# De-Anonymization of the Bitcoin Network Using Address Clustering

Changhoon Kang[+,1], Chaehyeon Lee[1], Kyungchan Ko[1], Jongsoo Woo[2], and James Won-Ki Hong[1]

[1] Dept. of Computer Science and Engineering, POSTECH, Pohang 37673, Korea
{chkang, chlee0211, kkc90, jwkhong}@postech.ac.kr
[2] Graduate School of Information Technology, POSTECH, Pohang 37673, Korea
woojs@postech.ac.kr

**Abstract.** Bitcoin is the first cryptocurrency that was invented by a pseudonymous person called Satoshi Nakamoto in 2008. Users of bitcoin are not required to provide any of their personal information, and this pseudo-anonymity attracts people to exploit bitcoin for illegal transactions. A previous study tried to de-anonymize the bitcoin by using P2P network traffic and find out an IP address of each bitcoin address' owner. However, this method could only obtain the small number of reliable mappings between a bitcoin address and its owner's IP address. To improve this study, we added a bitcoin address clustering process so that we could suppose that all addresses in each cluster are owned by one entity. We then mapped each cluster to an IP address and showed that this change increased the percentages of reliable mappings that we could find. We also suggested some ways to obtain improvements of our method.

**Keywords:** Bitcoin; Pseudo-Anonymity; De-Anonymization; IP address; Bitcoin address clustering

## 1 Introduction

Bitcoin [1] is a peer-to-peer network that records all transaction history in one distributed ledger. Without providing any personal information, any person can freely participate in the network by running a bitcoin node or simply generating a wallet with its bitcoin addresses. This feature provides pseudo-anonymity on the network so that no one knows an owner of each bitcoin address or a creator of each transaction. By taking advantage of this pseudo-anonymity, many people have used bitcoin as a means of payment in illegal trades. Silk Road, one of the well-known but now-defunct black market, had exploited bitcoin to deal with illegal transactions [2]. In addition, there was a study about the amount of bitcoin that used in illegal activities, and this study argues that roughly one-quarter of bitcoin users are related to illegal activities [3].

De-anonymization of the bitcoin network is to find out information hidden behind anonymity by analyzing disclosed data. Previous studies have already suggested various methods to do this. For instance, some studies drew a bitcoin

transaction graph to link bitcoin address to real entities [4, 5], and some of them also used external information of users to uncover anonymity of the bitcoin network [6]. Another study obtained mappings between a bitcoin address and an IP address that was likely to own it [7]. Our study was inspired by this work and tried to improve the efficiency of their analysis method.

Bitcoin address clustering is a process to gather bitcoin addresses that are likely to be owned by the same entity into one cluster. We added this process to the previous method. This process is important in a statistical analysis that counts the number of each bitcoin address' usage in transactions relayed from a specific IP address because most bitcoin addresses are used only once and this frequency is insufficient to use them in the statistical analysis. By using bitcoin address clustering, we can even count bitcoin addresses that were used only once in the past together with other addresses in the same cluster. In other words, in contrast to the previous method, our method relates each cluster of bitcoin addresses to a specific IP address and each cluster represents one entity.

In this paper, we introduce some backgrounds about the bitcoin system and several previous studies that were attempted to de-anonymize the bitcoin network in the following two sections. We then give a detailed explanation of our method and show that our method could find out about 80∼105 times as many reliable mappings as the method used in the previous study.

