Blockchain is Watching You: Profiling and Deanonymizing Ethereum Users

Ferenc Béres[*][†], István A. Seres[†], András A. Benczúr[*], Mikerah Quintyne-Collins[‡]
[*]Institute for Computer Science and Control (SZTAKI)
{beres, benczur}@sztaki.hu
[†]Eötvös Loránd University
istvanseres@caesar.elte.hu
[‡]HashCloack
mikerah@hashcloak.com

***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. Due to its privacy shortcomings, recently, several privacy-enhancing overlays have been deployed on Ethereum, such as non-custodial, trustless coin mixers and confidential transactions.**

In our privacy analysis of Ethereum's account-based model, we describe several patterns that characterize only a limited set of users and successfully apply these "quasi-identifiers" in address deanonymization tasks. Using Ethereum Name Service identifiers as ground truth information, we quantitatively compare algorithms in a recent branch of machine learning, the so-called graph representation learning, as well as time-of-day activity and transaction fee based user profiling techniques. As an application, we rigorously assess the privacy guarantees of the Tornado Cash coin mixer by discovering patterns of careless usage to link the mixing parties. To the best of our knowledge, we are the first to propose and implement Ethereum user profiling techniques based on quasi-identifiers.

I. Introduction

The narrative around cryptocurrency privacy provisions has dramatically changed since the inception of Bitcoin [19]. Initially many, especially criminals, thought Bitcoin and other cryptocurrencies provide privacy to hide their illicit business activities [7]. The first extensive study about Bitcoin's privacy provisions was done by Meiklejohn et al. [17], 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 cryptocurrencies.

However, until today, there are no substantial empirical 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 many cryptocurrency use cases. Furthermore, there are state-sponsored companies and other entities, e.g. Chainalysis [23], performing large-scale deanonymization tasks on cryptocurrency users.

In contrast to the UTXO-model, many cryptocurrencies that provide smart contract functionalities operate with accounts. In an account-based cryptocurrency, users store their assets in accounts rather than in UTXOs. Already in the Bitcoin white paper, Nakamoto suggested that "a new key pair should be used for each transaction to keep them from being linked to a common owner" [19]. Despite this suggestion, account-based cryptocurrency users tend to use only a handful of addresses for their activities. In an account-based cryptocurrency, native transactions can only move funds between a single sender and a single receiver, hence in a payment transaction, the change remains at the sender account. Thus, a subsequent transaction necessarily uses the same address again to spend the remaining change amount. Therefore, the account-based model essentially relies on address-reuse on the protocol level. This behavior practically renders account-based cryptocurrencies inferior to UTXO-based currencies from a privacy perspective.

Previously, several works had identified the privacy shortcomings of the account-based model, specifically in Ethereum. Those works had proposed trustless coin mixers [16], [31], [33] and confidential transactions [4], [6], [39]. Until recently, none of these schemes has been deployed on Ethereum.

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.
- In the cryptocurrency domain, we are the first to quantitatively assess the performance of a recent area of machine learning in graphs, the so-called node embedding algorithms.
- We describe several patterns that weaken the privacy guarantees of non-custodial mixers on Ethereum.
- We collect and analyze a wide source of Etherum related data, including Ethereum name service (ENS), Etherscan blockchain explorer, Tornado Cash mixer contracts, and Twitter. We release the collected data (in anonymized form) as well as our source code for further research[1].

The rest of the paper is organized as follows. In Section II, we review related work. In Section III, a brief background is given on non-custodial mixers in Ethereum along with the general idea behind node embedding. In Section IV, we

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

describe our collected data. In Section V, we overview the literature on evaluating deanonymization methods and propose our metrics. Our main methods to pair Ethereum addresses that belong to the same user and link Tornado deposits and withdrawals are detailed in Section VI and VII. Section VIII describes best practices to maintain privacy in an account-based currency. Finally, we conclude our paper in Section IX.

II. Related Work

First results on Ethereum deanonymization [13] attempted to directly apply both on-chain and peer-to-peer (P2P) Bitcoin deanonymization techniques. The starting point of our work is the recognition that common deanonymization methods for Bitcoin *are not* 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 approach analyzes Ethereum smart contracts with unsupervised clustering techniques [24]. Kiffer et al. [12] assert a large degree of code reuse which might be problematic in the case of vulnerable and buggy contracts.

The second branch of literature assesses Ethereum addresses. A crude and initial analysis had been made by Payette et al., who clusters the Ethereum address space into only four different groups [25]. More interestingly, Friedhelm Victor proposes address clustering techniques based on participation in certain airdrops and ICOs [35]. 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 [36].

