Towards Malicious address identification in Bitcoin

Deepesh Chaudhari, Rachit Agarwal, Sandeep Kumar Shukla
CSE, IIT Kanpur, India
Email: {deepesh,rachitag,sandeeps}@iitk.ac.in

Abstract

The temporal aspect of blockchain transactions enables us to study the address's behavior and detect if it is involved in any illicit activity. However, due to the concept of change addresses (used to thwart replay attacks), temporal aspects are not directly applicable in the Bitcoin blockchain. Several pre-processing steps should be performed before such temporal aspects are utilized. We are motivated to study the Bitcoin transaction network and use the temporal features such as burst, attractiveness, and inter-event time along with several graph-based properties such as the degree of node and clustering coefficient to validate the applicability of already existing approaches known for other cryptocurrency blockchains on the Bitcoin blockchain. We generate the temporal and non-temporal feature set and train the Machine Learning (ML) algorithm over different temporal granularities to validate the state-of-the-art methods. We study the behavior of the addresses over different time granularities of the dataset. We identify that after applying change-address clustering, in Bitcoin, existing temporal features can be extracted and ML approaches can be applied. A comparative analysis of results show that the behavior of addresses in Ethereum and Bitcoin is similar with respect to in-degree, out-degree and inter-event time. Further, we identify 3 suspects that showed malicious behavior across different temporal granularities. These suspects are not marked as malicious in Bitcoin.

Keywords— Suspect Identification, Bitcoin, ML

1 Introduction

Blockchain technology was first introduced with the introduction of Bitcoin by Satoshi Nakamoto in 2008 [13] to increase the trust and transparency of users by shifting the controlling power from central authority to a decentralized network. This also allowed the users to transact anonymously and without any fear of being traced. In the Bitcoin blockchain, users transact using cryptocurrency called Bitcoin (BTC). Due to these features, cyber-criminals quickly adopted Bitcoin to perform malicious activities such as *ransomware attacks*, *money laundering*, *gambling*, *ponzi schemes*, and *phishing* [10]. With increased adoption of blockchain by different industries, such activities have grown to cause more financial losses. In [7], the authors identify that in the year 2020, illicit entities made ≈$10 Billion worth of transactions.

With more and more security threats appearing with the use of blockchains, we ask *whether malicious user's addresses can be detected in a blockchain using approaches such as machine learning (ML)*? Many state-of-the-art algorithms aim to detect malicious addresses in Bitcoin, Ethereum, and other permissionless blockchains. These algorithms use either *(a)* graph-based techniques [20, 2] or *(b)* ML algorithms [15, 12, 8, 5]. To perform ML-based detection, it is important to understand how malicious activities are different from benign activities. Temporal aspects inherent in the users's behavior eases such understanding. Nonetheless, such temporal aspects are rarely used by the state-of-the-art approaches and no feasibility study is performed to identify whether the features apply to the blockchain or not. For example, in the Bitcoin blockchain, to receive the remaining balance after a transaction, a new address of the address is generated (also called as change address). Such aspects allow a user to hold multiple addresses. In such a scenario, the application of temporal features such as in-degree would be instantaneous.

Hence, the research questions (RQ) that we ask are: **(RQ1)** Can we detect malicious addresses in Bitcoin using ML techniques and consider the temporal behavior? **(RQ2)** Are the features identified in state-of-the-art approaches targeting blockchains such as Ethereum applicable in Bitcoin? As there are change addresses in Bitcoin, **(RQ3)** What changes occur if we cluster the addresses? Further, once addresses are clustered **(RQ4)** Does behavioral changes exist in Bitcoin? To answer these research questions, we create two graphs (**user/address** and **transaction** graph) from the Bitcoin transaction data, extract required features from these two graphs, and apply ML techniques to detect addresses we suspect to be malicious. We find that features such as *degree burst* (an abrupt rise in the degree of an address), *balance burst* (an abrupt increase in the balance of an address), and *attractiveness* (probability of

interaction with the same address that the address interacted with before [2]) are not directly applicable in Bitcoin because of the concept of change address. Note that for this work we focus on those malicious entities for which we have ground truth data. Our approach is agnostic to the type of malicious activity.

Change addresses are used to protect the privacy and increase the user's anonymity. Thus, features like degree burst, balance burst, and attractiveness that use temporal aspects of transactions are spread over different addresses. Calculating these features directly from the Bitcoin transaction data does not provide any helpful information about the address's behavior. To identify the addresses potentially held by the same user, in [14], the authors develop two heuristic approaches *(a)* **multi-input** heuristics (based on the assumption that if a transaction uses more than one address as an input, then they must be created and controlled by the same user) and *(b)* **change address** heuristic (addresses that are created by the same user to keep the change amount (remaining BTCs)). Note that there are other approaches (described in Section 2) that cluster the addresses, but we apply the multi-input and change address heuristics to cluster the Bitcoin addresses in this work as they provide the best results [14]. After using the heuristics, we find that the total number of users that transacted during the considered period reduces considerably.

Next, we segment the data using different temporal granularities (T_G) to understand the changes in the behavior. These temporal granularities include 15Days and 1Month granularities, i.e., T_G={15Days, 1Month}. Each data segment (SD) is mutually exclusive in these granularities and comprises of transactions in a given particular period defined by that segment. On these SDs, we first apply unsupervised ML (K-Means) to identify malicious addresses. For comparison, we then apply heuristics-based change address clustering on each of the SDs and apply unsupervised ML.

In summary, our main contributions are:

- ***Validation of the state-of-the-art methods***: We apply the state-of-the-art ML approaches to detect malicious addresses in the Bitcoin. We find that the features present in other works are applicable to Bitcoin, only after applying change-address heuristics. Using the Bitcoin transaction data, we verify that temporal features such as in-degree, out-degree and inter-event time follow either *power-law* distribution or its variations. Thus, proving existence of burst. We also find that using temporal features more suspects are detected.

- ***Comparative analysis***: Using KL divergence, we find that in Bitcoin, behavior of addresses in similar to those in Ethereum with respect to features such as in-degree, out-degree and inter-event time.

- ***Behavior analysis***: We perform the behavior analysis on Bitcoin addresses using the temporal and non-temporal features extracted from their transactions. Using temporal features we find a total of **3,273** and **19,712** addresses that change their behavior in the considered two different time granularities. Whereas, using non-temporal features we find a total of **868** and **159** addresses that change their behavior in the two different time granularities.