## 2 Background and Related Work

### 2.1 Bitcoin Network

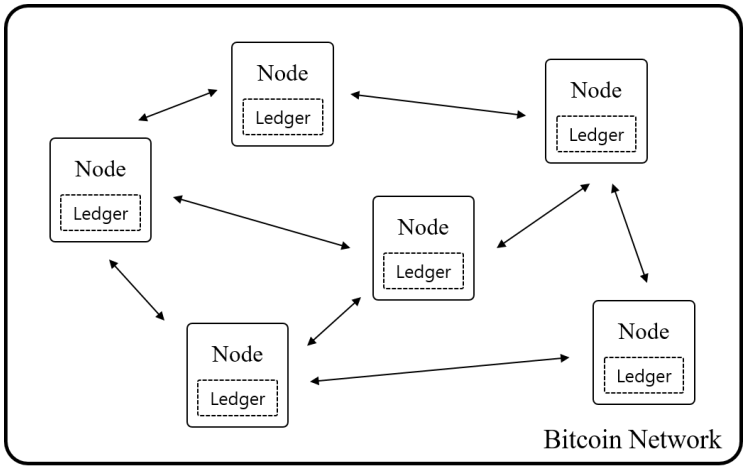


**Fig. 1.** A structure of the bitcoin network

The bitcoin network is a peer-to-peer network that consists of many nodes (Fig. 1). Each node can relay and generate blockchain-related data, including transactions and blocks. Once a node receives or generates these data, the node broadcasts them to the other nodes that are connected with this node. To operate the system, the network needs specific nodes called miners who validate received transactions and produce a new block with a certain volume of transactions. A distributed ledger called blockchain records all validated blocks so that every participant can access to the entire transaction history (Fig. 2).

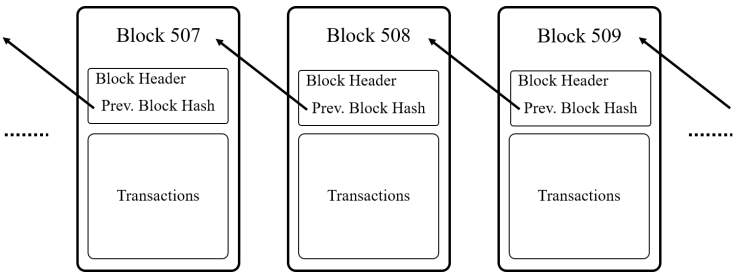


**Fig. 2.** A structure of a bitcoin blockchain

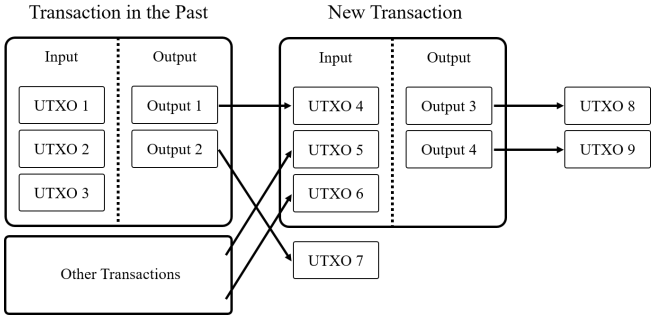


**Fig. 3.** A structure of a transaction and the use of UTXOs

Transactions of the bitcoin network are records of the flow of bitcoins. A structure of a transaction can simply be divided into input and output parts (Fig. 3). A creator of the transaction puts owned bitcoins that are outputs of an already confirmed transaction to the input and several sets of a receiver's bitcoin address and the number of bitcoins to send. If the transaction is validated and included in a new block, each of the output sets is called "Unspent Transaction Output (UTXO)" which is a unit of transaction input and output. Each receiver of new UTXOs can use them as the input of a new transaction that they create in the future.

### 2.2 Related Work

De-anonymization of the bitcoin network would be a common interest for both people who investigate illegal activities and who attack bitcoin users or system because it provides information that is useful for both actions. For that reason, many previous studies have worked on this in various ways of approaching.

Fleder, et al [4] explored the bitcoin system's anonymity level by drawing a transaction graph and linking bitcoin addresses to real people. They also used this graph analysis to compare the behaviors of known and unknown users. They eventually showed that the bitcoin network is not entirely anonymous.