A completely different and unique approach is taken by [14], which uses stylometry to deanonymize smart contract authors and their respective accounts. The work had been used to identify scams on Ethereum.

III. Background

In this section, we provide some background on cryptocurrency privacy-enhancing technologies as well as node embedding algorithms.

A. Non-custodial mixers

Coin mixing is a prevalent technique to enhance the transaction privacy of cryptocurrency users. Coin mixers may be custodial or non-custodial. In case of custodial mixing, users send their "tainted" coins to a trusted party, who in return sends back "clean" coins after some timeout. This solution is not satisfactory as the users do not retain ownership of their coins during the course of mixing.

Motivated by these drawbacks, recently several non-custodial mixers have been proposed in the literature [16], [31], [33], [38]. 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 deposit equal amounts of ether or other tokens into a mixer contract from an address $\mathcal{A}$. After some user-defined time interval, they can withdraw their deposited coins with a withdrawal transaction to a fresh address $\mathcal{B}$. In the withdrawal 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 [16], verifiable shuffles [31], threshold signatures [33], and zkSNARKs [38].

B. Ethereum Name Service

Ethereum Name Service (ENS) is a distributed, open, and extensible naming system based on the Ethereum blockchain. 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 domains, with the owner of a domain having full control over subdomains. ENS maps human-readable names like `alice.eth` to machine-readable identifiers, e.g., Ethereum addresses. Therefore, ENS provides a user-friendly way of moving assets on Ethereum, where users can use ENS names (`alice.eth`) as recipient addresses instead of the error-prone hexadecimal Ethereum addresses.

C. Node embeddings

Node embedding methods form a class of network representation learning methods that map graph nodes to vectors in a low-dimensional vector space. They are designed to represent vertices with similar graph neighborhood by vectors that are close in the vector space. Intuitively, addresses that interact with the same set of addresses in the Ethereum transaction graph should be close in the embedded space. For example, Laplacian eigenmaps [3] and graph factorization [1] are some of the well-known network representation techniques. Research in node embedding has recently been catalyzed by Word2Vec [18], an embedding method for natural language processing. Several node embedding methods have been proposed recently [10], [26], [27], [34] and applied successfully for multi-label classification and link prediction in a variety of real-world networks from diverse domains.

In this work, we experiment with node embedding algorithms on the Ethereum transaction graph to link addresses owned by the same user. As these feature mapping techniques are not limited to single-hop transaction neighbors, they have the potential to surpass intersection attacks [9]. For example, GraRep [5] can represent multi-hop neighborhood similarity, while Diff2Vec [29] can preserve distances between nodes by sampling diffusion trees during the training process. On the other hand, Role2Vec [2], a structural node embedding method assigns similar vectors to two addresses if they have similar motif structure in the transaction graph despite their distance

[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 I
NUMBER OF ETHEREUM ACCOUNTS COLLECTED FROM THREE DIFFERENT SOURCES. *IN THE TORNADO MIXER, WE CONSIDER ACCOUNT PAIRS IDENTIFIED BY CARELESS USAGE PATTERNS AS GROUND TRUTH, SEE SECTION VII. DUE TO OVERLAPS BETWEEN THE DATA SOURCES, THE TOTAL NUMBER OF INVESTIGATED ADDRESSES IS LESS THAN THE SUM OF THE RECORDS IN THE TOP THREE ROWS.

in the network. To the best of our knowledge, we are the first to apply node embedding for Ethereum user profiling.

IV. Data collection

Our experiments were motivated by a Twitter movement that gained momentum in November 2019. Many ENS users posted their ENS name on their Twitter profile in order to facilitate transactions for those who want to interact with them on the Ethereum network. By discovering the Ethereum addresses linked to the published ENS names, we can quantitatively evaluate profiling techniques.

We collected addresses related to regular users and not automatic (trader or exchange) bots from the following publicly available data sources. **Twitter:** By using the Twitter API[3], we were able to collect 890 ENS names included in Twitter profile names or descriptions, and discover the connected Ethereum addresses, see Figure 1. **Humanity DAO:**[4] 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 up to 2020-04-04. Table I 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, see the last row of Table I.

By using the Etherscan blockchain explorer API, we collected 1,155,188 transactions sent or received by the 4259 regular user accounts in our collection (see Table I). The final transaction graph contains 159,339 addresses, and the transactions span from 2015-07-30 until 2020-04-04. Figure 2 shows the average number of transactions sent and received only by the accounts in the three data sources. Addresses collected from Twitter and Humanity DAO have similar characteristics, while Tornado accounts have fewer transactions since Tornado Cash has only recently been launched at the time of writing.

