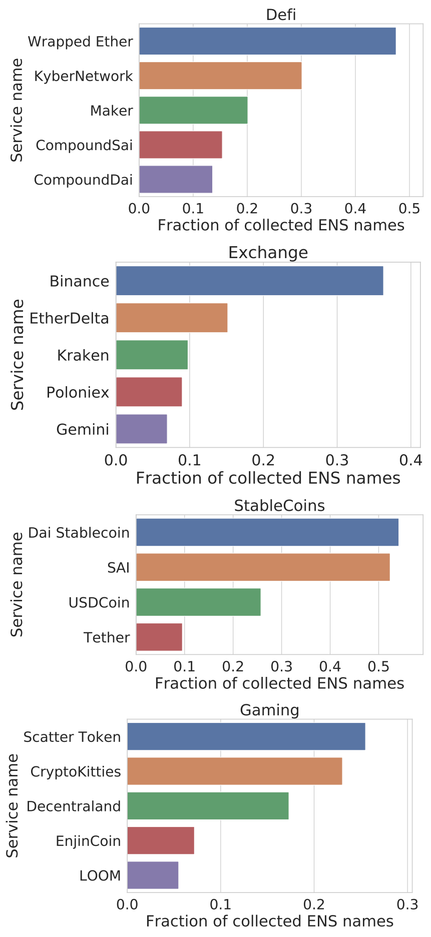

Figure 5: Most popular services within the Defi, Exchange, Stablecoins and Gaming categories.

as risk for a potential systematic deanonymization attack. For this reason, we quantify non-exact matches, since even though our deanonymizing tools might not exactly find a mixing address, they can radically reduce the anonymity set, which is still harmful to privacy. We want to quantify the information leaked from network structure, time-of-day activity, and gas price usage to assess the implications for the *future privacy* [36] of the account owners.

In our first two deanonymization experiments, our algorithms will return a ranked list of candidate pairs for each account in our testing set. Based on the ranked list, we propose a simple metric, the **average rank** of the target in the output.

Recent results consider deanonymization as a classification task and use AUC for evaluation [30]. In our experiments, we will compute AUC by the following claim:

Lemma 5.1. Consider a set of accounts a, each with a set of candidate pairs $c(a)$ such that exactly one in $c(a)$ is the correct pair of a. Let an algorithm return a ranked list of all sets $c(a)$. The AUC of this algorithm is equal to the average of $r(a)/|c(a)|$ over all a, where $r(a)$ is the rank of the correct pair of a in the output.

Proof. Follows since AUC is the probability that a randomly selected correct record pair is ranked higher than another incorrect one [23]. $\square$

Finally, we consider evaluation by variants of entropy, which quantify privacy loss by the number of bits of additional information needed to identify a node. Defining entropy is difficult in our case for two reasons. First, our algorithms provide a ranked list and not a probability distribution. Second, for Tornado mixer deanonymization, the anonymity set size is dynamic, as users can freely deposit anytime they wish, hence increasing the anonymity set size.

In the literature, entropy based evaluation considers the a priori knowledge without a deanonymization method and the a posteriori knowledge after applying one [50]. Several papers compute the entropy of the a posteriori knowledge [50, 15, 36], however they assume that the deanonymizer outputs a probability distribution of the candidate records [36].

The information the attacker has learned with the attack can be expressed as the difference of the a priori and a posteriori entropy. We call this difference the **entropy gain**, denoted as $\text{gain}(n, p)$ where n and p are the anonymity set size and probability distribution, respectively. The a priori entropy of the target record is typically the base-2 logarithm of the a priori anonymity set size. The problem with varying a priori anonymity set size is that while correctly selecting ten candidate users from a pool of a million is a great achievement, the same entropy of $\log_2(10)$ is achieved without deanonymization if the initial pool size, for example in a low-utilization mixer, is only 10. We note that in [15], the authors also divide the entropy gain to normalize the value.

Next, we describe a new method to infer the a posteriori distribution given varying a priori knowledge and appropriately normalize with respect to the a priori entropy. More precisely, first we give a heuristic argument that the a priori anonymity set size has little effect on the entropy gain, and hence we can compare and average across different measurements. In the formula below, given an a priori anonymity set size $2n$ vs. n, we compare the entropy gain of the same distribution p, $\text{gain}(2n, p) - \text{gain}(n, p)$. In the formula below, p_i denotes the probability $p([(i-1)/(2n), i/(2n)])$.

$$\begin{aligned}\text{gain}(2n, p) &= \log_2(2n) + \sum_{i=1}^{2n} p_i \log_2(p_i);\\ \text{gain}(n, p) &= \log_2(n) + \sum_{i=1}^{n} (p_{2i-1} + p_{2i}) \log_2(p_{2i-1} + p_{2i}).\end{aligned}$$

Since $\log_2(2n) - \log_2(n) = 1 = \sum_i p_i$, we may group the terms to obtain the difference in the entropy gain as the sum for $1 \le i \le n$ of

$$p_{2i-1} \log_2 \left(\frac{2p_{2i-1}}{p_{2i-1} + p_{2i}} \right) + p_{2i} \log_2 \left(\frac{2p_{2i}}{p_{2i-1} + p_{2i}} \right), \qquad (1)$$

which can be bounded from above by using $\log x < x - 1$ as

$$\frac{(p_{2i-1} - p_{2i})^2}{p_{2i-1} + p_{2i}}. \qquad (2)$$

If the probability distribution is smooth with little density changes in a neighborhood, the above value is very small. For example, the value is small if p_i is monotonic in i, which at least approximately holds in our experiments.

Based on the above argument, we may infer an **empirical probability distribution** of the candidates ranked by an algorithm. For each a priori size n and rank r for the ground truth pair of a target record, we define the distribution $P(n, r)$ to be uniform in $[(r-1)/n, r/n]$, and 0 elsewhere, in accordance with formula (1). The empirical probability distribution of an algorithm will be the average of $P(n, r)$ over all the output of the algorithm. In the discussion, we will use the **entropy gain** of the above empirical probability distribution to quantify the deanonymization power of our algorithms.

6 Finding Ethereum accounts of the same user

In this section, we introduce our approach to identify pairs of Ethereum accounts that belong to the same user. In our measurements, we investigated three quasi-identifiers of the account owner: the active time of the day, the gas price selection, and the location in the Ethereum transaction graph.

We evaluate our methods by using the set of address pairs in our collection that belong to the same name in the Ethereum Name Service (ENS), see Figure 2. We consider 129 ENS names with exactly two Ethereum addresses

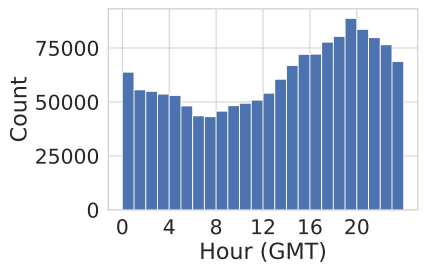

Figure 6: Time-of-day distribution of Ethereum transactions

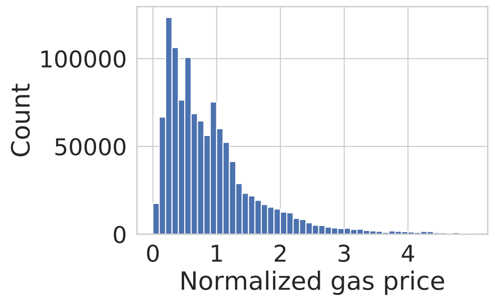

Figure 7: Normalized gas price distribution of Ethereum transactions. Outliers above 5 are omitted.

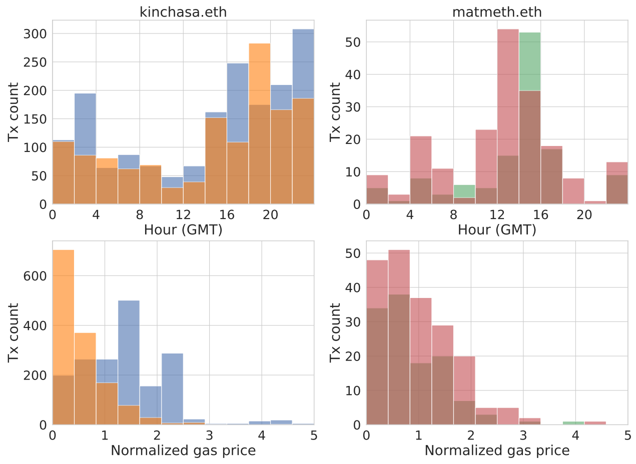

Figure 8: Time-of-day and normalized gas price profiles for two ENS names with a pair of addresses each. Both the time-of-day and gas price selection are similar in case of matmeth.eth addresses while the addresses of kinchase.eth have different gas price profiles. Addresses are denoted by different colors.

to avoid the possible validation bias caused by ENS names with more than two addresses. We also note that Ethereum addresses connected to multiple ENS names were excluded from our experiments.