This paper is organized as follows. In Section 2, we present the state-of-the-art techniques for identifying malicious addresses and compare them. In Section 3, we give a detailed description of our methodology. This section is followed by an in-depth evaluation of the results in Section 4 that includes a description of the data. We finally conclude in Section 5 and also provide prospective future directions.

2   Related work

In this section, we first present different state-of-the-art approaches towards de-anonymizing the Bitcoin address and then present those approaches that tackle the identification of malicious addresses in the Bitcoin blockchain. Further, we also survey the state-of-the-art approaches from a more general perspective, especially those that deal with permissionless blockchains and claim that their approach applies to any permissionless blockchains. Most of the approaches towards identifying malicious addresses, either in Bitcoin or in permissionless blockchains, are ML-oriented, where the feature vector is based on graph and transaction analysis. Note that although there are lot of state-of-the-art approaches that use temporal aspects, we do not focus on them as they predict price of Bitcoin and not illicit entities.

2.1   De-anonymizing the Bitcoin addresses

Assume a transaction T_x is performed between input addresses (senders represented by the set $inputs(T_x)$) and output addresses (receivers represented by the set $outputs(T_x)$).

In [11, 16], the authors introduced the multi-input heuristics which assumes that a single user controls all the $input(T_x)$ of the T_x. A sender needs to sign all the inputs using the private keys to prove that the user owns the BTCs present in those addresses to make a valid transaction. Hence, the user that created the transaction must control all those private keys associated to $input(T_x)$. For a transaction T_x, consider the C_{T_x} to be the cluster of addresses controlled by one user. Then, using this method, $C_{T_x}=inputs(T_x)$.

In Bitcoin, when a user makes a transaction, he needs to transfer all the BTCs he has to receiver's address. If the user wants to transfer only some BTCs he has to another user's address, then he must create a new address

and send the remaining BTCs to that address. Since, the user controls the input addresses and this change address, these addresses should be clustered together. In [11], the authors proposed a heuristics-based method to identify such addresses. An address $o_j(T_x) \in outputs(T_x)$ is said to be a "change" address if:

1. The address $o_j(T_x)$ appears first time.
2. Transaction T_x is not a "coin-generate" transaction.
3. There must be no common address between output addresses and input addresses (No **self-change address**).
4. The first condition is only satisfied for $o_j(T_x)$ and not for other addresses in $outputs(T_x)$.

Thus, combining the two methods, for a transaction T_x, C_{T_x}=$inputs(T_x) \cup \{o_j(T_x)\}$.

In [16], the authors proposed two more conditions to capture the "*no*" change address scenario. A T_x contains no change address if there is an address in the output such that:

5) It has already received only one input.

6) It has appeared before in a self-change transaction.

In [14], the authors introduced a ***value-based heuristics*** method as an extension to the above conditions. They assumed that users try to minimize the size of the transaction by reducing the number of inputs and outputs. An $o_j(T_x)$ is expected to be a change address if the T_x contains only one output with a value less than its inputs. If the change amount was more than any input, then we could ignore the input. Further, in [14], the authors proposed a ***growth-based heuristics*** method and showed that clusters usually increase steadily. It's rare to see the amalgamation of two big clusters by a new transaction and might indicate a false positive from one of the heuristics.

In [19], the authors, besides using multi-input and change-address heuristics methods, use an address reuse-based heuristics method. Here, instead of previously defined condition '4', the authors proposed:

4') Output addresses except for $o_j(T_x)$ are reused as an output address in some later transactions.

In [14], the authors also compared the heuristics methods to know which method provides better results. They found that multi-input together with change address heuristics gives a more reliable cluster.

2.2 ML Based detection approaches

In [15], the authors used unsupervised learning to detect malicious addresses in Bitcoin. They used K-Means as a baseline method and calculate the *local outlier factor (LOF)*. For the feature extraction, they analyzed the Bitcoin transaction data and created two graphs, **user** and **transaction** graphs. They extracted features corresponding to graph properties such as inter-event time, balance, and activity period. Using transactions of 6.3 Million addresses, they found that for K=8, K-Means provided optimal results for both the user and the transaction graph. However, they did not mention how they create the user graph.

In a similar work, in [12], the authors compared the results of K-Means and trimmed K-Means on the Bitcoin transaction data of 1 Million addresses. However, they limited their study to graph-based and balance-based features. They found that both trimmed K-Means and K-Means provided optimal results when K=8 and thus also validated [15]. Nonetheless, they found that trimmed K-Means provided better results with an improved detection rate.

In [5], the authors used supervised ML algorithms to detect addresses involved in *Ponzi schemes* in Bitcoin. A Ponzi scheme is an investment fraud that generates returns to existing investors with funds raised from new investors. Here, the authors de-anonymized the Bitcoin addresses using the above mentioned heuristics methods. They, used a rule-based learner called RIPPER, Bayes Network, and Random-Forest classifiers on the features (such as graph properties, inter-event time, balance, and activity period) extracted from 6432 Ponzi scheme related addresses. They found that both RIPPER and Random-Forest classifier achieved high accuracy. However, the method only focused on Ponzi schemes and did not detect addresses involved in any other malicious activity.

In [20], the authors studied transactions between entities (organizations or people with multiple addresses) to attack Bitcoin address anonymity. They present a novel *cascading machine learning* approach that requires only a few features (graph-based, balanced-based, and transaction fee-based) extracted from transactions of 1000 Million addresses present in Bitcoin. Cascading, used to enrich an entity's information with data from previous classifications, considerably improves the multi-class classification performance for classes such as benign and malicious. As an initial step, the authors performed the classification of addresses into entities. From the data associated with these entities, they extract features and finally apply ML (especially Adaboost, Random-Forest, and Gradient Boosting models) to detect malicious addresses. For classification of addresses into entities, the authors used *C_address*, *C_motif1*, and *C_motif2*[1]. Here, C_address contains features such as the number of transactions related to a Bitcoin address.

---

[1]*N-motifs* are frequently occurring sub-graphs of a network. In [20], the authors define *N-motif* as a sub-graph of maximum *2*N* path length between two entities, where transactions are also considered as vertices.

C_motif1 contains the information such as the number of transactions extracted from the *1_motif* graph while C_motif2 contains information such as the number of transactions gathered from the *2_motif* graph. Overall, the Gradient Boosting model provided the best classification scores when C_address, C_motif1, and C_motif2 are used together.