Due to ethical considerations, we took several steps to anonymize our published dataset. For example, we transformed ENS names to hash values and we also removed links to real-world identities, e.g., Twitter accounts.

[3]Using the Twitter Search and People API endpoints, we collected tweets containing the following keywords {'@ensdomains','.eth','ENS name','ENS address', 'ethereum', '#ethereum'} as well as profiles with an ENS name in their displayed profile name or description. We also searched for ENS names in the name and description of every tweeter in our data. Twitter data collection lasted from 2019-11-15 until 2020-03-05.

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

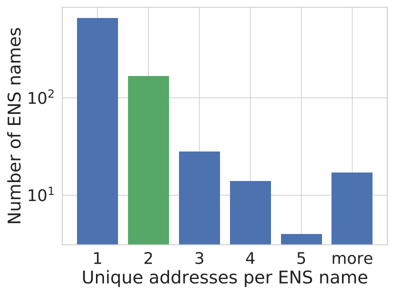

Fig. 1. Unique address count of ENS names collected from Twitter. Most of the ENS names in our collection are linked to a single Ethereum address, while some entities use multiple accounts. In Section VI, we use ENS names with exactly two unique addresses **(green)** to measure the performance of different profiling techniques.

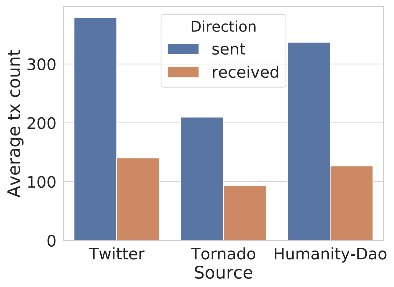

Fig. 2. Average number of transactions sent or received by the addresses of each data source. Tornado accounts have less transactions as the service has only recently been launched.

V. Evaluation measures

In this paper, we propose deanonymization methods, i.e., pairing Etherum addresses of the same user (Section VI), Tornado deposits and withdrawals (Section VII). 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 [37].

The simplest metrics consider the success rate of a deanonymizing adversary. Metrics such as accuracy, coverage, the fraction of correctly identified nodes [20], [22] 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 limited public information to rank candidate pairs and quantify the leaked information as the 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* [21] 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 [15]. In our experiments, we will compute AUC by the following claim:

Lemma V.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 [11]. □

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 the Tornado Cash 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 [32]. Several papers compute the entropy of the a posteriori knowledge [8], [21], [32]. However, they assume that the deanonymizer outputs a probability distribution of the candidate records [21].

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 [8], 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

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

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

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

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.

VI. Linking 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 investigate three quasi-identifiers for each account: the time-of-day activity, the gas price selection as well as a low-dimensional vector space representation that represents the Ethereum transaction graph structure.

A. Ground truth data

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 1. We consider 129 ENS names with exactly two Ethereum addresses to avoid the possible validation bias caused by ENS names with more than two addresses. We note that Ethereum addresses connected to multiple ENS names were excluded from our experiments.

B. Time-of-day transaction activity

Ethereum transaction timestamps reveal the daily activity patterns of the account owner. In the top row of Figure 3, 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 time-of-day activity histogram divided into b_{hour} bins.

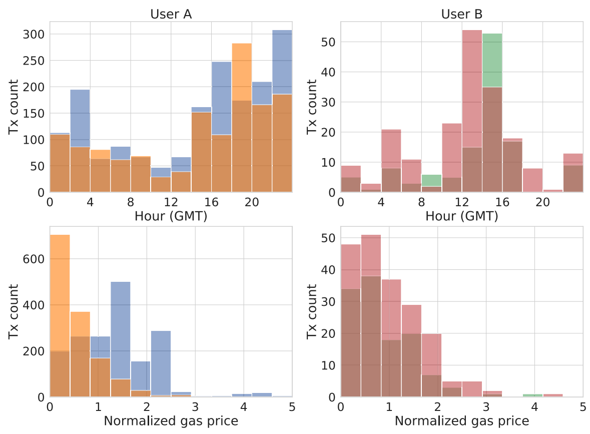

Fig. 3. Time-of-day and normalized gas price profiles for two ENS names (*User A*, *User B*) with a pair of addresses each. Both the time-of-day and gas price selection are similar in case of *User B* addresses (red, green) while the addresses of *User A* (blue, orange) have different gas price profiles. Addresses are denoted by different colors.