6.1 Time-of-day transaction activity

Ethereum blockchain transaction timestamps reveal the daily activity patterns of the account owner, see Figure 6. In the top row of Figure 8, we show time-of-day profiles for two ENS names that are active in different time zones.

Given the set of timestamps, an account is represented by the vector including the mean, median and standard deviation, as well as the histogram divided into b_{hour} bins.

6.2 Gas price distribution

Ethereum transactions also contain the gas price, which is usually automatically set by wallet softwares. Users rarely change this setting manually. Most wallet user interfaces offer three levels of gas prices, slow, average, and fast where the fast gas price guarantees almost immediate inclusion in the blockchain.

The changes in daily Ethereum traffic volume sometimes cause temporary network congestion, which affect user gas prices. Hence we normalized the gas price by the daily network average. In Figure 7, the two peaks of the normalized gas price around 0.5 and 1 correspond to the slow and average gas price options. On the other hand, users only occasionally charge more than three times the daily average gas price. The combination of these gas price levels forms the so-called gas price profile for each Ethereum user.

Given the normalized gas prices of the transactions sent, an account is represented by the vector including the mean, median and standard deviation, as well as the histogram divided into b_{gas} bins.

6.3 Transaction graph analysis

The set of addresses used in interactions characterize a user. Users with multiple accounts might interact with the same addresses or services from most of them. Furthermore, as users move funds between their personal addresses, they may unintentionally reveal their address clusters.

To exploit the transaction network for deanonymization, we constructed a transaction graph G with nodes as Ethereum addresses and edges as transactions in our data collection. To find similar node pairs in this network, we use node embedding methods that map graph vertices into a Euclidean space in a way that nodes with similar neighbourhood are close in the embedded space. To the best of our knowledge, we are the first to apply node embedding for Ethereum user profiling. In our measurements, we used the graph embedding library[4] of Rozenberczki et al. [45], which includes ten graph neighbourhood preserving embedding methods [28, 63, 46, 44, 43, 40, 11, 42, 52, 4] as well as two structural ones [16, 46].

We applied the embedding methods after the following preprocessing steps. First, we considered transactions as undirected edges and removed loops and multi-edges. We also removed nodes with degree one as well as vertices that are not present in the largest connected component. The resulting graph has 16,704 nodes and 132,231 edges. We

[4]https://github.com/benedekrozemberczki/karateclub

generated 128-dimensional representations for the accounts. In order to combine with timestamp and gas price representations, we assign the overall average of the network embedding vectors to the removed nodes.

6.4 Evaluation

Based on timestamp and gas price distributions as well as network embedding, we generate Euclidean feature vectors for 3321 Ethereum addresses with each having at least five transactions sent, see Table 1. Given a target address, we order all other addresses in our set by their Euclidean distance from the target. We consider multiple representations by concatenating the vectors of the timestamp, gas price and network embedding representations.

In the evaluation, we use 129 address pairs for testing that belong to the same ENS name. The accuracy metrics of Section 5 for identifying accounts of the same user by using time-of-day activity and normalized gas price is given in Figures 9–11. While time-of-day representation works best with $b_{\text{hour}} = 4$ to 6 (four to six hour long bins), normalized gas price representation performs weaker and the histogram gives only very small improvement with $b_{\text{hour}} = 50$ over mean, median and standard deviation.

The performance of 12 different node embedding algorithms is shown in Figures 12–14 based on 10 independent experiments. Diff2Vec [46], a neighbourhood preserving embedding technique performed best, followed by Role2Vec [1], which captures the structural node properties in the graph. Reciprocal rank combination of Diff2Vec and Role2Vec gives the best performance.

In Figure 15, we show the fraction of pairs where the rank of the ground truth pair is not more than a given value. Surprisingly, Diff2Vec and Role2Vec find the corresponding ENS address pairs within 100 closest representations by almost 20% more likely than time-of-day activity and gas price statistics. The combination of Diff2Vec and Role2Vec further improves the performance.

7 Deanonymizing trustless mixing services on Ethereum

As the Ethereum community realises the consequences of the lack of privacy on Ethereum, more and more emphasis is put on increasing transaction privacy [31, 49, 51]. Hence, privacy-enhancing tools became crucially important gadgets in the Ethereum ecosystem. Without doubt, the most popular is Tornado Cash (TC), a non-custodial zkSNARK-based mixer. It allows its users to enhance their anonymity by hiding their identity among a set of participating users. In this section, we provide techniques and heuristics allowing one to decrease the anonymity achieved in a TC mixer.

The Tornado Cash (TC) Mixers are sets of trustless Ethereum smart contracts allowing Ethereum users to enhance their anonymity.

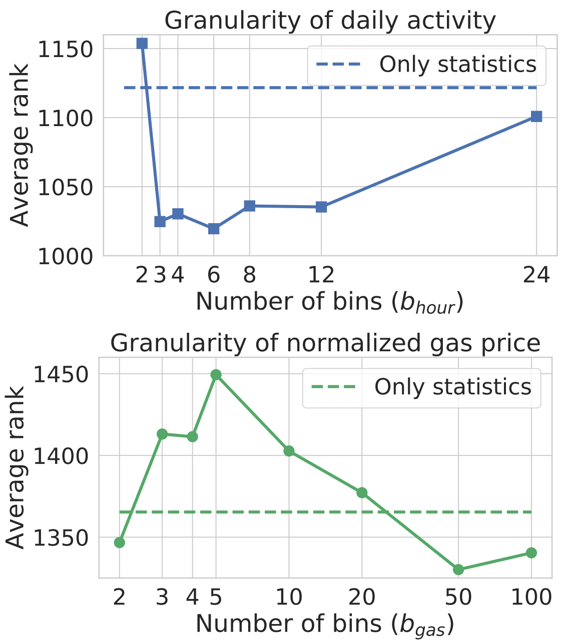

Figure 9: Average rank at different granularity for daily activity **(top)** and normalized gas price **(bottom)**. **Dashed lines** show performance with only mean, median and standard deviation used.

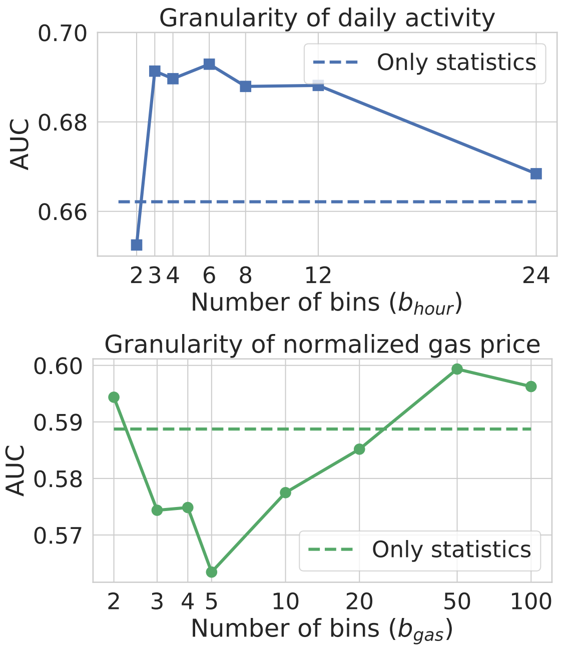

Figure 10: AUC at different granularity for daily activity **(top)** and normalized gas price **(bottom)**. **Dashed lines** show performance with only mean, median and standard deviation used.