In [18], the authors used supervised learning to classify 957 Bitcoin addresses that performed 385 Million transactions. They used heuristics-based algorithm first to cluster the Bitcoin addresses. Then, they used the clustered data to train the ML model and classify the addresses into 12 categories (exchange, merchant-services, mixing, gambling, personal-wallet, mining-pool, ransomware, hosted-wallet, scam, darknet-market, stolen-Bitcoins, other). They used various supervised learning algorithms and found that Gradient Boosting gave the best result with a 79.64% F1-Score. However, the authors do not comment on the bias induced by the address clustering heuristics while identifying malicious addresses and perform oversampling using SMOTE. On the contrary our work does not oversample to achieve better results.

In [4], the authors introduced *Topological Data Analysis* (TDA) to train the ML model to detect ransomware-associated addresses in Bitcoin. The concept of TDA is to extract pattern hidden inside the data. The authors found that TDA performs best amongst different ML algorithms. They trained the algorithm using 6 features (total incoming balance, address from which the address has received BTC, sum of fraction of coins that reach the address, longest path to that address, number of addresses that have a path to the address, and number of address that can reach the address via multiple paths).

Given such state-of-the-art techniques to identify malicious addresses in Bitcoin, in [2], the authors identified that these above-mentioned state-of-the-art approaches do not use temporal aspects to identify malicious addresses. Thus, they analyzed the temporal behavior of transactions present in the blockchain using temporal graphs and introduce new temporal features (such as burst and attractiveness) to detect malicious addresses. They studied the behavior of 700,000 addresses and applied supervised and unsupervised learning approaches. They found that ExtraTreesClassifier performed the best with balanced accuracy >87%. While for the unsupervised learning case, they found that K-Means with $K=10$ performed the best. Although the authors validated their work on Ethereum transaction data, they claimed that their features are applicable to all the permissionless blockchains. For such a generic claim, where different permissionless blockchains have different architecture and implementations, such aspects need to be validated on a wider set of permissionless blockchains.

In summary, all the state-of-the-art approaches have bias concerning the available ground truth. Except for [20, 5], all the approaches consider all the addresses associated with malicious activities under one class. As malicious activity-specific data is hard to find for Bitcoin, we also focus on identifying malicious addresses without associating them with any malicious activity. However, in Ethereum, such information is available, which authors in [3] leveraged to study how the ML approaches perform with respect to different activities. However, as we do not focus on Ethereum, we refrain from explaining them in this work and only focus on Bitcoin.

3 Methodology

In this section, we discuss an outline of our methodology and the process pipeline. Here, we first describe how we pre-process transaction data to convert it. Then, we discuss the feature extraction, data configurations, and the ML techniques we use in our work.
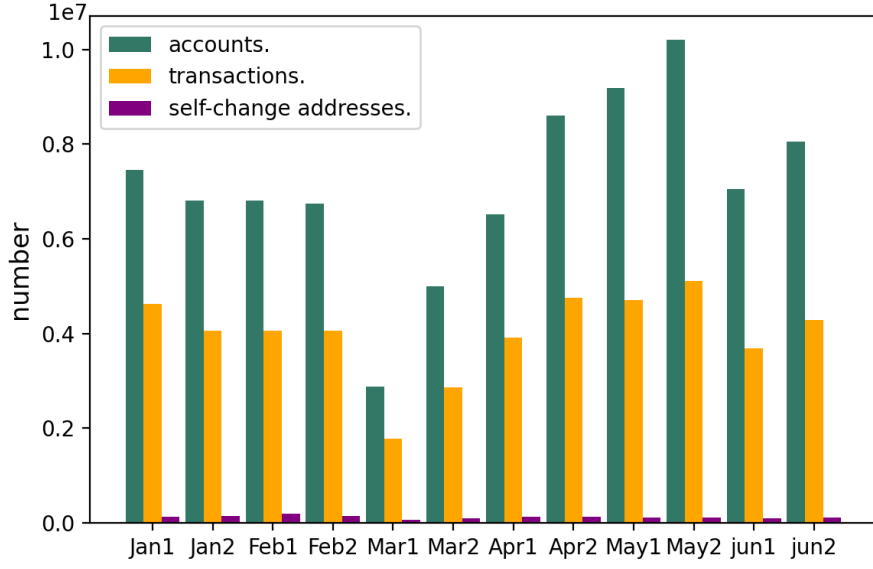
3.1 Pre-processing and Graph generation

A collection of Bitcoin transactions represents a temporal-directed social interaction graph. From this, we extract **Aggregated graph/Transaction graph** $(AG^t(V^t, E^t))$ and we convert it into **User graph** $(UG^t(V^t, E^t))$ to extract true nature of features such as indegree and outdegree. Here t is defined as block number in our case. Note that these graphs are temporal as edges or interactions are intermittent causing graph properties to change over time. In $AG^t(V^t, E^t)$, V^t is a set of two types of addresses, one that belong to users and another to transactions at a time t. Every transaction node input and output is to addresses that belong to users. On the other hand, in $UG^t(V^t, E^t)$, V^t is a set of addresses that belong to users at a time t, and E^t is the set of edges or transactions between these addresses at a time t. To create these graphs, we neglect those transactions in which users transact using *op_return* or transact with *script hash* because output addresses are not available in such cases.
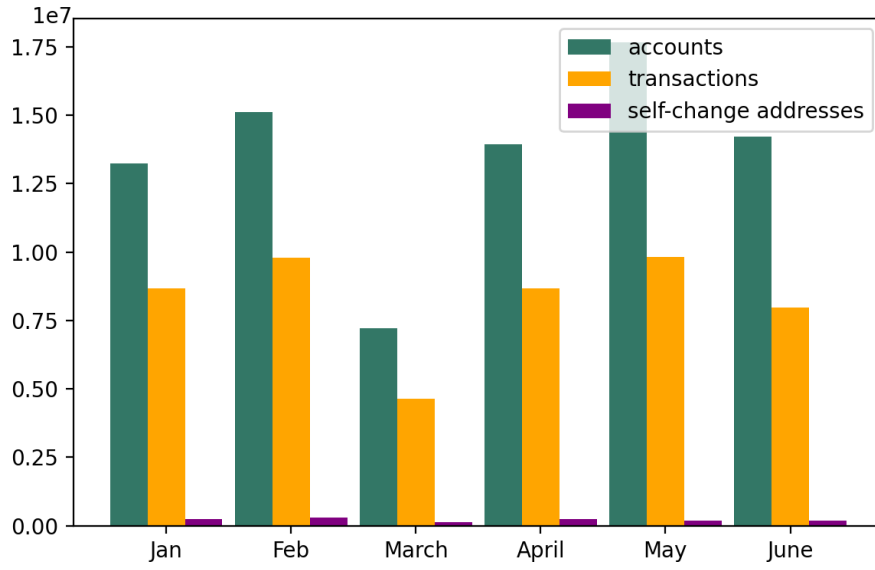
In Bitcoin, due to the concept of change address, each user has multiple addresses. Such aspects make it hard to extract the temporal behavior of the user. To know what addresses belong to which users, many studies use heuristics-based approaches to cluster addresses. We also apply the heuristics approaches mentioned in Section 2.1 to identify addresses held by a user to both the user and aggregated graph.

