**QUEEN'S
UNIVERSITY
BELFAST**

Can SDN deanonymize Bitcoin users?

Wallace, V., & Scott-Hayward, S. (2020). Can SDN deanonymize Bitcoin users? In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)* Institute of Electrical and Electronics Engineers Inc.. <https://doi.org/10.1109/ICC40277.2020.9148936>

Published in:

ICC 2020 - 2020 IEEE International Conference on Communications (ICC)

Document Version:

Peer reviewed version

Queen's University Belfast - Research Portal:

[Link to publication record in Queen's University Belfast Research Portal](#)

Publisher rights

© 2020 IEEE. This work is made available online in accordance with the publisher's policies. Please refer to any applicable terms of use of the publisher.

General rights

Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

Can SDN deanonymize Bitcoin users?

Victoria Wallace

*Centre for Secure Information Technologies
Queen's University Belfast
Belfast, BT3 9DT, N. Ireland*

Sandra Scott-Hayward

*Centre for Secure Information Technologies
Queen's University Belfast
Belfast, BT3 9DT, N. Ireland
s.scott-hayward@qub.ac.uk*

Abstract—According to the Internet Organised Crime Threat Assessment (IOCTA) 2019 report, Bitcoin is still the currency of choice in criminal markets and as payment for cyber-related extortion attempts, such as from ransomware or a Distributed Denial-of-Service (DDoS) attack. Bitcoin is a peer-to-peer electronic cash system first proposed by Satoshi Nakamoto in 2008. By design, Bitcoin is a pseudonymous coin, meaning that users can transact with the currency without revealing their true identity. To tackle the challenge of Bitcoin-related crime, a range of deanonymization techniques have been proposed. In general, these solutions are limited by the time and resources required to predict likely transaction owners. In this paper, we propose the first software-defined network (SDN)-based Bitcoin transaction mapping solution. We analyse the Bitcoin transaction process in an SDN environment and demonstrate a deterministic approach to deanonymize users in Bitcoin's network.

Index Terms—Network Forensics, Software-Defined Networks, Bitcoin, Transaction Monitoring, Deanonymization techniques.

I. INTRODUCTION

IN 2008, a paper [1] published by the author pseudonymously known as Satoshi Nakamoto marked the birth of today's most successful cryptocurrency, Bitcoin. First deployed in 2009, Bitcoin is a decentralized virtual currency, which allows its users to join the network and transact freely without the approval of a central authority. Instead, the issuance of bitcoins (BTC)¹ relies on a consensus algorithm performed by a community of Bitcoin users known as miners [2]. In Bitcoin, each node verifies transactions, which are then forwarded to other nodes, assembled in a block, and once the block is mined, the transactions are added to the Blockchain, a distributed, public ledger introduced by Nakamoto to facilitate the exchange of BTCs.

Bitcoin has exploded in value and demand since its first few years in production. On the 1st January 2011, 30 cents would buy you 1 BTC. By December 2017, this investment would yield \$19,783 [3]. There are some favourable characteristics of Bitcoin that can be attributed to its rapid growth; the permissionless nature of the Blockchain allows for fast, frictionless trade, and BTC provides its users with pseudonymity, which some experts believe is the main driver behind its growth [4]. Users on the Blockchain are identified by a pseudonym, which, at first glance, isn't explicitly tied to a real-world identity. For this reason, BTC is typically perceived as an anonymous currency, providing a convenient platform for those who wish to conceal their identity. As

such, BTC has been earmarked by cybercriminals to facilitate fraud, ransomware payments, money laundering, and other illicit activity [5]. Law enforcement officials are challenged with convicting these criminals due to the difficulty in tracing suspicious transactions back to their source.