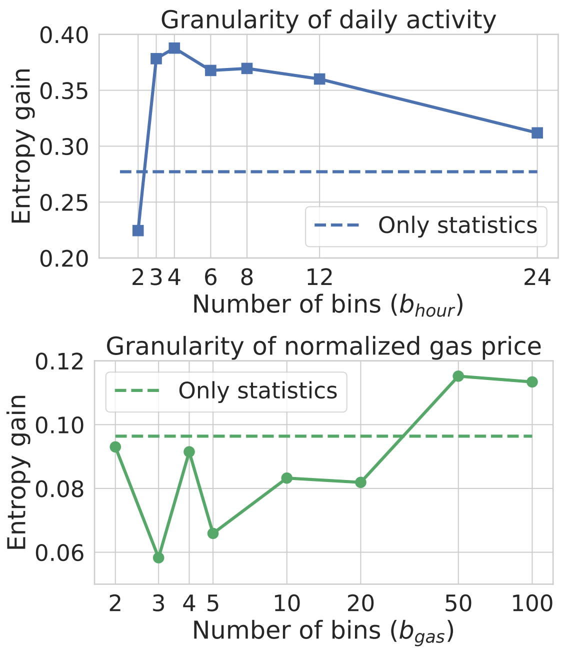

Figure 11: Entropy gain at different granularity for daily activity **(top)** and normalized gas price **(bottom)**. **Dashed lines** show performance with only mean, median and standard deviation used.

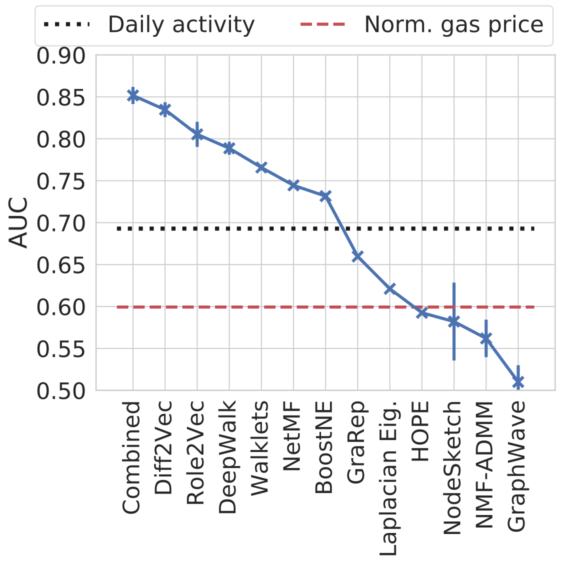

Figure 13: AUC for node embedding methods. Vertical lines show standard deviation in 10 independent experiments. Reciprocal rank combination of Diff2Vec and Role2Vec gives the best performance.

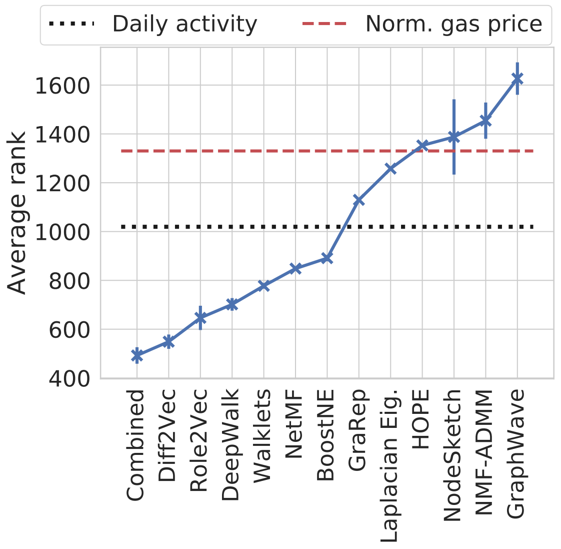

Figure 12: Average rank for node embedding methods. Vertical lines show standard deviation in 10 independent experiments. Reciprocal rank combination of Diff2Vec and Role2Vec gives the best performance.

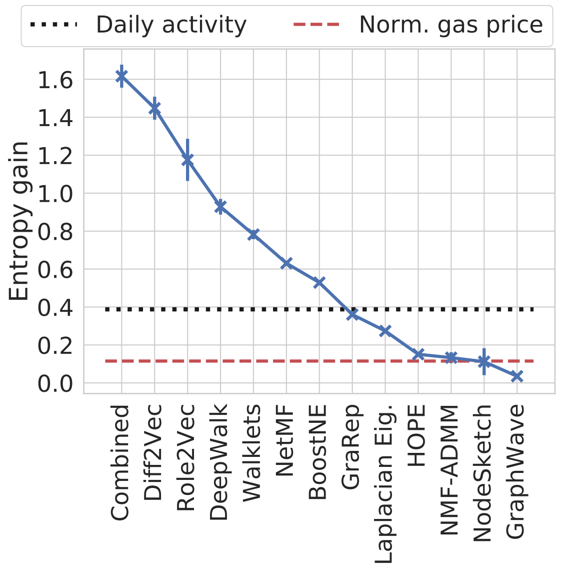

Figure 14: Entropy gain for node embedding methods. Vertical lines show standard deviation in 10 independent experiments. Reciprocal rank combination of Diff2Vec and Role2Vec gives the best performance.

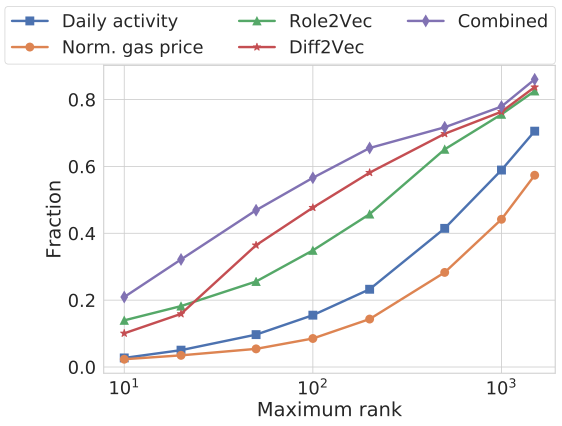

Figure 15: Fraction of ENS address pairs correctly identified within a given maximum rank, for different embedding methods.

A TC mixer contract holds equal amounts of funds (ether or other ERC-20 tokens) from a set of depositors. One mixer contract typically holds one type of asset. In case of the TC mixer, anonymity is achieved by applying zkSNARKs [22]. Each depositor inserts a hash value in a Merkle-tree. Later, at withdraw time, each legitimate withdrawer can prove unlinkably with a zero-knowledge proof that they know the pre-image of a previously inserted hash leaf in the Merkle-tree. Subsequently, users can withdraw their asset from the mixer whenever they consider that the size of the anonymity set is satisfactory.

Cryptocurrency mixers typically provide k-anonymity (also known as plausible deniability) to their users [48]. Generally speaking, a k-anonymized dataset has the property that each record is indistinguishable from at least $k-1$ others. Specifically, if a mixer contract holds n deposits out of which $n-k$ had already been withdrawn, then the next withdrawer will be indistinguishable among at least those k users who have not withdrawn from the mixer yet. Hence each withdrawer can enhance their transaction privacy and make their identity indistinguishable among at least k addresses. We call the set containing the k indistinguishable addresses the anonymity set of the user.

In Figure 16, we show the changes in the anonymity set size over time for four TC mixer contracts (0.1 ETH, 1 ETH, 10 ETH, 100 ETH) respectively. Since TC was launched in December 2019, hundreds of deposits were placed in the mixers as more and more user interacted with this service. In general, we observe orders of magnitude lower activity for the 100ETH mixer, thus it does not provide as much anonymity as mixers with lower values (0.1ETH, 1ETH, 10ETH).

7.1 Heuristics for linking mixer deposits and withdraws

Unfortunately, careless usage easily reveals links between deposits and withdraws and also impact the anonymity of other users, since if a deposit can be linked to a withdraw, it will no longer belong to the anonymity set. Next, we list three usage patterns that can be used to link deposits and withdraws. The simplest careless usage is applying the same address for deposit and withdraw transactions as well:

Heuristic 1. *If there is an address from where a deposit and also a withdraw has been made, then we consider these deposits and withdraws linked.*

The next heuristic is based on salient gas price settings. Most wallet softwares, e.g. Metamask or My Ether Wallet, automatically sets gas prices as multiples of Gwei (10^9 wei, i.e. giga wei). However, one can observe gas prices whose last 9 digits are non-zero, hence those gas prices are likely set by the transaction issuer manually. These custom-set gas prices can be used to link deposits and withdraw transactions. For instance, one might observe the deposit transaction[5] at block height 9,418,956 with 5.130909091 Gwei gas price. Later on, there is a withdraw transaction[6] at block height 9,419,096 in the Ethereum blockchain with exactly the same custom-set gas price. This deposit and withdraw pair can be linked.