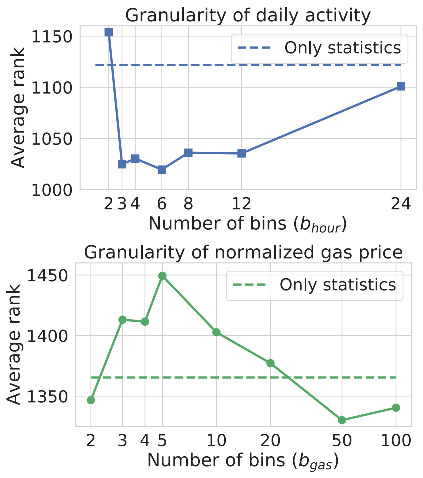

Fig. 4. Average rank at different granularity for daily activity **(top)** and normalized gas price **(bottom)** quasi-identifiers. **Dashed lines** show performance with only mean, median and standard deviation used.

C. Gas price distribution

Ethereum transactions also contain the gas price, which is usually automatically set by the wallet software. 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 affects user gas prices. Hence, we normalized the gas price by the daily network average. In the bottom row of Figure 3, we show normalized gas price profiles for two ENS names.

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

D. Graph representation learning

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.

Our deanonymization experiments are conducted on a transaction graph with nodes as Ethereum addresses and edges as transactions. We deployed twelve node embedding methods from the library[5] of Rozenberczki et al. [28] to discover address pairs that might belong to the same user.

To compute the node embedding, certain graph preprocessing steps are required. First, we considered transactions as undirected edges and removed loops and multi-edges. We also excluded nodes outside the largest connected component. Due to the data collection techniques described in Section IV, our transaction graph has two main components. The majority of the nodes have only one transaction, hence they reside on the periphery of the network, see Table I. On the other hand, the core of the network contains the popular Ethereum services. Since the periphery adds little to the graph structure, we excluded nodes with degree one before training the node embedding models.

The resulting graph has 16,704 nodes and 132,231 edges. We generated 128-dimensional embeddings for the nodes (addresses), which is the standard setting for graph representation learning tasks [26], [34]. In order to compare with timestamp and gas price representations, we assign the overall average of the network embedding vectors to the removed nodes.

E. Evaluation

Based on timestamp, gas price distributions or network embedding, we generate Euclidean feature vectors for 3321 Ethereum addresses with each having at least five transactions sent, see Table I. Given a target address, we order the remaining addresses by their Euclidean distance from the target with respect to their representations.

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

The average performance of the twelve different node embedding algorithms is shown in Table II based on ten independent experiments. The two best performing methods are Diff2Vec [29] and Role2Vec [2]. Note that these algorithms capture different aspects of the same graph. Diff2Vec is a

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

	Average rank	AUC	Entropy gain
Combined	492.418217	0.851681	1.616420
Diff2Vec	548.859302	0.834681	1.447579
Role2Vec	646.332946	0.805321	1.175364
DeepWalk	701.619380	0.788669	0.928413
Walklets	777.676357	0.765760	0.781871
NetMF	848.343798	0.744475	0.630712
BoostNE	891.038372	0.731615	0.528663
GraRep	1129.914341	0.659664	0.360408
Laplacian Eig.	1257.755039	0.621158	0.273857
HOPE	1352.750388	0.592545	0.150919
NodeSketch	1387.583721	0.582053	0.111869
NMF-ADMM	1454.363566	0.561939	0.132965
GraphWave	1626.702713	0.510029	0.035338
Daily activity	1019.546512	0.692908	0.387737
Norm. gas price	1330.155039	0.599351	0.115192

TABLE II

PERFORMANCE OF DIFFERENT NODE EMBEDDING MODELS AND SIMPLE TIME-OF-DAY AND GAS PRICE ACTIVITY-BASED REPRESENTATIONS IN THE ETHEREUM ACCOUNT LINKING TASK. FOR AUC AND ENTROPY GAIN THE HIGHER THE SCORE THE BETTER THE PERFORMANCE, WHILE IT IS THE OPPOSITE FOR THE AVERAGE RANK PERFORMANCE MEASURE.