Moser [5] also built a bitcoin transaction graph to analyze the anonymity of the bitcoin network. He chose three bitcoin mixing services that increase the anonymity of users and tracked the flow of bitcoins by analyzing a bitcoin transaction graph. He warned people that he could track the bitcoin that he used for the research in the analysis of one bitcoin mixing service that he worked on. From this study, we thought that this graph analyzing method also can be used for our study.

Reid, et al [6] created two network graphs derived from bitcoin's transaction history and analyzed their topological structure. They also showed the usage of external information and techniques to investigate an alleged theft of bitcoins.

Koshy, et al [7] suggested a method to de-anonymize the bitcoin network by using peer-to-peer network traffic data. They collected many other network-related data with bitcoin transactions, and they used an IP address data to find an IP address of each bitcoin address' owner. They used a statistical analysis on finding an owner IP of each bitcoin address, but this method wasted too much collected data because most bitcoin addresses are used only once and this frequency is not enough for the statistical analysis. To solve this problem, we propose our new method that clusters bitcoin addresses.

## 3 Bitcoin Address Clustering Method

In this section, we introduce the architecture of our method and explain each component in detail.

### 3.1 Architecture

The architecture of our method can be divided into data collection and statistical analysis (Fig. 4). In the data collection part, we collect incoming transaction message packets from a running bitcoin core client and store processed data that we need in the statistical analysis. After data collection, we analyze stored data and produce mappings between bitcoin addresses and IP addresses in the statistical analysis part. We also cluster bitcoin addresses and map each cluster on an IP address.

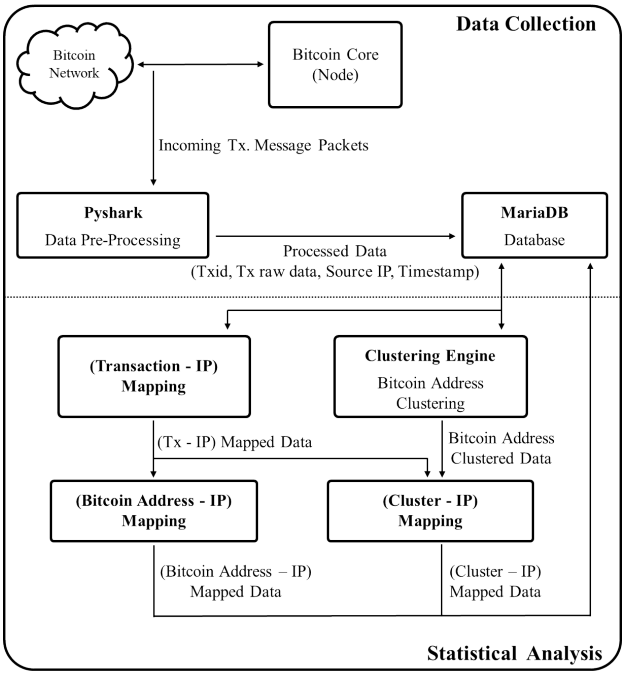


**Fig. 4.** The architecture of our method

### 3.2 Data Collection

**Construction of a Custom Bitcoin Client** A bitcoin client is a software that makes a computer act as one node in the bitcoin P2P network. This means that a computer that is running the bitcoin client sends and receives messages that are necessary to operate a bitcoin blockchain system. We seek to collect transaction messages with their senders' IP addresses.

To do this, we first downloaded the open-source code of the bitcoin core client (Version. 0.19.1) from its GitHub [8]. We modified some network configuration related constants in the code to increase a default limitation on the number of connections that the client can make with other nodes. This step was necessary because we had to find a creator of each transaction by analyzing senders' IP addresses, so connecting directly with as many nodes as possible would increase accuracy. Thus, we changed the limitation on the number of connections from 125 to 1125. Besides, there were some other parameters that we also had to change the default value to make more connections (Table 1). Connections of a bitcoin node can be divided into inbound and outbound connections, and they work exactly in the same way after once the connection is made. The second and third constants of the table are related to the number of outbound connections. We also increased these values, as we increased the number of total maximum peer connections. The last constant in the table is for configuring the length of

connection timeout. To keep being connected with other nodes, we extended this value.