Heuristic 2. *If there is a deposit-withdraw pair with* unique and manually set gas prices*, then we consider them as linked.*

Frequently, users reveal links between their deposit and withdraw addresses if they sent transactions from one of their addresses to another address belonging to them. We conjecture that users falsely expect that withdraw addresses are clean, therefore they can send transactions from any address to their clean withdraw addresses. However, if the withdraw address can be linked to one of their deposit addresses, then they effectively lose all privacy guarantee accomplished by the fresh withdraw address. Express differently, if users run out of clean funds at their fresh addresses, they might feel tempted to move "dirty" assets to their "clean" addresses. Again, such a transaction links "clean" and "dirty" addresses which is captured by the following heuristic.

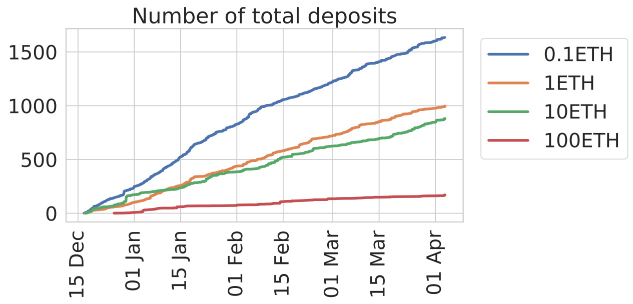

Figure 16: The number of total deposits in each TC mixer over time. This is an upper bound for the achievable anonymity set size when a withdraw transaction is executed. The popularity of the 0.1ETH mixer is superior compared to higher value mixers.

[5]Depositor: *0x074a3e9451fe3fb47be47786cf2dc4e84e797a6f*
[6]Withdrawer: *0x0f2437ff38e032596f2226873038230dcb22c485*

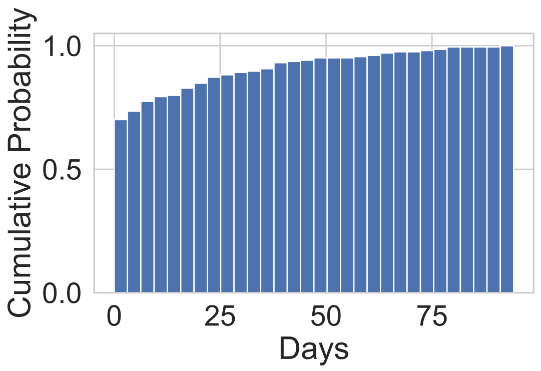

Figure 17: Elapsed time in days between linked deposit and withdraw transactions for the 0.1 ETH mixer contract. Vast majority of users do not wait more than one day to withdraw their deposits.

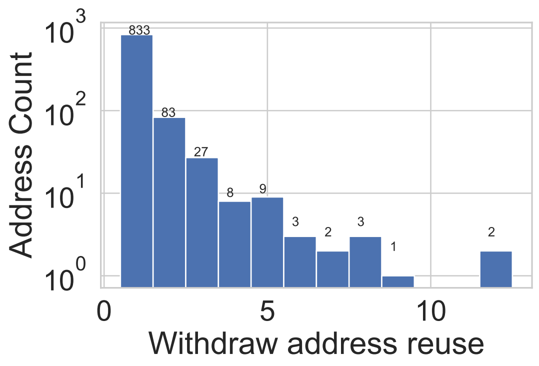

Figure 19: Withdraw address reuse in the 0.1 ETH mixer contract. Many users withdraws multiple deposits to the same address, which ease deanonymization and reduce overall the privacy in the mixer.

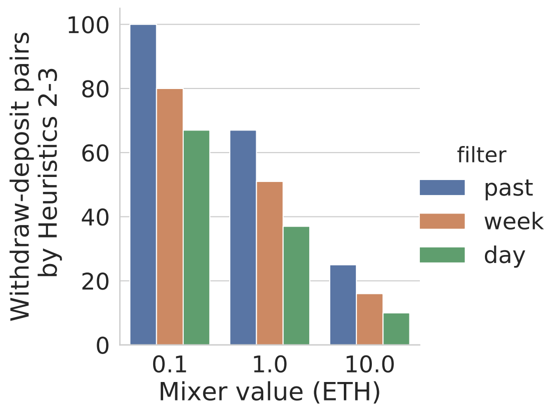

Figure 18: For each mixer, the number of withdraw-deposit pairs linked by Heuristics 2–3 such that the deposit is not later than the day or the week before, or any time in the past.

Heuristic 3. Let d be a deposit and w a withdraw address in a TC mixer. If there is a transaction between d and w (or vice versa), we consider the addresses linked.

One could easily generalize Heuristic 3 by requiring transactions to be sent from not only a depositor address d, but rather from any address in the cluster of addresses containing d. However, we leave the implementation of this generalization for future work.

Applying Heuristics 1–3, we found 218, 110, 60, and 7 withdraws linked in the four mixer contracts (0.1 ETH, 1 ETH, 10 ETH, 100 ETH) respectively, see Table 2. We note that withdraws identified by Heuristic 2 can also overlap with other withdraws identified by Heuristic 1 or 3. Hence the number of total linked withdraws are less than the sum of all withdraws individually identified by each heuristic.

7.2 Elapsed time between deposit and withdraw

In Figure 17, we observe that most users of the linked deposit-withdraw pairs leave their deposit for less than a day in the mixer contract. This user behavior can be exploited for deanonymization by assuming that the vast majority of the deposits are always withdrawn after one or two days.

Even worse, in Figure 19 we observe several addresses receiving more than one withdraws from the 0.1 ETH mixer contract. For instance, there are 85 addresses with two withdraws and 27 addresses with three withdraws. Withdraw clusters cause privacy risk not just for the owner but for all other mixer participants as well. Note that proper usage requires withdraw always to fresh addresses.

7.3 Deanonymization performance

Next we measure how well the techniques of Section 6 identify the linked withdraw-deposit address pairs. We build ground truth by using Heuristics 2–3 of Section 7.1. We define three different **ground truth sets**, one when the deposit is within the past day of the withdraw, another when within the past week, and the unfiltered full set, see Fig. 18.

Note that our ground truth sets are compiled by using Heuristics 2–3, and hence are correct up to our best knowledge on the data. Since in Heuristic 2 we used gas prices and in Heuristic 3 an edge between the two addresses, in this section, we show gas price only as reference, and omit the edges used by the heuristic for the network analysis algorithms. As we will see, gas price distribution performs weak for finding the account pairs identified by the Heuristics despite that Heuristic 2 is based on gas price, adding the edges between accounts identified by Heuristic 3 would yield overly strong deanonymization results, since the same information is used for deanonymization and testing.

Figure 20 shows that an address with withdraw within a day or week has significantly smaller anonymity set size, on average, since we only search for the corresponding deposit

	Deanonymized withdraws				All
Mixer	Heuristic 1	Heuristic 2	Heuristic 3	Total	Withdraws
0.1ETH	95 (7.5%)	80 (6.2%)	113 (8.8%)	218 (17.1%)	1272
1ETH	21 (2.5%)	40 (4.8%)	75 (9%)	110 (13.2%)	833
10ETH	8 (1.1%)	9 (1.2%)	46 (6.2%)	60 (8.1%)	738
100ETH	2 (1.5%)	5 (3.8%)	3 (2.3%)	7 (5.3%)	132

Table 2: Number of all withdraws and deanonymized withdraws using the corresponding heuristics in each mixer contract.