3.2 Features Extraction

We generate our training and test dataset from $UG^t(V^t, E^t)$ and $AG^t(V^t, E^t)$ and extract two types of features: non-temporal features and temporal features as introduced in [2]. We discuss the non-temporal and temporal features

(a) 15Days.

(b) 1Month.

Figure 1: Distribution of transactions, addresses, and self-change addresses when considering different temporal granularities. Here, 'Month*' represent whether it is first 15Days or second 15Days of the month.

next.

3.2.1 Non-Temporal features

are those that are independent of time. We classify non-temporal features into those that are based on graph properties and balance.

- **Graph properties based features**: The graph properties are one of the best ways to characterize addresses. For example, *in-degree* of an address shows the number of transactions the address has performed as a receiver, and *unique in-degree* indicates the number of transactions that address performed as a receiver with unique addresses. Most malicious addresses receive BTCs from different addresses and have a higher number of transactions than benign addresses [2]. This makes in-degree and unique in-degree an important feature to differentiate between malicious and benign addresses. Similarly, when an address acts as a sender, *out-degree* and *unique out-degree* features are important. Most benign addresses prefer to perform transactions with already known addresses and pose a relatively less unique out-degree than malicious addresses for whom the neighborhood is rather less stable. This also impacts the *Clustering-Coefficient* feature (which defines how tightly addresses are coupled with the neighbor addresses [17]). Nonetheless, we also include the total number of transactions in which an address is involved as *total degree* and the total number of transactions performed with unique addresses as the *unique total degree.*

- **Balance-based feature**: The amount of funds transferred/associated with a transaction plays an important role in determining whether an address is malicious or not. Malicious addresses at times transact large amounts as compared to benign addresses. Thus, features such as maximum payment received (*max_in_payment*), maximum payment sent (*max_out_payment*), the total amount received (*in_balance*), total amount sent (*out_balance*), and the balance of the address (*txn_sum*) are essential features. Nonetheless, most malicious addresses are less active as compared to benign addresses. Thus, features such as when the address first transacted (*first_txn_block*), last transacted (*last_txn_block*), and how long an address was active are essential features towards classifying an address as malicious.

3.2.2 Temporal Features

Due to the temporal nature, graph properties evolve and are not static. It is essential that the properties are not studied as static but studied as dynamic properties. One should extract features considering time to represent them better. In [2], the authors identified features such as *inter-event time* (time interval between two transactions of an address), *attractiveness* (the probability of an address interacting with previously unknown address), and *bursty* (abnormal rise and fall in the value of the property) behavior of *in-degree*, *out-degree*, and *balance* as important features that distinguish malicious and benign addresses. As these features are temporal, we use time-series analysis to extract top 3 representative features of the above mentioned features using *tsfresh* [9]. Using such features, we also validate their applicability in Bitcoin to identify malicious addresses.

3.3 Data Configuration

We segment the data into several sub-datasets (SD) of different time granularities (T_g) to study the behavior of an address. Due to the limited availability of computational resources, we currently validate our approach on only 2 temporal granularities: 15Days and 1Month. Here, each granularity has n different SDs that are mutually exclusive. For each SD, in each temporal granularity, we further generate 3 different datasets ($SD^{i,j}(T_g)$ where i is the ith SD in a T_g and $j \in \{1, 2, 3\}$). Here, $SD^{i,1}(T_g)$ is the dataset with only non-temporal features, $SD^{i,2}(T_g)$ is the dataset generated after clustering addresses and contains only non-temporal features, and $SD^{i,3}(T_g)$ is the dataset generated after clustering addresses and contains both temporal and non-temporal features. Note that $SD^{i,j}(T_g) \in SD^{j}(T_g)$.

3.4 Machine Learning

After generating different SDs mentioned in Section 3.3, we apply results (K-Means with $K=10$) of [2] for unsupervised learning and validate it. Other supervised approaches like DNN or ensemble based methods although can provide better results but are out of the scope of this work. After clustering, we take the cluster with the most number of malicious addresses and compute their cosine similarity (CS) with other benign addresses in the corresponding cluster. CS is one of the most widely used measure that was also used in [2]. We thus use the same method as we want to validate the approach. Such approach allows us to be within our computational constraints and label only a few addresses as potential suspects. To find benign addresses with high cosine similarity with malicious addresses, we study the variation of the number of benign addresses tagged as malicious while changing the ϵ parameter. Here,

ϵ is the threshold that identifies which benign addresses should be marked as suspected to be malicious. Formally, we mark benign addresses if equation 1 is satisfied. We evaluate for $\epsilon \in [0, 20]$.

$$k^{label} = \begin{cases} malicious, & \text{for any } l \in X : \ 1 - CS(k, l) \leq 10^{-\epsilon} \\ benign, & \text{otherwise} \end{cases} \tag{1}$$

Here, k is the benign address and X is the set of malicious addresses. After identifying suspects in all the SDs, we calculate the probability p^k of a particular address k to be malicious in a given T_g as stated in [2]. Here, $p^k = \frac{\sum_{i \in SD^j(T_g)} M_i^k}{n^k}$ where, M_i^k depicts whether the address k was identified as suspect in the ith sub-dataset $(SD^{i,j}(T_g))$ using j features at temporal granularity T_g, and n^k is the total number of SDs in which address k transacted.

4 Evaluation: Data and Results Analysis

We evaluate the effectiveness of the state-of-the-art method using Bitcoin's transaction data. We set up a Bitcoin node[2] and fetch transactions using Bitcoin RPC API[3]. Note that the APIs do not provide sensitive information about the users (such as the name, date of birth, and account type). Although as the address's hash is publicly available, one can look up the associated information using any Blockchain explorer such as Blockchain.com[4]. We perform all our evaluations using Python3 and scikit-learn.