**Table 1.** Default and changed values of modified constants related to network configuration

| Constant Name | Default | Changed |
|---|---|---|
| DEFAULT_MAX_PEER_CONNECTIONS | 125 | 1,125 |
| MAX_ADDNODE_CONNECTIONS | 8 | 1,008 |
| MAX_OUTBOUND_FULL_RELAY_CONNECTIONS | 8 | 1,008 |
| DEFAULT_PEER_CONNECT_TIMEOUT | 60 | 1,800 |

**Data Collection and Pre-processing** To collect transaction data with its sender's IP address, we used Pyshark [9] that is an open-source tool for network packet capture and analysis. We installed Pyshark to catch incoming transaction message packets. By using Pyshark, we could get both raw transaction data from packets and its sender's IP address. We saved collected data in a database that we built with an open-source database MariaDB [10]. For each transaction in the database, we collected transaction ID, the raw data of the transaction, the IP address of the sender, and the arrival time.

For each raw transaction data, we needed to extract bitcoin addresses that are used as a sender in a "vin" field and a receiver in a "vout" field. However, we could not obtain bitcoin addresses belonged to a sender's side directly from raw transaction data, while a "vout" field contains exact bitcoin addresses of receivers. In a "vin" field, there are only a transaction id and a "vout" index of each UTXO used for input of the transaction. To find bitcoin addresses that owned these UTXOs, we used a Blockchain Data API served by BLOCKCHAIN.COM [11]. We could get single transaction data with its transaction id by sending the GET request of the HTTP protocol. With this transaction data, we got bitcoin addresses from UTXOs in defined "vout" indexes. Finally, for each transaction, we stored the transaction id, the source IP address, the arrival timestamp, and (bitcoin address, value) sets of input and output.

### 3.3 Bitcoin Address Clustering

We clustered bitcoin addresses extracted from all collected transactions. Two different heuristics that we planned to use in our method are introduced below (Fig. 5).

**Multi-Input Addresses** We could assume that all bitcoin addresses that were used as input of one transaction belong to a single entity. This assumption is valid because a transaction creator has usually gathered his bitcoins that were owned by different addresses, and put them all together as input.

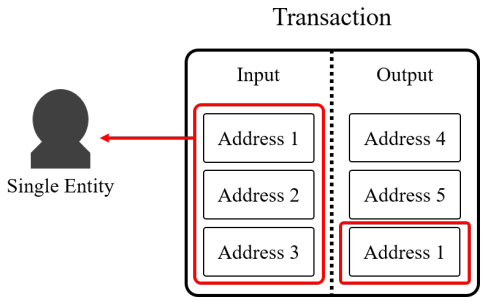


**Fig. 5.** According to explained two heuristics, address 1, 2, and 3 are owned by the same entity, and address 1 is a change address.

**A Change Address** We could also suppose that bitcoin addresses that were used both for input and output in one transaction are likely to be a change address of an entity who created the transaction. This assumption is valid because a transaction creator must spend all bitcoins in UTXOs used as input, he would return change to his bitcoin address again.

We described two different heuristics of bitcoin address clustering above. However, we only applied the "Multi-Input Addresses" heuristic in our method. This is because a change address is clustered as an input bitcoin address with the other addresses in the input.

### 3.4 Statistical Analysis

Except for applying bitcoin address clustering, we followed most processes in the previous method. To compare the improvement of the method, we conducted a statistical analysis twice with both the previous and our method.

**Mappings between Transactions and IP Addresses** We first had to infer an IP address of a user who was likely to create each transaction from its relayed pattern. We can divide relayed patterns into four different cases.