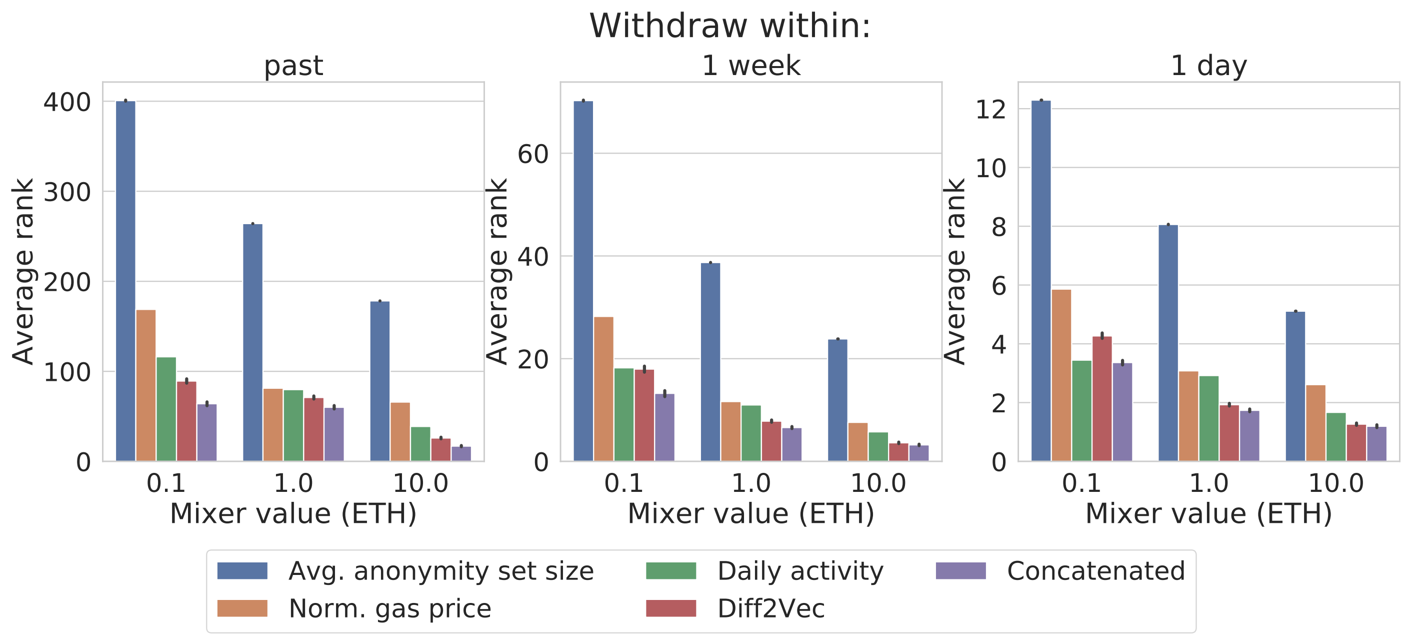

Figure 20: Average rank of the deposit address in the candidate list of our algorithms for the three different ground truth sets.

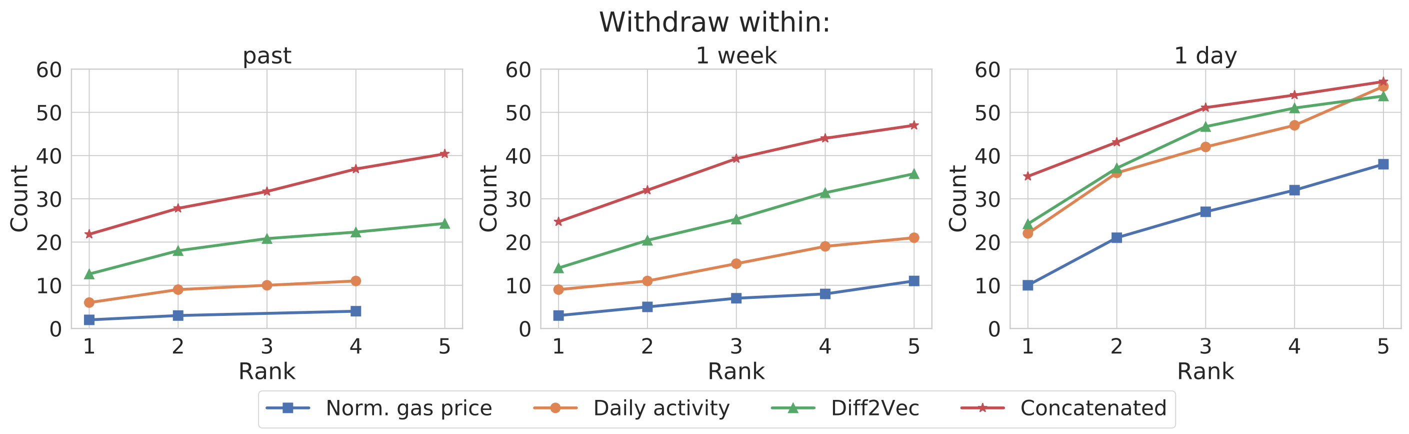

Figure 21: Number of withdraw addresses in the 0.1ETH mixer contract such that the corresponding deposit is identified within the given rank in the candidate list of each deanonymization technique, separate for the three ground truth sets.

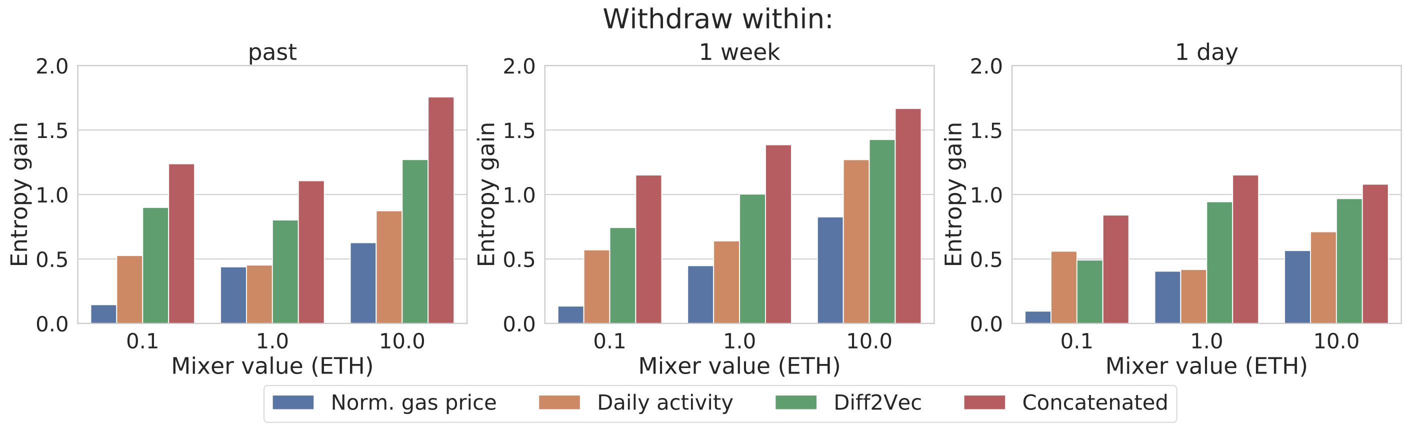

Figure 22: Entropy gain of our best deanonymization methods for the three different ground truth sets.

in a smaller set. For example, for the 0.1ETH mixer the original average anonymity set size of 400 could be reduced to almost 12 by assuming that the deposit occurred within one day of the withdraw.

We note that in Figure 20 and all other measurements over the filtered ground truth sets, we do not discount for the withdraw addresses that are not included in the filtered set. For example, as seen in Figure 17, for 80 0.1-Ether withdraw transactions, we list candidate deposits, but for the remaining 20, we make no deanonymization attempt. To normalize the results by considering these withdraws, we have to assume that the corresponding deposit is not in the 80-element candidate set but in the remaining 320, thus giving an average rank contribution of 160 for 20% of the data. Hence average rank for 0.1-Ether withdraws with deposit within a week have an additional correction of 32 for average rank; by similar calculations, the correction for transactions within a day is 63.

Daily activity and Diff2Vec have similar performance while their concatenated feature vectors proved to be the best address representation; for the smaller ground truth sets, they identify related deposit addresses within the 20 and 5 closest representations on average. Withdraw linking performance is further improved by concatenating the two models. Entropy gain is shown in Figure 22 and the number of withdraws linked to deposits within a given rank of the output for the best methods are in Figure 21.

In Figure 23, we show the withdraw linking performance over time. As the number of active deposits increases, it becomes harder to link withdraws to any of the past deposits. However withdraws that follow the deposit after a few days are still much easier to deanonymize.

7.4 Maintaining privacy

We do believe if users were using the technology in a sound way or a privacy-focused wallet software would have helped them and abstracted away potential privacy leaks, then TC mixers could possibly achieve higher degrees of anonymity.