Software-defined networking (SDN) is beginning to gain traction in the networking community. Google, Microsoft, Yahoo and Facebook are a few early adopters of SDN [6]. SDN is transforming the static architecture of traditional networks by introducing network-wide visibility at a logically centralized controller, dynamic flow control, programmability and easy implementation of network-wide policies [7]. In particular, a holistic view of the network state and centralized control of network behaviour provides the opportunity to easily monitor and control the flow of all traffic moving between devices in the network. For this reason, SDN has sparked the interest of network security researchers, who have proposed SDN-based Firewalls and Intrusion Detection Systems [8]. Several Blockchain applications [9]–[11] have also been proposed.

While research carried out to date provides evidence that the perceived anonymity of Bitcoin is under threat [12]–[19], a deterministic solution to deanonymize Bitcoin users is yet to materialise. The aim of this research is to investigate how SDN can be leveraged to assist law enforcement in tackling the challenge of tracking down the perpetrators of Bitcoin related crime. To this end, we characterize the Bitcoin transaction process in an SDN environment and investigate the extent of data capture at the controller to monitor and forensically analyse the flow of Bitcoin transactions. We make the following contributions:

- A first investigation into the potential of SDN to capture and analyse IP traffic to draw links between Bitcoin transactions and users.
- A Bitcoin transaction monitoring application (BTMA) using SDN.
- A deterministic approach to deanonymize Bitcoin by mapping transaction IDs.

The remainder of this paper is organised as follows. Section II provides a review of relevant literature. Section III introduces the elements of Bitcoin fundamental to this study. In Section IV, the methodology is described based on a set of investigation hypotheses. Section V provides an insight to the performance of this system with discussion in Section VI. Section VII concludes the article.

¹Throughout the paper, we refer to Bitcoin for the payment system and P2P network, and bitcoin (BTC) for the unit of currency.

II. RELATED WORK

A. Bitcoin Deanonimization: Blockchain Analysis

To date, researchers investigating the extent of anonymity in Bitcoin have generally focused their work on two methods; blockchain analysis and peer-to-peer (P2P) network analysis. Reid and Harrigan [12] were the first to propose a deanonymization technique using the public availability of the Blockchain. With transaction information from the Blockchain, the authors constructed a transaction and network graph, which they used to create transaction flows by following the path of BTC between users over time. The authors were also the first to introduce a clustering heuristic to group commonly owned Bitcoin addresses based on the universally accepted idea that transactions with multiple input addresses are typically controlled by the same user. Using off-network information, they attempted to link these address clusters to a real entity. They demonstrated the effectiveness of this method in deanonymizing high-profile Bitcoin users.

Sarah Meiklejohn et al. [13] introduced a second heuristic, which takes into account the role of change addresses. The authors assumed that for transactions with multiple inputs and at least two outputs, the output address with only one input must be the change address. Using this approach, the authors were able to collapse $\approx 12\text{M}$ unique addresses to $\approx 3.5\text{M}$ address clusters. Androulaki et al. [14] presented a method to cluster addresses based on the behaviour they exhibited. The authors presented an algorithm, which took into account the timing of transactions, the amount in BTCs spent, and the indexes of different addresses that appear within transactions to predict address clusters controlled by the same user. In [15], a study of the Blockchain specific to its misuse over the Dark Web is presented. The authors used their MFSScope tool to collect and analyse cryptocurrency addresses associated with dark websites. They came up with 85 addresses very likely to be associated with illicit activity and managed to link them to a further 3,029 addresses by utilizing the two address clustering heuristics. With an investment scam example, the authors demonstrated how their solution could be used to facilitate law enforcement in tracking down the perpetrators of crypto-related crime.

As address clustering and tracing methods emerged, cryptocurrency users responded with efforts to enhance anonymity, such as through the use and development of mixing services [20], which are designed to obfuscate the relationship between transactions and addresses, as seen from the Blockchain.