4.1 Data

In Bitcoin, more than 650 Million transactions are performed till June 2021 [6], with the highest being 400,000 transactions per day in January 2021 [1]. On average, in Bitcoin, a new block is created every 10 minutes. For our experiments, we choose Bitcoin transaction data from 1th January 2020 till 30th June 2020 due to computational constraints. During this period, 60,963,231 unique addresses performed 49,244,236 transactions. Figure 1 show the number of transactions performed and the number of addresses that transacted at different periods that we consider. We see a dip in the number of transactions and the number of addresses that performed the transactions in March. This could be due to the rise of the COVID-19 pandemic around the world. We use databases such as Bitcoinabuse[5] and Cryptoscamdb[6] where malicious tagged Bitcoin addresses are publicly available. As of $27^{th} Feb$ of 2021, there were 54,036 maliciously tagged Bitcoin addresses and belong to types such as "Scams" and are reported by other users. However, in this work we do not consider the types of the malicious addresses rather consider them as malicious only.

4.2 Results Analysis

For a clear understanding, we first answer RQ2 and then proceed to answer RQ1, RQ3, and RQ4. Note that we do not tweek any parameter whatsoever.

4.2.1 RQ2: Are the features identified in state-of-the-art approaches targeting permissionless blockchains such as Ethereum applicable in Bitcoin or not?

Bitcoin has a different architecture than Ethereum. Due to the change-address concept, temporal features proposed in [2], such as attractiveness and bursty behavior of in-degree and out-degree, are spread over multiple addresses. Applying ML techniques on the temporal features as identified in [2] in Bitcoin directly does not provide any reliable results.

From Figure 2, assume that addresses $A_1, A_2, A_3, \cdots, A_n$ (denoted as a red oval) belong to the same user A. Let these addresses perform x number of transactions, $T_1, T_2, \cdots, T_x$, with addresses $\{B_1, B_2, \cdots, B_x\}$ (denoted as a black oval). Here, $\{A_3, A_5, \cdots, A_n\}$ are the change addresses of A (denoted as a dotted oval). The out-degree of A is spread over user's change addresses $A_3, A_5, \cdots, A_n$. Clustering such addresses give a temporal notion to the graph properties. If we do not cluster addresses, each address would be treated as different and temporal properties would not be correctly captured. Thus, the features identified in state-of-the-art approaches targeting permissionless blockchains such as Ethereum are not directly applicable in Bitcoin. In Bitcoin data, there exists few self-change addresses also (cf. Figure 1).

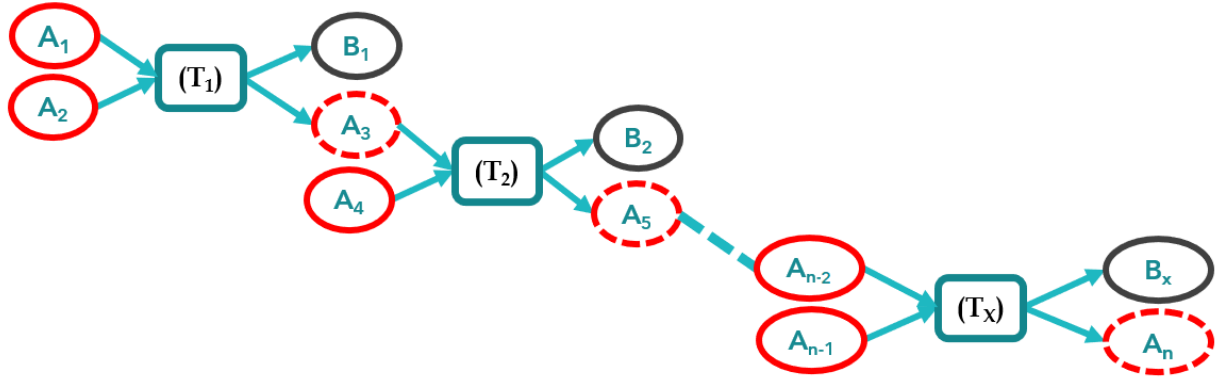[2]https://Bitcoin.org/en/full-node
[3]https://developer.Bitcoin.org/reference/rpc/
[4]https://www.blockchain.com/
[5]https://www.bitcoinabuse.com/
[6]https://cryptoscamdb.org/

Figure 2: Example of user's out-degree being spread over his change addresses.

Figure 3: Number of users after address clustering.

(a) 15Days.   (b) 15Days.   (c) 15Days.

(d) 1Month.   (e) 1Month.   (f) 1Month.
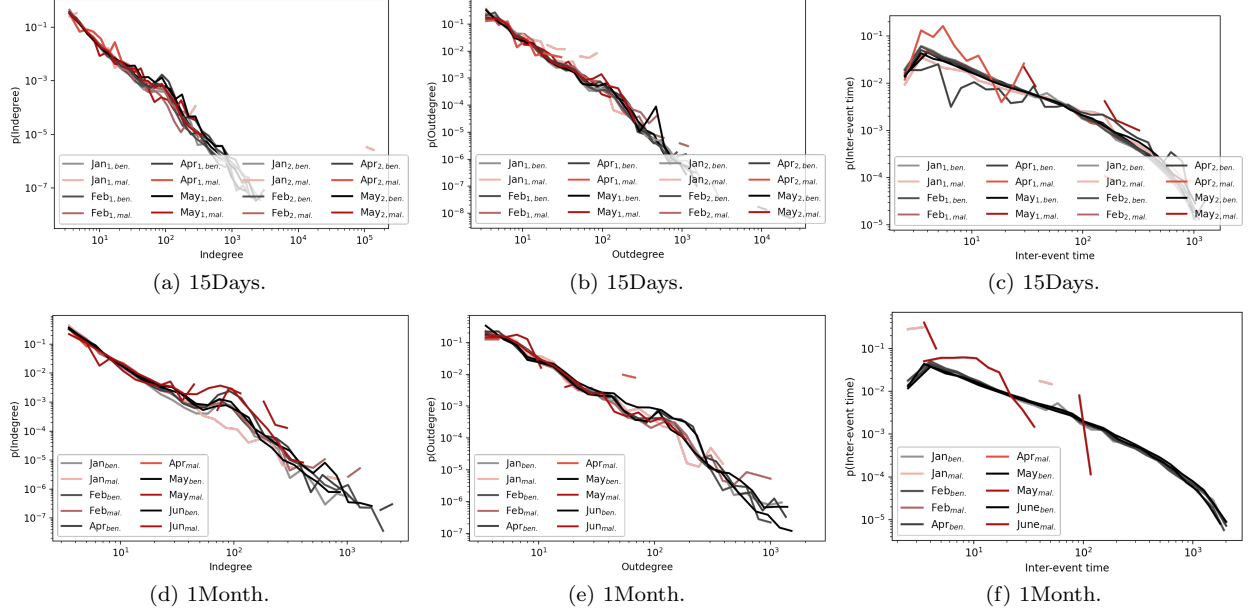
Figure 4: Distributions, (a, d) in-degree, (b, e) out-degree, (c, f) inter-event time. Here, $_{1,ben.}$ represents benign addresses of first 15Days of a month, $_{2,ben.}$ represents benign addresses of second 15Days. Similarly $_{1,mal.}$ represents malicious addresses of first 15Days of a month, $_{2,mal.}$ represents malicious addresses of second 15Days. For (c, f) the x-axis is time between 2 transactions in units=blocks. While for (a, b, d, e) x-axis unit is number.