1. **Relayed by a Single IP**: In this case, only one IP address sent a transaction. We simply considered the IP address to be the transaction's creator.

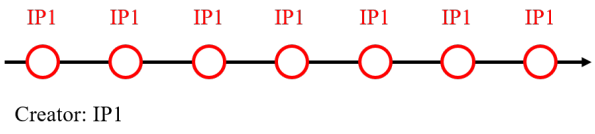


**Fig. 6.** Relayed by a single IP

2. **Relayed Once by Multiple IPs**: In this case, several IP addresses sent a transaction. We assumed that the sender of a transaction was the first one to receive it. This assumption may fail, because propagation delay may result in the transaction first reaching someone other than the real creator.

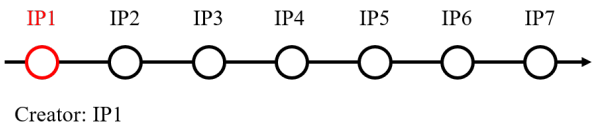


**Fig. 7.** Relayed once by multiple IPs

3. **Relayed Multiple times by a Single IP**: This case is the same as in the preceding case, except that one IP sends a transaction to our node multiple times. In this case, we selected the single IP that sent a transaction multiple times as the creator of the transactions. This assumption is reasonable because only a creator or bitcoin recipients of the transaction can relay the transaction multiple times and thus, the single IP address has the highest chance of being a creator of the transaction.

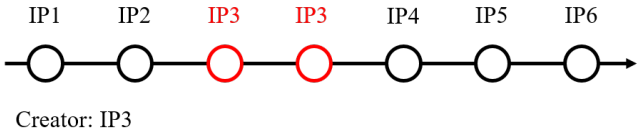


**Fig. 8.** Relayed multiple times by a single IP

4. **Relayed Multiple times by Multiple IPs**: In this case, multiple IPs send a transaction to us multiple times. There was no clear rule to choose one IP for the creator, and therefore, we did not handle this case in this study.

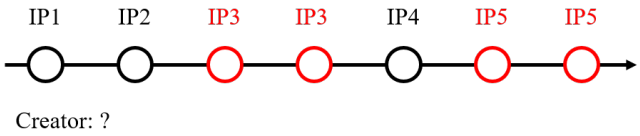


**Fig. 9.** Relayed multiple times by multiple IPs

**Mappings between Bitcoin Addresses and IP Addresses** The previous study conducted this analysis with their method, and to compare the efficiency of the method with ours, we reconstructed this analysis with data that we collected in this study. From the previous stage, we obtained mappings between transactions and IP addresses. For each transaction, we could extract bitcoin addresses from input and output separately, and relate them to an IP address that was mapped to the transaction. We calculated two probabilities for each bitcoin address $b$ and an IP address $i$ pairs, and all notations and explanations below are from the previous study [7].

1. $N_I(b, i)$: The number of different transactions that are owned by IP $i$ and contain bitcoin address $b$ in their input.
2. $N_O(b, i)$: The number of different transactions that are owned by IP $i$ and contain bitcoin address $b$ in their output.
3. $N_I(b)$: The number of different transactions that contain bitcoin address $b$ in their input.
4. $N_O(b)$: The number of different transactions that contain bitcoin address $b$ in their output.

$$P_I(b, i) = \frac{N_I(b, i)}{N_I(b)}, \qquad P_O(b, i) = \frac{N_O(b, i)}{N_O(b)}$$

$P_I(b, i)$ is a probability that a transaction containing bitcoin address $b$ in its input is owned by IP $i$. Similarly, $P_O(b, i)$ is a probability that a transaction containing bitcoin address $b$ in its output is owned by IP $i$.

**Mappings between Clusters and IP Addresses** This is a method that we suggest in this paper, using bitcoin address clustering with the previous method. Most processes are the same as the existing method, except for some differences in notations. In our method, we also calculated two probabilities but this was for each bitcoin address cluster $c$ and IP address $i$ pairs.