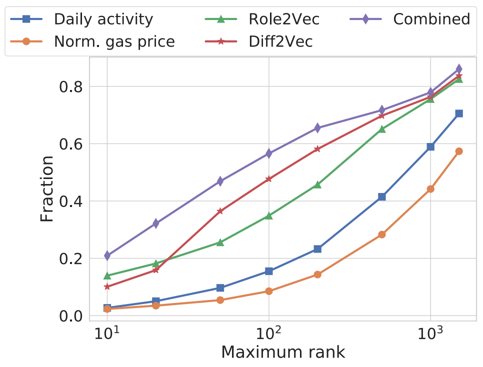

Fig. 5. Fraction of ENS address pairs correctly identified within a given maximum rank, for different embedding methods.

neighbourhood preserving model that performs strongly in community detection tasks [29]. In contrast, Role2Vec encodes the motif structure of the Ethereum addresses to successfully infer their function or role in the transaction graph. We achieved the best Ethereum address linking performance by the combination of Diff2Vec and Role2Vec, taking the harmonic average of the ranks for each account in the two candidate lists. Thus the combination of Diff2Vec and Role2Vec showed that different addresses of the same entity are usually in the same cluster or community performing similar roles.

In Figure 5, 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. Our combination based approach further improves the performance.

Our results show that the proposed profiling techniques link Ethereum addresses of the same user significantly better than random guessing. More precisely, the combination of Diff2Vec and Role2Vec yield 1.6 bits of additional information on account owners, see entropy gain results in Table II. In other words, we can reduce the anonymity set of a particular address by a factor of $2^{1.6} \approx 3.0314$.

In general, node embedding algorithms turn out to be more powerful for the Ethereum address deanonymization task than less elaborate time-of-day and gas price profiles. The latter techniques are only reliable if there are not many users in the same time zone or gas price profile. In contrast, graph representation learning can extract more complex descriptors that identify entities better.

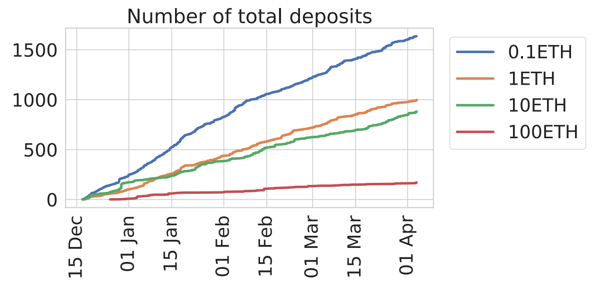

Fig. 6. The number of total deposits in each TC mixer from December 2019 to April 2020. This is an upper bound for the achievable anonymity set size when a withdraw transaction is executed. The popularity of the 0.1ETH mixer is superior compared to higher value mixers.

VII. DEANONYMIZING TRUSTLESS MIXING SERVICES

Privacy-enhancing tools became crucially important gadgets in the Ethereum ecosystem. The most popular is Tornado Cash (TC), a non-custodial zkSNARK-based mixer. The TC mixers are sets of Ethereum smart contracts allowing users to enhance their anonymity.

Cryptocurrency mixers typically provide k-anonymity to their users [30]. 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 6, 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).

Unfortunately, careless usage easily reveals links between deposits and withdrawals and also impact the anonymity of other users, since if a deposit can be linked to a withdraw, it can be excluded from the anonymity set of other withdraws.

Mixer	Deanonymized withdrawals Pattern 1	 Pattern 2	 Pattern 3	 Total	All withdrawals
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 III
NUMBER OF ALL WITHDRAWALS AND DEANONYMIZED WITHDRAWALS USING THE CORRESPONDING PATTERNS IN EACH MIXER CONTRACT.

The simplest careless usage is applying the same address for deposit and withdrawal transactions as well:
Pattern 1. *If there is an address from where a deposit and also a withdrawal has been made, then we consider these deposits and withdrawals linked.*

A. Ground truth data

Next, we define two more patterns we use for defining ground truth data for our machine learning methods that find linked pairs of accounts. Note that Pattern 1 finds a deposit-withdraw pair of the same address rather than an address pair. Hence, the anonymity guarantees are already broken without the need of machine learning methods.

The next pattern consists of the use of a salient gas price, which can be used to define linked address pairs in a ground truth set to evaluate our machine learning methods. Most wallet software, 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[6] at block height $9,418,956$ with 5.130909091 Gwei gas price. Later on, there is a withdraw transaction[7] at block height $9,419,096$ with exactly the same custom-set gas price.
Pattern 2. *If there is a deposit-withdraw pair with* unique gas prices*, then we consider them as linked.*

In a third pattern, users reveal links between their deposit and withdraw addresses if they sent transactions from one of their addresses to another address owned by them. We conjecture that users falsely expect that withdraw addresses are clean. Therefore, they 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 as follows.
Pattern 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.