B. Bitcoin Deanonimization: P2P Network Analysis

Recent research attempts to circumvent the impact of mixing services and address the limitations of address clustering methods. In general, these methods attempt to deanonymize users by mapping Bitcoin addresses to their originating IP address through network traffic analysis. Koshy et al. [16] observed and characterized the relay pattern of transactions across the network to deduce relationships between IP addresses and Bitcoin pseudonyms. Over a 5-month period, their

Bitcoin client, CoinSeer, collected transaction data from 2,678 peers. From the stored data, the authors classified distinct transaction relay patterns and designed heuristics for hypothesizing transaction ownership. They successfully generated Bitcoin-IP-address mappings. However, the majority of mappings derived from transactions that followed an anomalous relay pattern, which accounted for just 8% of the observed traffic. Furthermore, the authors were only able to connect to and collect data from server nodes, which comprise just 10% of Bitcoin's network.

Other methods exploit vulnerabilities in the P2P network's gossip protocol. Biryukov et al. [17] attempted to link unreachable nodes to their IP addresses by utilizing first relayer information. In order to identify unreachable peers (or clients), the authors used Bitcoin's address propagation mechanism to learn the addresses of clients connected to each server. Inferring that the first nodes to relay a transaction must be entry nodes of the originator, the authors mapped transactions to the clients with varying success (11-60% depending on their level of confidence). This method requires maintenance of hundreds of outgoing connections to listening peers on the network. In contrast to [17], Juhász et al. [18] require a direct incoming connection from each client node. Using a Bayesian mathematical model based on timings between message propagations, the authors assign a probability of transaction ownership to each address.

While these methods show potential to deanonymize bitcoin users, [16] and [17] first proposed their solutions when Bitcoin utilized a trickle spreading protocol. In 2015, the Bitcoin community responded to deanonymization attempts by changing the network's message propagation mechanism from trickle spreading to a diffusion spreading protocol. Fanti et al. [19] compare and analyse the anonymity properties of Bitcoin's networking stack both pre- and post- 2015, concluding that the diffusion spreading protocol offers greater anonymity guarantees than trickle spreading. As of today, the Bitcoin Core client ² utilizes a different relaying mechanism. Every client maintains a queue for the messages to be relayed for each connected client and relays them according to a poisson process [18]. No method proposed to date investigates the impact of this protocol on anonymity.

C. SDN and Blockchain

The need for SDN stemmed from the demands of the changing infrastructure of the Internet such as the rise of cloud computing, changing traffic patterns and the consumerisation of IT, which traditional networking cannot meet [21]. However, SDN is also being embraced by security researchers for its' potential in enhancing network security [8]. An overview of SDN monitoring solutions is provided in [22]. Recently, we have seen the emergence of SDN applications to secure the Blockchain. SecureChain [9] combined the properties of SDN and Ethereum's Blockchain to create a security gateway. Steichen et al. [10] proposed ChainGuard, an OpenFlow (OF)

²<https://bitcoin.org/en/bitcoin-core/>

based security application, which provides access control functionality to protect blockchain nodes against illegitimate traffic. Houda et al. [11] developed an SDN application, ChainSecure, designed to protect blockchain nodes from DNS amplification attacks. These solutions demonstrate that SDN can support network security for Blockchain nodes. In this research, we investigate the use of SDN to implement a Bitcoin deanonymization technique.

III. ANATOMY OF A BITCOIN TRANSACTION

A. Bitcoin Blockchain

Bitcoin is a virtual, decentralized cryptocurrency, which requires no central authority or trusted third-party to validate the issuance of coins [1]. Instead, public-key cryptography is used to enable users to send and receive BTC without going through a financial institution. Users participating in the network can generate any number of Bitcoin accounts which correspond to public-private key pairings. The hash of the public key serves as a unique identifier or pseudonym without disclosing personally identifiable information [2]. Users can then send and receive BTC at will by verifying ownership of transactions with their corresponding private key. To prevent the double-spending problem, transactions are recorded publicly on a distributed ledger known as the Blockchain [1]. The Blockchain is made up of a chain of blocks, which each consist of a group of confirmed transactions and a cryptographic hash of the previous block. Before a new block can be added to the Blockchain, it must contain a proof-of-work, which requires a secure cryptographic hash of the previous block to be generated in a process known as mining. A community of users known as miners race to compute a proof-of-work that works for the new block, before advertising it to the rest of the network for validation. This process is designed to be computationally expensive and extremely difficult to reverse in order to prevent untrustworthy peers from modifying the Blockchain. For this reason, the miner responsible for generating a new block is rewarded for their work with newly generated BTC, which become the first transaction in the new block known as a coinbase transaction.