7.4.1 Randomized mixing intervals

Mixing participants decrease largely their gained anonymity by withdrawing funds after short time intervals, cf. Figure 17 and 20. These heuristics can be defeated by randomized mixing intervals.

7.4.2 Fresh withdraw addresses

Currently, many users apply the same withdraw addresses across several withdraws, see Figure 19. This behavior greatly decreases the complexity of linking deposits and withdraws. Therefore users must use fresh withdraw addresses for each of their withdraws. This issue could have been easily fixed on the user interface level.

7.4.3 Mixer usage and user behaviors

Mixers mainly attempt to break the link between sets of transaction graphs associated with Ethereum accounts. As such, users need to ensure that their on-chain behaviors are unlinkable between uses of the TC mixers. Therefore, to ensure maximal privacy, users should use the TC mixers after every transaction. However, this decreases the user experience and ability to use applications on Ethereum.

8 Danaan-gift attack in Ethereum

The Danaan-gift attack, also known as malicious value fingerprinting, was introduced in [6]. In a value fingerprinting attack, an adversary sends a cryptocurrency transaction with a crafted amount to add a fingerprint to the receiver's account balance. Although value fingerprinting was originally introduced in the context of Zcash, we notice that these attacks are applicable to Ethereum as well. Most wallet software denominates gas prices in multiples of gwei (10^9 wei where $1ETH = 10^{18} wei$), hence transaction fees overwhelmingly (in $98,1\%$) do not change the last 9 digits of an account balance. Albeit, users might set transaction

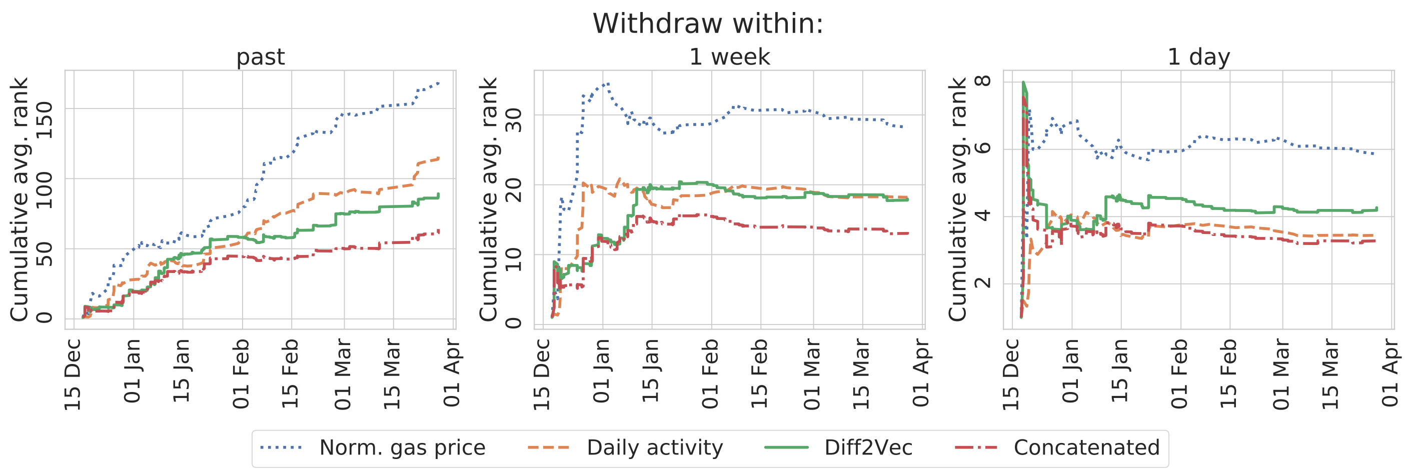

Figure 23: Change of average rank in time, cumulated from the beginning of our data, for the 0.1 ETH Tornado mixer by using our best deanonymization methods.

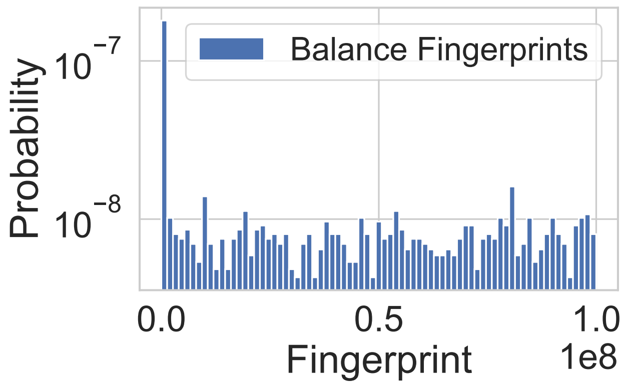

Figure 24: Ether account balance fingerprints. Many Ethereum accounts have an integer account balance. This allows an attacker to fingerprint the last 9 digits of an account balance. Account balance fingerprints distribution has a 4.01 bit entropy and 6.44 bit entropy gain.

fees manually, potentially changing their own fingerprint (in 1.9%). The last 9 digits of an account balance have no economic significance (1 gwei≈ 0.0000003$) but could be used as a fingerprint by an adversary.

First, we measure the fraction of ether transfer transactions that modify the account fingerprint $(43, 7\%)$. For the sake of robustness of the measurements, we chose fingerprints with the last eight digits. As seen in Figure 24, account balances are mostly integer values. However, the rest of the fingerprint values modulo 100,000,000 are moderately uniformly distributed. The entropy of the account balance fingerprints is 4.01 with a 6.44 entropy gain. These results suggest that account balances might be easily fingerprinted.

In the sequel, we estimate the average fingerprint survival probability.

Let F denote the event that a fingerprint of an address remains unchanged. To approximate the event probability $\Pr(F)$, let p denote the probability that a transaction modified the fingerprint and let x denote the number of transactions sent or received by the given address in our dataset. By assuming that each transaction is independent from all others, the fingerprint survival probability of this address is $(1-p)^x$.

We observe that the distribution of the number x of transactions sent and received by an address follow power-law distribution $\sim x^{-k}$ with $k = 1.91$. The average survival probability of all addresses can hence be approximated by the following integral, where we group by x, the number of transactions of an address:

$$\Pr(F) = \int_1^{\infty} x^{-k}(1-p)^x dx, \tag{3}$$

which can be computed in a closed formula. The numerical values are summarized in Table 3.

As the number of transactions sent follow a power-law distribution, the average value is skewed by the tail of the distribution. Therefore it makes sense to calculate the average survival probability for several cutoffs of the tail, see Table 3. Namely, in each cutoff we only consider addresses in our data set that sent less number of transactions than the cutoff value. One can observe how fingerprint survival probability increases among users with a small number of transactions. For example, an adversary could successfully fingerprint 21.83% of the addresses that send not more than 50 transactions. This result is comparable to the 16.6% fingerprint survival probability observed in Zcash [6].

Tx Cutoff	Addresses	Txs	Txs Fingerprinting	Avg. Sent Txs/Address	Fingerprint survival prob.
50	56,399	120,461	61,393	2.14	21.83%
100	56,973	161,427	73,340	2.83	17.97%
500	57,951	384,369	129,431	19.48	6.56%
All	58,367	1,137,558	352,042	19.49	0.073%

Table 3: Balance fingerprinting statistics for Ethereum users. In each cutoff, we only consider addresses that did not issue more transactions than the cutoff value. We observe that vast majority of fingerprinting transactions were sent by addresses that send numerous transactions. Fingerprinting an address with few sent transactions is obviously easier than an address with many issued transactions. Fingerprint survival probabilities were calculated as in Equation 3.

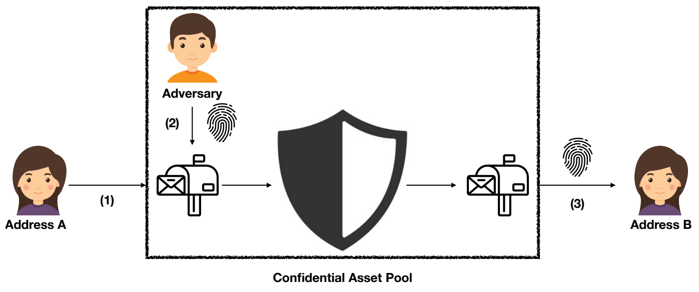

Figure 25: Danaan-gift attack in confidential transaction layers. An adversary can fingerprint (2) an unsuspecting user's account balance after she deposited assets (1) in a confidential asset pool, e.g. AZTEC. Adversary can track the user when she leaves (3) the confidential pool.