1. $N_I(c, i)$: The number of different transactions that are owned by IP $i$ and contain a bitcoin address belonged to cluster $c$ in their input.
2. $N_O(c, i)$: The number of different transactions that are owned by IP $i$ and contain a bitcoin address belonged to cluster $c$ in their output.
3. $N_I(c)$: The number of different transactions that contain a bitcoin address belonged to cluster $c$ in their input.
4. $N_O(c)$: The number of different transactions that contain a bitcoin address belonged to cluster $c$ in their output.

$$P_I(c, i) = \frac{N_I(c, i)}{N_I(c)}, \qquad P_O(c, i) = \frac{N_O(c, i)}{N_O(c)}$$

$P_I(c, i)$ is a probability that a transaction containing a bitcoin address belonged to cluster $c$ in its input is owned by IP $i$. Similarly, $P_O(c, i)$ is a probability that a transaction containing a bitcoin address belonged to cluster $c$ in its output is owned by IP $i$. We can apply these probabilities of cluster $c$ to each bitcoin address contained in the cluster.

**Extracting Reliable Mappings** Finally, we extracted reliable mappings that satisfies the configured thresholds. We only picked pairs that have 50% or higher probability for $P_I(b, i)$ or $P_I(c, i)$. We also removed pairs if their $N_I(b, i)$ or $N_I(c, i)$ value is lower than 10.

## 4 Result

### 4.1 Data Collection

We started our bitcoin client on Mar.10, 2020, and collected data until May.09, 2020. Although we set the maximum number of connections to 1,125, the client software automatically reduced this number to 1,124 because of a lack of computing resources. After synchronizing with all previous blocks, the client started to send and receive bitcoin protocol message packets. We observed that during data collection, the client kept maintaining more than 1,025 connections. To increase accuracy, ideally, we had to connect with almost all bitcoin nodes, but we could not make more than 1,120 connections.

We collected incoming transaction data, pre-processed them, and then stored them in a database that we built. We could collect and save a total of 2,347,420 transactions. In the data collection stage, we had already filtered invalid transactions that have a wrong structure, but we again had to filter some of them that have invalid UTXOs as input. Eventually, we got a total of 2,081,891 valid transactions.

### 4.2 Bitcoin Address Clustering

We clustered bitcoin addresses in all collected valid transactions by using the simple heuristic explained in section 3.3. From 2,081,891 transactions, we could extract 5,445,185 bitcoin addresses. After clustering, we got 4,474,624 clusters that represent each user entity. Because each cluster consists of some bitcoin addresses, we could use a large number of bitcoin addresses even if they were only used once.

### 4.3 Statistical Analysis

**Mappings between Transactions and IP Addresses** We mapped each transaction to an IP address that was likely to have created; for this purpose, we used analysis on a relaying pattern of the transaction. By using four relaying patterns that we had set earlier, we could map all transactions to such IP addresses. Eventually, we got 2,078,243 (Transaction - IP Address) mappings from 2,081,891 valid transactions. Overlap of transactions in our dataset reduced the total number of transactions.

**Extracted Reliable Mappings from Each Method** We obtained mappings of (Bitcoin Address - IP Address) and (Cluster - IP Address) from each method. For each case, we counted the number of mappings that have reliability above each level (Table 2). As a result, we found that a method that clusters bitcoin addresses produced reliable mappings more than a method that maps bitcoin addresses separately onto IP addresses.