We found 218, 110, 60, and 7 withdrawals linked by Patterns 1–3 in the four mixer contracts (0.1 ETH, 1 ETH, 10 ETH, 100 ETH), respectively, up to April 4th 2020, see Table III. We note that withdrawals identified by Pattern 2 can overlap with other withdrawals identified by Pattern 1 or 3. Hence the number of total linked withdrawals are less than the sum of all withdrawals individually identified by each pattern.

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

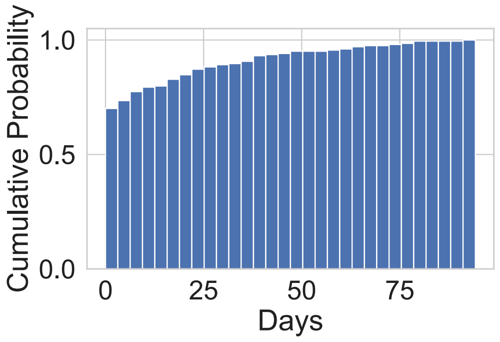

Fig. 7. Elapsed time in days between linked deposit and withdraw transactions for the 0.1 ETH mixer contract. Vast majority of users do not wait more than one day to withdraw their deposits.

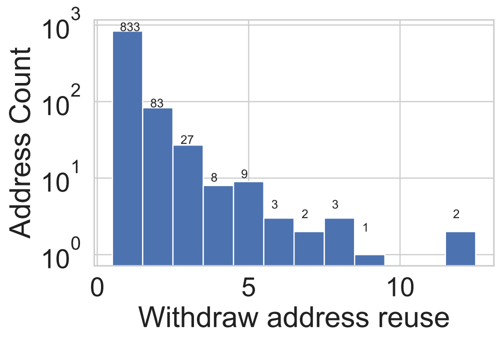

Fig. 8. Withdrawal address reuse in the 0.1 ETH mixer contract. Many users withdraw multiple deposits to the same address, which eases deanonymization and reduces the privacy properties of the mixer.

B. Elapsed time between deposits and withdrawals, withdraw address reuse

In Figure 7, 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 8, we observe several addresses receiving multiple withdrawals from the 0.1 ETH mixer contract.

For instance, there are 83 addresses that have withdrawn twice and 27 addresses with 3 withdrawals each. This phenomenon causes privacy risk not just for the owner of these addresses but also reduces the privacy properties of the mixer. Proper usage always requires a withdraw to a fresh address.

C. Deanonymization performance

Next, we measure how well the techniques of Section VI identify the linked withdraw-deposit address pairs. We build ground truth by using careless Tornado usage Patterns 2–3. 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. For example, for the 0.1 ETH mixer contract, we found 100, 80 and 67 deposit-withdraw pairs for these ground truth sets, respectively. Experiments on the unfiltered full set are labelled *past* in Figures 9-10 and Table IV.

Note that in Pattern 2, we used gas prices. Hence, in this section, we include measurements for gas price based linking performance only as reference. Similarly, Pattern 3 relies on an edge (transaction) between the two addresses, hence we discard these edges (transactions) in the network analysis algorithms. As we see, gas price distribution performs weakly for finding the account pairs, despite that Pattern 2 is based on gas price. On the other hand, adding the edges between accounts identified by Patterns 3 would yield misleadingly strong performance, since the same information is used for defining the ground truth and for testing.

Table IV shows that an address with withdraw within a day or week has a significantly smaller anonymity set size, on average, since we only search for the corresponding deposit in a smaller set. In the 0.1ETH mixer, the original average anonymity set size of 400 can be reduced to almost 12 by assuming (e.g., an adversary might obtain this information by any means as background knowledge) that the deposit occurred within one day of the withdraw.

Daily activity and Diff2Vec perform much better in the TC withdraw linking task than gas price based user profiles. For the smaller week and day 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 representations of daily activity and Diff2Vec. The number of withdrawals linked to deposits within a given rank of the candidate list for different methods are in Figure 9. The concatenated model identifies almost twice as many withdraw deposit pairs than Diff2Vec for the unfiltered ground truth set.

In Figure 10, we show the withdraw linking performance over time. As the number of active deposits increases, it becomes harder to link withdrawals to any of the past deposits. However withdrawals that follow the deposit after a few days are still much easier to deanonymize.

VIII. Maintaining privacy

In this section, we propose countermeasures to mitigate the effect of the main privacy leaks. Section VIII-1 and VIII-2 are strictly related to non-custodial mixers while VIII-3 is a more general discussion on user behavior reflecting our results in Sections VI-E and VII-C.