The following subsections provide an overview of Bitcoin's P2P network, transactions, and transaction propagation.

B. P2P Network

Connections between nodes in Bitcoin's P2P network are established over an unencrypted TCP channel. To join the network and run a Bitcoin node, a user can download Bitcoin client software such as the Bitcoin Core client³. There are two main classes of nodes in the P2P network; reachable or server nodes, and unreachable or client nodes. Typically, nodes can maintain up to 8 outgoing connections to other nodes and a further 117 incoming connections if they run as a server [2].

Bitcoin utilizes a gossip protocol to enable nodes to communicate. For example, to establish an outgoing connection on the network and enable communication, a client and server

node must first exchange VERSION and VERACK messages. The protocol consists of a large number of such application-layer messages⁴. A description of the relevant messages for this research is given in Table I.

TABLE I
BITCOIN PROTOCOL MESSAGE TYPES

Message	Description
ping & pong	PING is sent to check the status of a connection is still alive. PONG is sent in response if it is.
addr & getaddr	The Bitcoin protocol implements an address propagation mechanism to enable nodes to discover others on the network. Each node maintains a list of IP addresses of other nodes on the network. Using a GETADDR message, a node (Alice) can request a list of IP addresses from another node (Bob). Bob responds with a list of IP addresses to Alice using the ADDR message.
inv	Allows a node to advertise its' knowledge of one or more objects. For example, a node will send an INV message to its' peers before advertising a block or transaction.
getdata	GETDATA is sent in response to an INV message to retrieve the full content of a specific object.
tx	TX describes a bitcoin transaction. When a node wants to propagate a transaction, the transaction ID is broadcast in the body of an INV message. Connected nodes will respond with a GETDATA message before the bitcoin transaction is sent in a TX message.

C. Transaction Propagation

Bitcoin's protocol implements a transaction propagation mechanism to facilitate the exchange of BTCs, as illustrated in Figure 1. When a client node, O , creates a transaction, she advertises it to her connected peers, $E[1..5]$. When these peers, or entry nodes, receive the transaction, they will proceed to advertise it to their connected peers. This process continues until the transaction has traversed the whole network. As a consequence of this mechanism, an arbitrarily chosen node is not necessarily directly informed about the transaction by the originator [17].

Fig. 1. Bitcoin transaction propagation

D. Anatomy of a transaction

Transactions, tx , are data structures that facilitate the movement of BTCs between nodes in the network. They are composed of at least one input and one output, each containing amount and Bitcoin address information. The input of a transaction must refer to an unspent output of a previous transaction

³<https://bitcoin.org/en/bitcoin-core/>

⁴https://en.bitcoin.it/wiki/Protocol_documentation

(UTXO) [2]. A list of UTXOs owned by a particular user are contained in the users’s wallet and make up the user’s balance. For example, as shown in Figure 2, suppose Alice has two UTXOs in her wallet, one worth 10 BTCs and the other worth 12 BTCs. Alice wants to send 15 BTCs to Bob. Bob generates a new address (public-private key pairing) and sends Alice his public key. Alice generates a transaction with 2 inputs and 2 outputs. The 2 inputs consist of the 2 UTXOs contained in Alice’s wallet along with corresponding signatures (private keys) to confirm ownership. One of the transaction outputs refers to the transaction amount and Bob’s public key, the other output refers to a change address generated by Alice. Since UTXOs are indivisible, specifying a change address in the output will ensure change is returned to the payer. This tends to be a little less than the amount owed because miners usually charge a small transaction fee. When Alice executes the transaction, a new transaction hash is generated. The transaction hash is coupled with an index for each output address associated with the transaction.