**Table 2.** The number of bitcoin addresses that could find their owner's IP address for each reliability level. ($N_I(b,i)$ or $N_I(c,i) \geq 10$)

| **Reliability Threshold** ($P_I(b,i)$ **or** $P_I(c,i)$) | **Mapping Object** | |
|---|---|---|
| | **Bitcoin Address** | **Addresses' Cluster** |
| ≥50% | 1,246 | 131,599 |
| ≥60% | 1,102 | 112,144 |
| ≥70% | 1,008 | 93,803 |
| ≥80% | 925 | 77,227 |
| ≥90% | 846 | 69,116 |
| ≥95% | 766 | 61,901 |
| ≥99% | 735 | 58,587 |

## 5 Discussion

We collected bitcoin transaction data with their senders' IP addresses, then clustered bitcoin addresses in transactions, and used statistical analysis to find reliable mappings between bitcoin addresses and IP addresses. We performed this study to improve the efficiency at which IP addresses of bitcoin addresses' owners can be found. The previous study that did not cluster bitcoin addresses could exploit only a few bitcoin addresses for statistical analysis because most of them were used only once in the whole record and this frequency was too low to allow reliable statistical analysis. Thus, we clustered bitcoin addresses and because of this, addresses in each cluster could be counted together as the same entity.

We found that adding the clustering of bitcoin addresses increased the number of reliable mappings between bitcoin addresses and IP addresses that were likely to own the bitcoin addresses. Our method with clustering found about 80∼105 times as many mappings as the method used in the previous study. This result means that clustering reduced the number of bitcoin addresses that are unusable in the statistical analysis. However, in the clustering process, we only used a simple heuristic, and this method still left too many different small clusters. We expect that an improved clustering method might increase the number of reliable mappings.

The percentages of reliable mappings are still too low even though our method improved on the previous method. One possible reason for the low mapping rate

is the limited number of connections that we could make with other bitcoin nodes. We only connected with some parts of bitcoin nodes and received transaction data from them, so the inferred IP addresses of the creators would be wrong for most of the collected transaction data. To overcome this problem, we must find a way to connect with most of the bitcoin nodes in the world. However, this study shows that there is a chance to improve the efficiency of finding an IP address of each bitcoin address' owner. New attempts with some improvements in the analysis process can be done in the following research.

## 6 Conclusion

We added a bitcoin address clustering process to the previous method that deanonymized the bitcoin network by obtaining mappings between a bitcoin address and an IP address that was likely to own it. Unlike the previous study, we first made clusters of bitcoin addresses so that we could suppose all addresses in each cluster are owned by one entity. This change achieved a remarkable increase in the number of reliable mappings that we could find. Our method consists of several distinct analysis processes and thus, we have more chances to increase efficiency by improving each analysis process.

## Acknowledgments

This work was supported by the ICT R&D program of MSIT/IITP [No.2018-000539, Development of Blockchain Transaction Monitoring and Analysis Technology] in Republic of Korea.

## References

1. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.
2. Alice Huang. Reaching within silk road: the need for a new subpoena power that targets illegal bitcoin transactions. *BCL Rev.*, 56:2093, 2015.
3. Sean Foley, Jonathan R Karlsen, and Tālis J Putniņš. Sex, drugs, and bitcoin: How much illegal activity is financed through cryptocurrencies? *The Review of Financial Studies*, 32(5):1798–1853, 2019.
4. Michael Fleder, Michael S Kester, and Sudeep Pillai. Bitcoin transaction graph analysis. *arXiv preprint arXiv:1502.01657*, 2015.
5. Malte Moser. Anonymity of bitcoin transactions. 2013.
6. Fergal Reid and Martin Harrigan. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*, pages 197–223. Springer, 2013.
7. Philip Koshy, Diana Koshy, and Patrick McDaniel. An analysis of anonymity in bitcoin using p2p network traffic. In *International Conference on Financial Cryptography and Data Security*, pages 469–485. Springer, 2014.
8. Bitcoin core client. Available at https://github.com/bitcoin/bitcoin/.
9. Pyshark. Available at https://github.com/KimiNewt/pyshark/.
10. Mariadb. Available at https://mariadb.org/.
11. BLOCKCHAIN.COM. Blockchain data apis. Available at https://www.blockchain.com/en/api/blockchain_api/.