8.1 Danaan-gift attack for confidential transaction overlays

We foresee that a prominent application of Danaan-gift attacks in Ethereum might be linking confidential transactions in privacy-enhancing overlays like the AZTEC protocol [61].

In a confidential transaction overlay, users can convert public amounts into confidential notes. Subsequently, they can send confidential notes to intended recipients by splitting and or joining their confidential notes. The amount of confidential notes is hidden, yet publicly verifiable by applying range proofs. Users can also convert their confidential tokens back to public amounts.

In this scenario, an adversary can fingerprint unsuspecting users inside a confidential transaction overlay, see Figure 25. Whenever a user deposits a public amount to the confidential asset pool an adversary could fingerprint her account. Subsequently, the user might issue several confidential transactions in this privacy-enhanced overlay. If the victim's balance fingerprint survives during the course of issued confidential transactions, the adversary can observe when the user withdraws funds from the confidential asset pool by inspecting the fingerprint on the withdrawn amount. Thus the fingerprinting adversary can assess how much money the unsuspecting user paid in the confidential asset pool.

9 Future directions

We expect that in the near future more potent and powerful deanonymization tools and techniques will emerge. In this work, we solely applied on-chain data for deanonymizing Ethereum users. Subsequent tools will likely use a combination of on-chain and off-chain data. Therefore we deem the following directions would be extremely valuable for future work for the broader cryptocurrency research community.

9.1 Further quasi-identifiers

In this work we identified several quasi-identifiers of Ethereum accounts, such as time-of-day activity, gas price profile and position in the Ethereum transaction graph. However, we forecast that many more quasi-identifiers can be used for further profiling and deanonymizing Ethereum users. One such potential quasi-identifier is wallet fingerprints. One could establish which wallet a certain user employs by assessing how transaction gas prices are calculated. Different wallet softwares use different methods to compute suggested gas prices [59].

9.2 Network-level privacy

Assessing Ethereum's privacy provisions entirely can only be established if one considers the full life-cycle of a transaction. Specifically, one also needs to understand how much privacy is lost when users interact with full nodes or wallet providers.

As the history of Bitcoin and other cryptocurrencies showed, full nodes and wallet providers can deanonymize regular users and light clients already on the network layer [5, 7, 18, 19, 54, 33]. An attacker could establish many well-connected nodes in the peer-to-peer layer to log the timing information of transactions. Due to the symmetry of broadcast, the adversary could infer the origin of the transaction [19, 5]. Yet, there are solely measurement studies on Ethereum's P2P network structure [26, 20]. Therefore, it would be worthwhile to conduct a study on Ethereum's P2P network, but from a privacy point of view. Fortunately, several proposals had been made to enhance network-level privacy for cryptocurrencies [8, 17].

Additionally, in Ethereum, special nodes called relayers gain more and more popularity. Relayers allow senders to issue feeless transactions, i.e. users can send transactions from addresses that do not hold ether yet. Such relayer nodes can also easily deanonymize their users. This is especially problematic in case of non-custodial mixers, like Tornado Cash.

9.3 Wallet and Browser Privacy

It has been shown how online trackers and cookies can aid the deanonymization of cryptocurrency users even when their coins were mixed through the use of a mixer [21]. Many users of the Ethereum blockchain make use of a tool called MetaMask, a browser extension available in most desktop browsers. As such, for future research, it would be fascinating to analyze how the use of this extension affects the privacy of Ethereum users, even with the use of mixers. It may be possible to use the techniques presented in [21] to deanonymize users. Furthermore, as many Ethereum users also make use of mobile wallets, it may be useful to investigate how mobile phones can affect cryptocurrency users' privacy and assess the privacy guarantees of these mobile wallet providers [6].

9.4 Privacy of UTXO-based cryptocurrencies

We note that the deanonymizing power of quasi-identifiers (e.g. temporal activity, wallet fingerprints etc.) is also applicable to UTXO-based cryptocurrencies. Even though in that case deanonymization is slightly more involved as one need to apply our techniques not to individual addresses but rather to clusters of UTXOs. We do foresee that more potent agencies can and will engage in such deanonymization campaigns. We believe that in practice, due to the aforementioned quasi-identifiers, also Bitcoin non-custodial mixers provide drastically less privacy and fungibility than what currently the community expects from those privacy-enhancing technologies.

Acknowledgements

We thank Daniel A. Nagy, David Hai Gootvilig, Domokos M. Kelen and Kobi Gurkan for conversations and useful suggestions. Support from Project 2018-1.2.1-NKP-00008: Exploring the Mathematical Foundations of Artificial Intelligence and the "Big Data——Momentum" grant of the Hungarian Academy of Sciences.

References

[1] Nesreen Ahmed, Ryan Rossi, John Lee, Xiangnan Kong, Theodore Willke, Rong Zhou, and Hoda Eldardiry. Learning role-based graph embeddings. In *StarAI workshop, IJCAI 2018*, pages 1–8, 2018.

[2] Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. Evaluating user privacy in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 34–51. Springer, 2013.

[3] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web*, pages 181–190, 2007.

[4] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 585–591. MIT Press, 2002.

[5] Alex Biryukov and Daniel Feher. Deanonymization of hidden transactions in zcash, 2018.

[6] Alex Biryukov, Daniel Feher, and Giuseppe Vitto. Privacy aspects and subliminal channels in zcash. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1813–1830, 2019.

[7] Alex Biryukov and Sergei Tikhomirov. Security and privacy of mobile wallet users in bitcoin, dash, monero, and zcash. *Pervasive and Mobile Computing*, 59:101030, 2019.

[8] Shaileshh Bojja Venkatakrishnan, Giulia Fanti, and Pramod Viswanath. Dandelion: Redesigning the bitcoin network for anonymity. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(1):1–34, 2017.

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

[10] Benedikt Bünz, Shashank Agrawal, Mahdi Zamani, and Dan Boneh. Zether: Towards privacy in a smart contract world. *IACR Cryptology ePrint Archive*, 2019:191, 2019.

[11] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, page 891–900, New York, NY, USA, 2015. Association for Computing Machinery.

[12] Yu Chen, Xuecheng Ma, Cong Tang, and Man Ho Au. Pgc: Pretty good decentralized confidential payment system with auditability. Cryptology ePrint Archive, Report 2019/319, 2019. https://eprint.iacr.org/2019/319.

[13] Joao Otávio Massari Chervinski, Diego Kreutz, and Jiangshan Yu. Floodxmr: Low-cost transaction flooding attack with monero's bulletproof protocol. *IACR Cryptology ePrint Archive*, 2019:455, 2019.

[14] Nicolas Christin. Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In *Proceedings of the 22nd international conference on World Wide Web*, pages 213–224, 2013.

[15] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In *International Workshop on Privacy Enhancing Technologies*, pages 54–68. Springer, 2002.

[16] Claire Donnat, Marinka Zitnik, David Hallac, and Jure Leskovec. Learning structural node embeddings via diffusion wavelets. In *International ACM Conference on Knowledge Discovery and Data Mining (KDD)*, volume 24, 2018.

[17] Giulia Fanti, Shaileshh Bojja Venkatakrishnan, Surya Bakshi, Bradley Denby, Shruti Bhargava, Andrew Miller, and Pramod Viswanath. Dandelion++ lightweight cryptocurrency networking with formal anonymity guarantees. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2(2):1–35, 2018.

[18] Giulia Fanti and Pramod Viswanath. Anonymity properties of the bitcoin p2p network. *arXiv preprint arXiv:1703.08761*, 2017.

[19] Giulia Fanti and Pramod Viswanath. Deanonymization in the bitcoin p2p network. In *Advances in Neural Information Processing Systems*, pages 1364–1373, 2017.

[20] Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert Van Renesse, and Emin Gün Sirer. Decentralization in bitcoin and ethereum networks. In *International Conference on Financial Cryptography and Data Security*, pages 439–457. Springer, 2018.

[21] Steven Goldfeder, Harry A. Kalodner, Dillon Reisman, and Arvind Narayanan. When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies. *Proceedings on Privacy Enhancing Technologies*, 2018:179 – 199, 2017.