We first validate this using different SDs in $SD^1(T_g)$ as done in [2]. Here, we apply the K-Means algorithm with $K=10$ on the entire considered data. As a result, we get millions of suspicious addresses. However, tagging millions of the addresses as malicious is similar to tagging entire Bitcoin as malicious, which is impractical.

We now perform address clustering using multi-input and change address heuristics. Figure 3 shows the number of unique users identified that performed a transaction every month. After address clustering, the total number of addresses that remain vary between 0.96% and 1.75% of the total number of addresses. Nonetheless, for June, the heuristics method cluster all the addresses as a single user. We do not know the exact cause for such a result. However, as one of the future investigations we would like to investigate the context and the reason behind such a result.

To understand whether the temporal features (such as burst and attractiveness) are applicable in Bitcoin or not, we generate distribution for in-degree and out-degree for both malicious and benign addresses. In [2], the authors identified that *power-law* distribution fits in-degree and out-degree in Ethereum. Here, from Figure 4, we find that power-law distribution also fits the in-degree and the out-degree for malicious and benign addresses present in Bitcoin. We find that, on average, power-law distribution with $x_{min}=2.3$, $\alpha \in \{2.04, 2.06\}$ for malicious addresses and $\alpha \in \{2.0, 1.9\}$ for benign addresses fits the in-degree in 15Days and 1Month temporal granularities, respectively (cf. Figure 4(a, d)). Here α and x_{min} are the power-law exponents and the minimum x from where the power-law distribution is observed. Further, on average, power-law distribution with $x_{min}=2.3$, $\alpha \in \{1.88, 2.1.78\}$ for malicious addresses and $\alpha \in \{1.84, 1.89\}$ for benign addresses fits the out-degree for different temporal granularities, respectively (cf. Figure 4(c, e)). To understand the difference between the two distributions identified using Bitcoin and Ethereum transaction data, we compute the *KL-divergence* (KLD) between them. The KLD score between Ethereum and Bitcoin addresses for in-degree is {0.051, 0.174} for benign and malicious addresses, respectively. While the KLD for out-degree is {0.124, 0.059} for benign and malicious addresses. This means there is a minimal difference between the distributions of these features. Thus, showing that the addresses in two blockchains behaves similarly.

In 15Days granularities, on average, (cf. Figure 4(c, f)), *truncated power-law* with $x_{min}=2.3$, $\lambda=\{0.00135, 0.00164\}$, and $\alpha=\{1.0002, 1.00008\}$ fits the inter-event time for benign and malicious addresses, respectively. Note that the truncation occurs at $\beta=1/\lambda$. In 1Month temporal granularity (cf. Figure 4f), for benign addresses, the distribution of inter-event times fit truncated power-law with $\lambda=0.0008$, $\alpha=1.00002$, and $x_{min}=2.3$. However, for malicious addresses *positive log-normal* distribution fits the best with $x_{min}=2.3$, $\mu=5.088$ and $\sigma=1.737$. The existence of power-law distribution indicates a bursty behavior both in in-degree, out-degree, and inter-event times of addresses. Hence, the features identified in state-of-the-art approaches targeting Ethereum are also applicable in Bitcoin but after change address clustering.
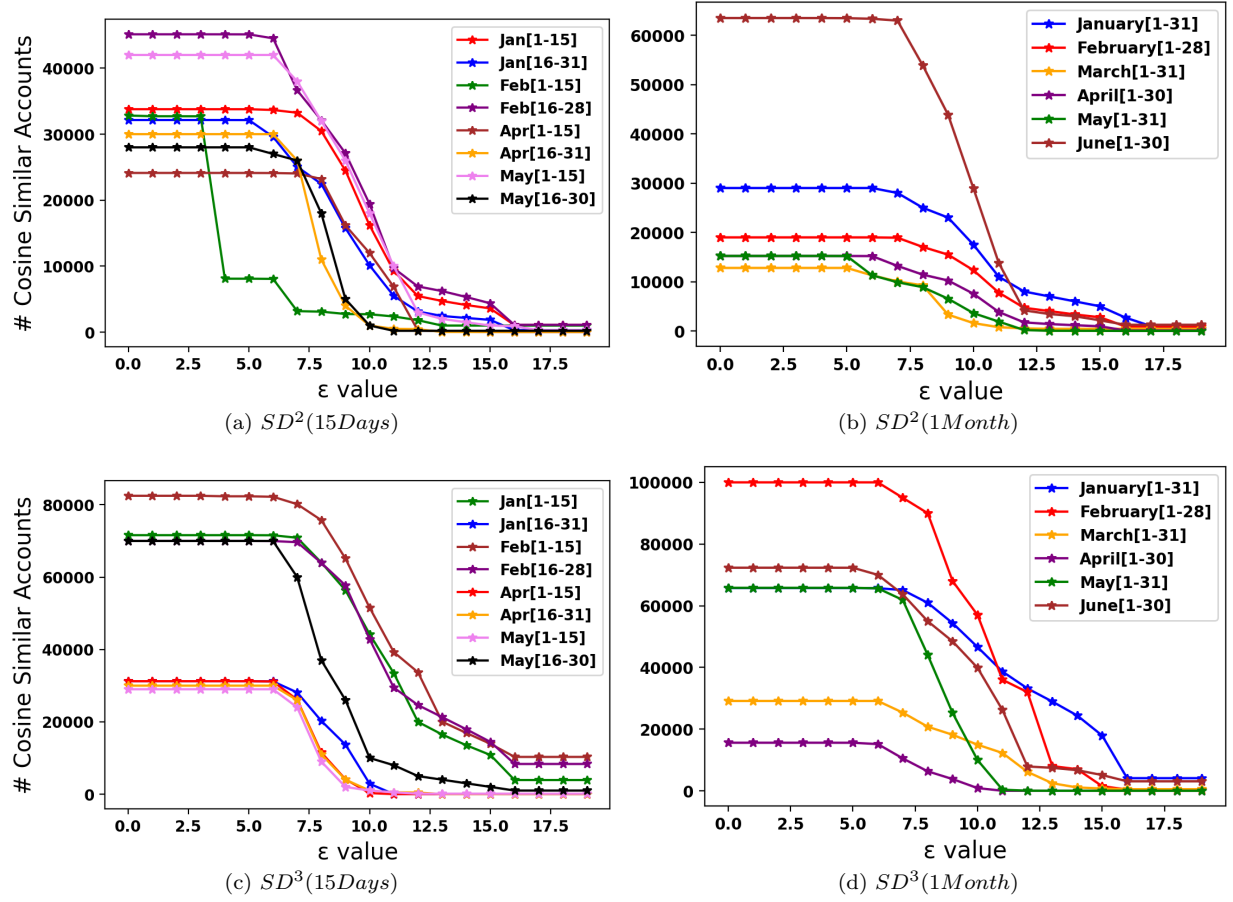
9