1) Randomized mixing intervals: Participants in mixer contracts greatly decrease anonymity if they withdraw funds after short time intervals, cf. Figure 7 and Table IV. A possible workaround is the use of randomized mixing intervals. Such a delay in withdrawal cannot be enforced by the mixing contract itself, since withdrawals cannot be linked to the deposits. Thus, delaying should be accomplished by the user wallet software.

2) Fresh withdraw addresses: Currently, many users apply the same withdraw addresses across several withdrawals, see Figure 8. This greatly decreases the complexity of linking deposits and withdrawals. Therefore, users must use fresh withdraw addresses for each of their withdrawals.

3) Mixer usage and user behaviors: Mixers mainly attempt to break the link between accounts associated with the same entity. 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.

Making deposits and withdrawals always during the same time of the day can also decrease the anonymity set for an account. An option in the user wallet software to schedule transactions for the future with random time delay could easily hide the time zone for the given entity.

Unfortunately, fooling complex graph representation learning algorithms is harder and requires constant monitoring of the whole Ethereum transaction graph, which the users themselves will unlikely be able to perform. For example, fooling Diff2Vec representations can require carefully chosen transactions to other address clusters. Similarly, several transactions are needed to change the motif profile to mislead Role2Vec. Companies or other large entities could probably act to obfuscate the network structure, but eventually, the end users will have to pay the price for the additional transactions. Finally, we note that using payments with negligibly small amounts to artificially link unrelated addresses are insufficient for obfuscation, as such transactions can be easily ignored in a privacy attack.

IX. Conclusion and Future Directions

In this paper, we studied how graph representation learning, time-of-day activity and gas price profile can be used to link Ethereum addresses owned by the same user. The Ethereum Name Service (ENS) relations in our data set provided ground truth information to quantitatively compare and analyze the performance of these quasi-identifiers. Our results showed that recent node embedding methods had superior performance compared to user activity based profiling techniques.

As an application, we applied these profiling techniques on recently deployed privacy-enhancing overlays, such as Tornado Cash (TC) mixers. By our measurements, their decreased usability and immature user behavior prevent them from reaching their highest attainable privacy guarantees. Evaluation

Evaluation metric: Withdraw within:	Average rank past	week	day	Entropy gain past	week	day
Norm. gas price	168.784483	28.224719	5.861111	0.145438	0.134766	0.095546
Daily activity	116.146552	18.213483	3.444444	0.526595	0.570671	0.560121
Diff2Vec	89.222414	17.968539	4.268056	0.899869	0.743705	0.491611
Concatenated	64.012069	13.241573	3.362500	1.238389	1.151222	0.840526
Avg. anonymity set size	400.672414	70.247191	12.291667			

TABLE IV
WITHDRAW LINKING PERFORMANCE FOR THE 0.1ETH MIXER CONTRACT. ENTROPY GAIN AND THE AVERAGE RANK OF THE DEPOSIT ADDRESS IN THE CANDIDATE LIST OF OUR ALGORITHMS ARE SHOWN FOR THE THREE DIFFERENT GROUND TRUTH SETS (PAST, WEEK, DAY), CF. SECTION VII-C.

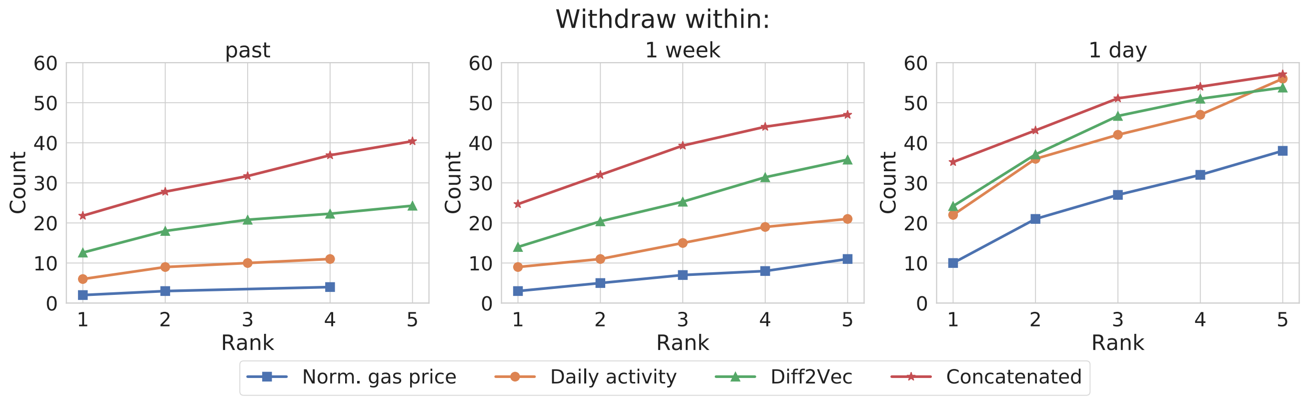

Fig. 9. Number of withdraw addresses in the 0.1ETH mixer contract such that the corresponding deposit is identified within the given rank in the candidate list of each deanonymization technique, separate for the three ground truth sets described in Section VII-C.