Fig. 2. Two UTXOs from Alice’s wallet form the input to the transaction. Once the transaction is executed, it generates a new transaction hash; the transaction at index 0 is sent to Bob’s address, the transaction at index 1 is returned to Alice.

The structure of the tx message is fundamental to the developed solution and is illustrated in Figure 3.

Field Size	Field name	Description/Example
4	version	Transaction data format version e.g. 02000000
0 or 2	Flag	If present, always 0001; indicates presence of witness data
1+	tx_in count	Number of transaction inputs (never zero) e.g. 02
41+	tx_in	A list of one or more transaction inputs (UTXOs)
1+	tx_out count	Number of transaction outputs
9+	tx_out	A list of 1 or more transaction outputs
0+	tx_witnesses	A list of witnesses, one for each input
4	lock_time	The block number or timestamp at which this transaction is unlocked

Fig. 3. Bitcoin protocol API structure of a tx message

IV. METHODOLOGY

In this section, we explain our technique for Bitcoin transaction analysis in SDN through controller network monitoring. By emulating the Bitcoin network in a SDN environment, we can create a Bitcoin transaction monitoring application (BTMA) capable of inspecting traffic sent between every node in the network without having to maintain hundreds of connections with every listening peer in the network.

We make the following observations based on the background and related work presented in Sections II and III: the first node to relay a transaction is the creator of that transaction, Bitcoin messages are sent over an unencrypted

TCP channel, tx messages contain as their input, information pertaining to the creator of the transaction including the hash and index of UTXOs, and, in SDN, the controller has a holistic view of the network and traffic sent between devices in the data-plane can be redirected to and monitored by the controller. Based on these observations, the following hypotheses were derived:

- 1) In an SDN network, the first IP address to relay a transaction message to the controller is the owner of that transaction.
- 2) The payload of unencrypted transaction packets can be extracted at the SDN controller to uncover the transaction ID, TX_{ID} , for any given transaction.
- 3) Given Hypotheses 1 and 2, transactions can be mapped to their originating IP address: $TX_{ID_i} \rightarrow IP_{src_i}$
- 4) Given Hypotheses 1 and 2, the Blockchain can be utilized to map transactions to their originating Bitcoin address: $TX_{ID_i} \rightarrow addr_{src_i}$

A. Private Bitcoin Network

In order to test these hypotheses, a test network was constructed using Mininet [23] with the Ryu SDN Controller⁵, as shown in Figure 4.

Fig. 4. BTMA Test Network with transaction monitoring

Within Mininet, Bitcoin Core’s regression testing mode (regtest) was used to build a private Bitcoin network. Bitcoin Core client is free, open-source software that serves as a Bitcoin node. Regtest closely mirrors the behaviour of the real network by using the same fundamental rules, but with one major difference; the user has complete control over the generation of new blocks. Bitcoin Core provides a remote procedure call (RPC) interface to perform various tasks such as wallet operations, administrative events and queries about the network and blockchain. The constructed network consists of 12 client nodes and 3 server nodes. Each of the client nodes will maintain an outgoing connection to at least one of the server nodes. Each of the server nodes maintains up to 8 incoming connections and 2 outgoing connections with the two neighbouring servers. Each of the nodes runs as a process from one of the Mininet hosts in the test network.

⁵<https://osrg.github.io/ryu/>

Hypothesis 1: Identify the first relay

BTMA is a Ryu application. The application was developed to identify the first relay of a transaction by observing Bitcoin traffic at the controller. Bitcoin operates over port 8333. However, to avoid relying on port number monitoring, the application logic identifies Bitcoin packets for monitoring based on their protocol, TCP, and the presence of a unique 4-byte magic command in their header⁶. The monitoring process is illustrated in Figure 4. Bob generates a transaction of 10 BTC and uses Alice's address as the output address. The packet containing the transaction message is destined to Bob's only outgoing connection, 'Network'. The switch flow table does not contain a rule to handle the packet, so sends an OF message to the controller. The controller, running our application, recognises the packet as a Bitcoin transaction message and logs the source and destination IP addresses and timestamp. The packet is then routed to the server node, 'Network', which propagates the received transaction to the only other connected peer, Alice. Once more, the switch redirects the packet to the controller, which then parses the packet and routes it to Alice. The BTMA thus records the first relay information.