(a) $SD^2(15Days)$

(b) $SD^2(1Month)$

(c) $SD^3(15Days)$

(d) $SD^3(1Month)$

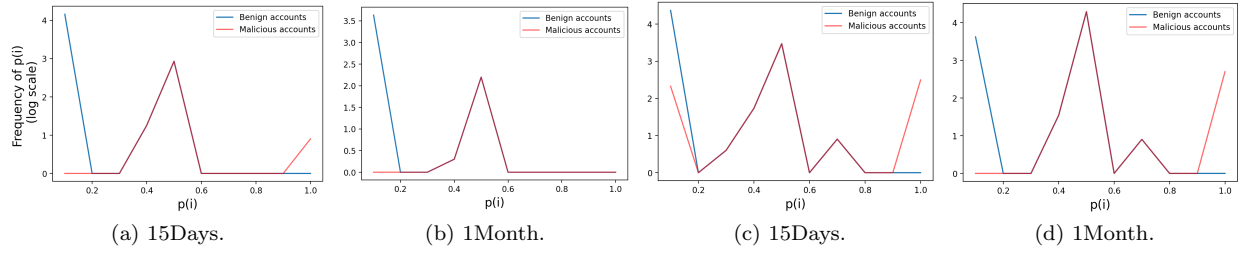Figure 5: Number of benign addresses having high CS with malicious addresses for different values of ϵ.

Figure 6: Distribution of p^k for malicious and benign addresses.

4.2.2 RQ1: Can we detect malicious addresses in Bitcoin using ML techniques and consider the temporal evolution of the graph?

To answer this, we first apply K-Means with K=10 on all SDs in $SD^2(15Days)$ and $SD^2(1Month)$. On the clusters identified by the K-Means, we choose the cluster with maximum tagged malicious addresses and find the CS scores between malicious and benign addresses in that cluster. Figure 5(a, b) shows the number of benign addresses having high CS score obtained for different ϵ. For SDs in $SD^2(15Days)$, we observe that $\epsilon\in[7,16]$ provides acceptable results where all benign addresses are not marked as malicious but only a few highly similar addresses are marked. While for SDs in $SD^2(1Month)$, we observe that the $\epsilon\in[12, 16]$. Note that, in Figure 5(a, c) as we did not had any significant number of malicious addresses, we could not compute the cluster with maximum number of malicious addresses for March for 15Days temporal granularity. While for June, all addresses clustered into one cluster.

Next, we extract all the temporal features introduced in [2] except the Ethereum blockchain specific features such as *max gas price* and *gas price burst*. After applying K-Means on all the SDs in $SD^3(15Days)$ and $SD^3(1Month)$, we perform similar steps as before.

Figure 5(c, d) shows the number of high similarity addresses obtained for different ϵ for different sub-datasets when using both temporal and non-temporal features. For $SD^3(15Days)$, we observe that $\epsilon\in[10, 16]$ provides acceptable results where all benign addresses are not marked as malicious but only a few highly similar addresses are marked. While for $SD^3(1Month)$, we observe that the $\epsilon\in[11, 16]$.

4.2.3 RQ3: What changes occur in the result after the change address clustering?

From the analysis in RQ1, we find 87,907 and 68,215 malicious addresses in $SD^3(15Days)$ and $SD^3(1Month)$. While we detect 43,366 and 16,575 malicious addresses in $SD^2(15Days)$ and $SD^2(1Month)$. Here, we can see that there is significant increase in the number of suspect addresses after adding the temporal features. We also observe that 5,080 suspect addresses are common in $SD^2(15Days)$ and $SD^3(15Days)$. While 10,292 suspect addresses are common in $SD^2(1Month)$ and $SD^3(1Month)$.

We find that the detected malicious addresses in the period of 6 months performed 1,77,907 transactions. These transactions worth 25,824.326 BTC which is 0.01% of total worth of benign transactions in that period. Thus, we verify that temporal features provide effective results towards detecting malicious addresses and identify changes that occur when we add temporal features.

4.2.4 RQ4: Does behavior change exists in Bitcoin addresses?

Here, we study the distribution of the probability of an address k to be malicious (p^k) using SDs in $SD^2(15Days)$, $SD^2(1Month)$, $SD^3(15Days)$, and $SD^3(1Month)$.

Figure 6(a, b) shows the frequency of p^k for benign addresses (blue line) and the frequency of p^k for addresses detected as malicious (red line), using SDs in $SD^2(15Days)$ and $SD^2(1Month)$. As expected, for most benign addresses, we observe that their behavior does not change. While for few addresses there is a very high probability of being malicious. We also observe that some addresses change their behavior and their $p^k\in[0.3,0.6]$. We find that 868 addresses in $SD^2(15Days)$ change their behavior between malicious and benign. While 159 addresses in $SD^2(1Month)$ change their behavior between malicious and benign. Further, we find that 8 addresses using 15Days temporal granularity and 0 addresses using 1Month temporal granularity and have p^k=1.