[22] Jens Groth. On the size of pairing-based non-interactive arguments. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 305–326. Springer, 2016.

[23] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[24] George Kappos, Haaroon Yousaf, Mary Maller, and Sarah Meiklejohn. An empirical analysis of anonymity in zcash. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 463–477, 2018.

[25] Lucianna Kiffer, Dave Levin, and Alan Mislove. Analyzing ethereum's contract topology. In *Proceedings of the Internet Measurement Conference 2018*, pages 494–499, 2018.

[26] Seoung Kyun Kim, Zane Ma, Siddharth Murali, Joshua Mason, Andrew Miller, and Michael Bailey. Measuring ethereum network peers. In *Proceedings of the Internet Measurement Conference 2018*, pages 91–104, 2018.

[27] Robin Klusman. Deanonymisation in ethereum using existing methods for bitcoin. 2018.

[28] Jundong Li, Liang Wu, and Huan Liu. Multi-level network embedding with boosted low-rank matrix approximation. *CoRR*, abs/1808.08627, 2018.

[29] Shlomi Linoy, Natalia Stakhanova, and Alina Matyukhina. Exploring ethereum's blockchain anonymity using smart contract code attribution. 10 2019.

[30] Jiangtao Ma, Yaqiong Qiao, Guangwu Hu, Yongzhong Huang, Arun Kumar Sangaiah, Chaoqin Zhang, Yanjun Wang, and Rui Zhang. De-anonymizing social networks with random forest classifier. *IEEE Access*, 6:10139–10150, 2017.

[31] Sarah Meiklejohn and Rebekah Mercer. Möbius: Trustless tumbling for transaction privacy. *Proceedings on Privacy Enhancing Technologies*, 2018(2):105–121, 2018.

[32] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140, 2013.

[33] Malte Möser, Kyle Soska, Ethan Heilman, Kevin Lee, Henry Heffan, Shashvat Srivastava, Kyle Hogan, Jason Hennessey, Andrew Miller, Arvind Narayanan, et al. An empirical analysis of traceability in the monero blockchain. *Proceedings on Privacy Enhancing Technologies*, 2018(3):143–163, 2018.

[34] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[35] Arvind Narayanan, Elaine Shi, and Benjamin IP Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *The 2011 International Joint Conference on Neural Networks*, pages 1825–1834. IEEE, 2011.

[36] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125. IEEE, 2008.

[37] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. In *2009 30th IEEE symposium on security and privacy*, pages 173–187. IEEE, 2009.

[38] Danny Nelson. Inside chainalysis' multimillion-dollar relationship with the us government. coindesk, 2020. https://www.coindesk.com/inside-chainalysis-multimillion-dollar-relationship-\with-the-us-government.

[39] Robert Norvill, Beltran Borja Fiz Pontiveros, Radu State, Irfan Awan, and Andrea Cullen. Automated labeling of unknown contracts in ethereum. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6. IEEE, 2017.

[40] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1105–1114, New York, NY, USA, 2016. Association for Computing Machinery.

[41] James Payette, Samuel Schwager, and Joseph Murphy. Characterizing the ethereum address space, 2017.

[42] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA, 2014. ACM.

[43] Bryan Perozzi, Vivek Kulkarni, and Steven Skiena. Walklets: Multiscale graph embeddings for interpretable network classification. *CoRR*, abs/1605.02115, 2016.

[44] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. *CoRR*, abs/1710.02971, 2017.

[45] Benedek Rozemberczki, Oliver Kiss, and Rik Sarkar. An api oriented open-source python framework for unsupervised learning on graphs, 2020.

[46] Benedek Rozemberczki and Rik Sarkar. Fast sequence based embedding with diffusion graphs. In *International Conference on Complex Networks*, pages 99–107, 2018.

[47] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. Coinshuffle: Practical decentralized coin mixing for bitcoin. In *European Symposium on Research in Computer Security*, pages 345–364. Springer, 2014.

[48] Pierangela Samarati and Latanya Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. 1998.

[49] István András Seres, Dániel A Nagy, Chris Buckland, and Péter Burcsi. Mixeth: efficient, trustless coin mixing service for ethereum. In *International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[50] Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In *International Workshop on Privacy Enhancing Technologies*, pages 41–53. Springer, 2002.

[51] Omer Shlomovits and István András Seres. Sharelock: Mixing for cryptocurrencies from multiparty ecdsa. *Cryptol. ePrint Arch., Tech. Rep*, 563:2019, 2019.

[52] D. L. Sun and C. Févotte. Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6201–6205, 2014.

[53] Florian Tramer, Dan Boneh, and Kenneth G Paterson. Ping and reject: The impact of side-channels on zcash privacy. 2019.

[54] Florian Tramèr, Dan Boneh, and Kenneth G Paterson. Remote side-channel attacks on anonymous transactions. 2020.

[55] Luke Valenta and Brendan Rowan. Blindcoin: Blinded, accountable mixes for bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 112–126. Springer, 2015.

[56] Friedhelm Victor. Address clustering heuristics for ethereum.

[57] Friedhelm Victor and Bianca Katharina Lüders. Measuring ethereum-based erc20 token networks. In *International Conference on Financial Cryptography and Data Security*, pages 113–129. Springer, 2019.

[58] Isabel Wagner and David Eckhoff. Technical privacy metrics: a systematic survey. *ACM Computing Surveys (CSUR)*, 51(3):1–38, 2018.

[59] Sam M Werner, Paul J Pritz, and Daniel Perez. Step on the gas? a better approach for recommending the ethereum gas price. *arXiv preprint arXiv:2003.03479*, 2020.

[60] Barry Whitehat. Miximus: zksnark-based trustless mixing for ethereum. github, 2018. https://github.com/barryWhiteHat/miximus.

[61] DZJ Williamson. The aztec protocol. *URL: https://github. com/AztecProtocol/AZTEC*, 2018.

[62] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32, 2014.

[63] Dingqi Yang, Paolo Rosso, Bin Li, and Philippe Cudre-Mauroux. Nodesketch: Highly-efficient graph embeddings via recursive sketching. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, page 1162–1172, New York, NY, USA, 2019. Association for Computing Machinery.

[64] Jan Henrik Ziegeldorf, Fred Grossmann, Martin Henze, Nicolas Inden, and Klaus Wehrle. Coinparty: Secure multi-party mixing of bitcoins. In *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, pages 75–86, 2015.

A Ethereum basics

Ethereum is a cryptocurrency built on top of a blockchain [62]. There are two types of accounts in Ethereum: externally owned accounts (EOAs) and contract accounts, also known as smart contracts. The global state of the system consists of the state of all different accounts. EOAs are controlled by an asymmetric cryptographic key pair, while smart contracts are controlled by their code stored in persistent, immutable storage. EOAs can issue transactions, which might alter the global state. Transactions can either create a new contract account or call existing accounts. Accounts have balances in ether, the native currency of Ethereum, and are denominated in wei where $1ETH = 10^{18}wei$.

Calls to EOAs can transfer Ether to the callee, while contract calls execute the code associated with the smart contract. The contract execution might alter the storage of the account, moreover can call to other accounts - these are called *internal transactions*. Contract code is executed in the Ethereum Virtual Machine (EVM).

A.1 Gas mechanism

A crucial aspect of the EVM is the gas mechanism. To every EVM opcode, there is a gas amount assigned, which is deemed to price the computational complexity of that opcode. For instance, adding two elements on top of the stack consumes only 3 gas, but storing a non-zero stack element in the persistent storage burns 20,000 gas. The base gas fee for every transaction is 21,000 gas, which is not paid for internal transactions. Therefore, whenever one executes a smart contract code in the EVM, the execution consumes a certain amount of gas. At each transaction, the sender needs to define the maximum number of gas, called gas limit, they allow their transaction to consume. Usually, due to the dynamic nature of the state, one does not know statically how much gas would her transaction burn. If a transaction does not consume all the gas assigned to it, then surplus gas is refunded to the caller, however, if a transaction runs out of gas, then all state changes are reverted and assigned gas is taken from the caller.

As of now, gas can only be purchased by Ethereum's native currency, ether, at a dynamically changing price, called gas price. Miners are naturally incentivised to insert transactions with higher gas prices into their blocks to increase their collected transaction fees.