Algorithm 1 Pseudocode to parse transaction hash/index

```

1: function PARSE_BITCOIN_TX
2:    $TX_{IDS} = []$ 
3:   if segwit then offset=14
4:   else offset=10
5:   end if
6:   for  $tx_{in} \in tx_{in\_count}$  do
7:     From initial_offset, get first 32-byte transaction
       hash
8:     set  $TX_{HASH}$ 
9:     From initial_offset, get next 4-byte index
10:    set  $TX_{INDEX}$ 
11:     $REVERSE(TX_{HASH}, TX_{INDEX})$ 
12:    Append to  $TX_{IDS}$ 
13:    Calculate length of  $tx_{in}$ , increase offset
14:   end for
15:   return  $TX_{IDS}$ 
16: end function

```

Hypothesis 2: Extract the Transaction ID

As described in Section III and shown in Figure 2, the input of a transaction is made up of at least one unspent output from a previous transaction (UTXO). As illustrated in Figure 5, this can be identified by a unique transaction ID (hash), TX_{ID} , and index. Furthermore, this TX_{ID} , *index* tuple can be directly linked to a users' wallet. By extracting the UTXO(s) from the payload of Bitcoin transaction messages, the TX_{ID} and index can then be used to uniquely identify transaction messages as seen from the controller.

BTMA parses the transaction hash and index from the payload as per Algorithm 1. First, the function defines an empty list and checks whether the 2-byte *segwit* flag exists

to determine an initial offset. Then, for each transaction input (tx_{in}), the transaction hash and index are extracted, based on the offset and the expected size of each field. A call is made to reverse the transaction input (from big to little endian), and the result is appended to the TX_{ID} list. At each iteration, the remaining tx_{in} length is calculated and the offset is set before the next tx_{in} is parsed. The resulting list of transactions (TX_{ID}) is returned to the main application function. Further

Field Size	Field Name	Description/Example
32	hash	Hash of previous output transaction
4	index	Index of the previous output transaction
1+	script length	The length of the signature script
?	signature script	Computational Script for confirming transaction authorization
4	sequence	Transaction version as defined by the sender

Fig. 5. Structure of the tx_{in} field of the Bitcoin tx message

additions to the code were made to log the parsed data to an in-memory table; for every transaction captured and parsed by the application, the IP source (IP_{src}), IP destination (IP_{dst}), transaction input (TX_{ID}) and timestamp are recorded for further analysis.

Hypothesis 3: Create IP/TX_{ID} Mappings

Using the outcomes of hypotheses 1 and 2, we run the BTMA and feed a pre-defined list of transactions as input to the Bitcoin network. Once the transactions are processed, the BTMA produces a data-frame containing a list of TX_{IDS} and IP_{src} tuples. An extract of the results is shown in Figure 6. The first record implies that a Bitcoin node at 10.0.0.5 is the original sender of the transaction identified by hash $03c01d7ac03d62a7e597780b0c1a84c7001f82b71508f6c812be7b86b018f5d2$ at index 0. To verify this, the TX_{ID} is compared against the original input, as shown in Figure 7.