Similar to the previous case, Figure 6(c, d) shows the frequency of p^k for benign address (blue line) and the frequency of p^k for addresses detected as malicious (red line), using SDs in $SD^3(15Days)$ and $SD^3(1Month)$. In both the figures, we see a hike in the frequency of p^k for both malicious and benign address when $p^k\geq0.3$. Further, as expected for the benign addresses, most of the addresses do not change their behavior, i.e., p^k=0. While, for some addresses p^k=1, which means that these addresses are highly suspicious. We find that 3,273 addresses in

$SD^3(15Days)$ change their behavior between malicious and benign. While 19,712 addresses in $SD^3(1Month)$ change their behavior between malicious and benign. Further, we find that 313 addresses using 15Days temporal granularity and 501 addresses using 1Month temporal granularity and have $p^k=1$. We find that 3 addresses are detected by both 15Days and 1Month temporal granularity. We do not reveal the identity so as to not publicize these addresses as suspects.

Thus, we validate that behavior changes also exist in Bitcoin. More generally, the claim of [2] about the applicability of temporal features in any permissionless blockchain is valid in Bitcoin.

5   Conclusions and Future Work

With an increase in the adoption of Bitcoin, security threats have become inevitable, more diverse, and are deployed using much advanced techniques. Currently, there is a limited work that focuses on detecting the addresses involved in malicious activities in Bitcoin, and those available do not consider the temporal aspects of transactions in Bitcoin. However, temporal aspects are well studied in other blockchain such as Ethereum. In this work, we consider the temporal nature of Bitcoin transactions to detect malicious addresses, study their behavior, and validate the state-of-the-art methods for Bitcoin.

We observe that change-address heuristics plays an important role in extraction of temporal features, the state-of-the-art temporal features have similar behavior in Bitcoin as well, and temporal features are effective in terms of detecting more suspects. We find, with high probability, 3 suspect addresses that were detected across different temporal granularities. In the future, we would like to test the state-of-the-art method with more data to provide more robust results. Bitcoin is an extremely large connected graph with more than 651 Million transactions over 11 years. Not utilizing complete data might induce bias in the study and might not capture the true behavior. In our case, the application of heuristics, the results might have a bias probably which we see in month of June. If the heuristics algorithm provides the best results, our results would be best. However, we can not guarantee the effectiveness of the heuristics approach as we lack the ground truth about the addresses held by an entity. We would like to study the effectiveness of the heuristics algorithm in future.

Acknowledgements

References

[1] Number of daily Bitcoin transactions worldwide from January 2017 to April 13, 2021, 2021. Accesses 22/06/2021 online: https://www.statista.com/statistics/730806/daily-number-of-bitcoin-transactions.

[2] R. Agarwal, S. Barve, and S. Shukla. Detecting malicious accounts in permissionless blockchains using temporal graph properties. *Applied Network Science*, 6(9):1–30, 02 2021.

[3] R. Agarwal, T. Thapliyal, and S. Shukla. Detecting malicious accounts showing adversarial behavior in permissionless blockchains, 01 2021.

[4] C. Akcora, Y. Li, Y. Gel, and M. Kantarcioglu. Bitcoinheist: Topological data analysis for ransomware prediction on the bitcoin blockchain. In C. Bessiere, editor, *29th International Joint Conference on Artificial Intelligence*, pages 4439–4445. International Joint Conferences on Artificial Intelligence Organization, 7 2020.

[5] M. Bartoletti, B. Pes, and S. Serusi. Data Mining for Detecting Bitcoin Ponzi Schemes. In *Crypto Valley Conference on Blockchain Technology*, pages 75–84, Zug, 06 2018.

[6] Blockchain.com. The total number of transactions on the blockchain. Accesses 22/06/2021, online: https://www.blockchain.com/charts/n-transactions-total.

[7] Chainalysis. The 2021 Crypto Crime Report: Everything you need to know about ransomware, darknet markets, and more, 2021. (Accessed 30/06/2021).

[8] W. Chen, Z. Zheng, J. Cui, E. Ngai, P. Zheng, and Y. Zhou. Detecting Ponzi Schemes on Ethereum: Towards Healthier Blockchain Technology. In *World Wide Web Conference*, pages 1409–1418, Lyon, 04 2018.

[9] M. Christ, N. Braun, J. Neuffer, and A. Kempa-Liehr. Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package). *Neurocomputing*, 307:72–77, 09 2018.

[10] Sesha Kethineni, Ying Cao, and Cassandra Dodge. Use of bitcoin in darknet markets: Examining facilitative factors on bitcoin-related crimes. *American Journal of Criminal Justice*, 43:141—-157, 05 2017.

[11] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. Voelker, and S. Savage. A fistful of bitcoins: Characterizing payments among men with no names. In *Conference on Internet Measurement Conference*, pages 127–140, Barcelona, Spain, 10 2013. ACM.

[12] P. Monamo, V. Marivate, and B. Twala. Unsupervised Learning for Robust Bitcoin Fraud Detection. In *Information Security for South Africa (ISSA)*, pages 129–134, Johannesburg, 08 2016. IEEE.

[13] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Bitcoin, 2008. (Accessed 08/01/2020).

[14] T. Neudecker and H. Hartenstein. Could network information facilitate address clustering in bitcoin? In *Financial Cryptography and Data Security*, pages 155–169, Malta, 04 2017. Springer.

[15] T. Pham and S. Lee. Anomaly detection in the bitcoin system a network perspective. *arXiv*, pages 1–8, 11 2017.

[16] F. Reid and M. Harrigan. An analysis of anonymity in the bitcoin system. In Y. Altshuler, Y. Elovici, A. Cremers, N. Aharony, and A. Pentland, editors, *Security and Privacy in Social Networks*, pages 197–223. Springer, New York, NY, 07 2013.

[17] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 06 1998.

[18] H. Yin, K. Langenheldt, M. Harlev, R. Mukkamala, and R. Vatrapu. Regulating cryptocurrencies: A supervised machine learning approach to de-anonymizing the bitcoin blockchain. *Journal of Management Information Systems*, 36(1):37–73, 03 2019.

[19] Yuhang Zhang, Jun Wang, and Jie Luo. Heuristic-based address clustering in bitcoin. *IEEE Access*, 8:210582–210591, 11 2020.

[20] F. Zola, M. Eguimendia, J. Bruse, and R. Urrutia. Cascading Machine Learning to Attack Bitcoin Anonymity. In *2nd International Conference on Blockchain*, pages 10–17, Atlanta, 07 2019. IEEE.