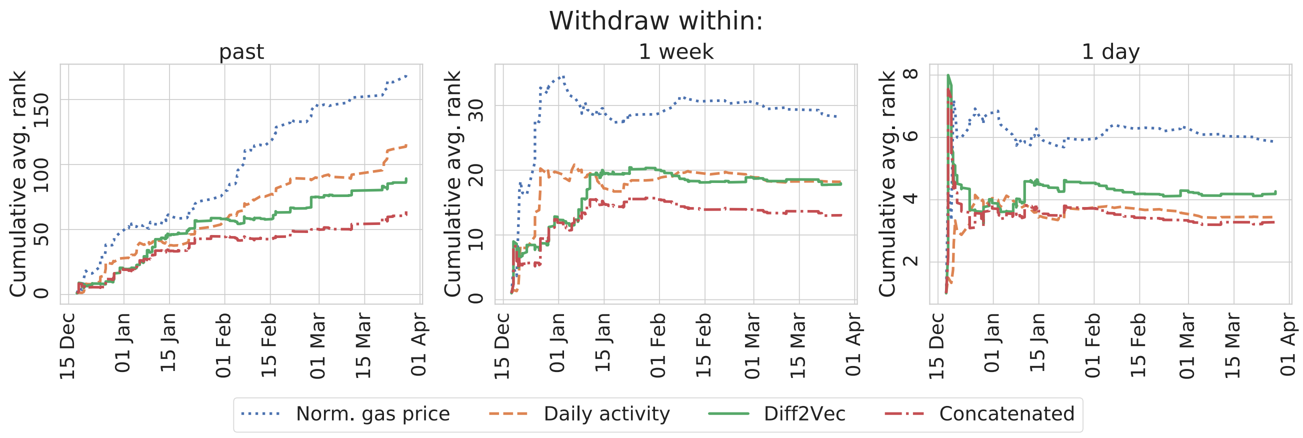

Fig. 10. Change of average rank in time, cumulated from the beginning of our data (December 2019), for the 0.1 ETH Tornado mixer by using our best deanonymization methods. Results are showed separately for the three ground truth sets described in Section VII-C.

on different ground truth sets compiled from careless usage patterns of the TC mixer showed that profiling techniques, especially novel node embedding algorithms, can significantly reduce the anonymity set sizes of the mixing parties.

We perceive three major directions for future work. First, it would be valuable to find and evaluate further quasi-identifiers for Ethereum address profiling tasks. Second, in this work, we solely relied on blockchain data to profile and deanonymize Ethereum users. Studying privacy on the peer-to-peer network is paramount to fully assess privacy provisions of Ethereum. Finally, systemtaically describing privacy leakage on the application- and browser-level would be fundamental.

We release the collected data as well as our source code to facilitate further research[8].

Acknowledgments

We thank Daniel A. Nagy, David Hai Gootvilig, Domokos M. Kelen and Kobi Gurkan for conversations and useful suggestions. The research was supported by the Ministry of Innovation and Technology NRDI Office within the framework of the Artificial Intelligence National Laboratory Program and by Project 2018-1.2.1-NKP-00008: Exploring the Mathematical Foundations of Artificial Intelligence.

References

[1] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J. Smola. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, page 37–48, New York, NY, USA, 2013. Association for Computing Machinery.

[2] 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.

[3] 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.

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

[5] 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.

[6] 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.

[7] 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.

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

[9] Steven Goldfeder, Harry A. Kalodner, Dillon Reisman, and Arvind Narayanan. When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies. *CoRR*, abs/1708.04748, 2017.

[10] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[11] 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.

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

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

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

[15] 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.

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

[17] 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.

[18] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA, 2013. Curran Associates Inc.

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

[20] 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.

[21] 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.

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

[23] 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.

[24] 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.

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

[26] 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.

[27] 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.

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

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

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

[31] 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.

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

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

[34] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, page 1067–1077, Republic and Canton of Geneva, CHE, 2015. International World Wide Web Conferences Steering Committee.

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

[36] 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.

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

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

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

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