src	tx_hash	
10.0.0.5	{'tx': [{'03c01d7ac03d62a7e597780b0c1a84c7001f82b71508f6c812be7b86b018f5d2'}	'00000000']})
10.0.0.14	{'tx': [{'03c01d7ac03d62a7e597780b0c1a84c7001f82b71508f6c812be7b86b018f5d2'}	'01000000']})
10.0.0.9	{'tx': [{'10a7fbfcdb73f6875b5c1bba9da386ff001db864bb3520f67858050634fc335'}	'00000000']})
10.0.0.5	{'tx': [{'178bdfc634d7df89d18ee48b40653fa22c114bb4215f7b6c9417dd8cd61564b'}	'00000000']})
	{'9e90dc7201ba98a786103382ff0e8a9d05350726b6abedbe2e0860db25ef96fd'}	'00000000']})
10.0.0.9	{'tx': [{'178bdfc634d7df89d18ee48b40653fa22c114bb4215f7b6c9417dd8cd61564b'}	'01000000']})
10.0.0.5	{'tx': [{'1cb34680d62e762145fb538697e9e884135761e5f0435ee0b1c3ce1491ab5aff'}	'00000000']})
10.0.0.1	{'tx': [{'4a3fd6d6a428b940f87c9bf90948b3a74be780a19c18fee8228452d774df155'}	'00000000']})
	{'ea96ff0ba4733b923953fb05c697702e89b3948cb8653c96c7864410e5c392d5'}	'00000000']})
	{'7fa79173f1ae3d7c6bd6e26a0cf00ee2b68db4f90c612b0c88e7162199b8bb6'}	'01000000']})

Fig. 6. IP/TX_{ID} mappings produced by the application

input ip	txid	seq number
10.0.0.5	a95899cc66dd465a9f603da77800be080e926075672b06e15a6614832ed6ed4e	0
10.0.0.5	ecd5b3343fe7d94fdb4c7c539e60af9c588b707e35362ee7a21912c30a6360917	0
10.0.0.5	d0032f8c85ede1d60b080d63ff5d433a5f862625aedaae5c44b4c7b011c30f	0
10.0.0.5	03c01d7ac03d62a7e597780b0c1a84c7001f82b71508f6c812be7b86b018f5d2	0
10.0.0.1	afbb52f6ba585295e9a294b3de38ad20989d843ed0b3448f12cd52854919e69	0
10.0.0.14	03c01d7ac03d62a7e597780b0c1a84c7001f82b71508f6c812be7b86b018f5d2	1

Fig. 7. Sample of test transaction input

Note that the *txid* and *seq number* in this case together make up the TX_{ID} . For every TX_{ID} , a query was performed to check that the IP_{src} from the results of this experiment matched the original input data. The output of this query revealed that every transaction generated was assigned to its correct source address. This proves that BTMA is capable of producing (IP_{src}, TX_{ID}) mappings with 100% accuracy thus proving hypotheses 1-3.

⁶https://en.bitcoin.it/wiki/Protocol_documentation

Hypothesis 4: Get TX_{ID} /BitcoinAddress Mappings

For this experiment, bitcoin-abe⁷ was used to generate a visualisation of our private Blockchain in a web interface. A subset of the transactions from the pairings generated in the previous step were randomly selected. Each TX_{ID} was looked up in the public Blockchain of our private network, and its' related Bitcoin address ($addr_{src}$) was extracted to create another table of ($addr_{src}, TX_{ID}$) pairings. To verify that each transaction was matched against the Bitcoin address it belonged to, another query was executed for every TX_{ID} against the original input data. The results revealed that the correct ($TX_{ID}, addr_{src}$) mappings were identified with 100% success. Furthermore, for any particular TX_{ID} , the following mappings hold: $TX_{ID} \rightarrow addr_{src}$ and $IP_{src} \rightarrow TX_{ID}$. By the rule of transitivity, it can be inferred that: $addr_{src} \rightarrow IP_{src}$. Therefore, ($IP_{src}, addr_{src}$) pairings can also be deduced.

The results presented with the methodology confirm the defined hypotheses and suggest that the objective of deanonymizing Bitcoin users can be achieved in an SDN environment.

V. PERFORMANCE EVALUATION

A. Bandwidth and Latency

These tests were performed to measure the impact of the BTMA on the speed of data traversal across the network. Firstly, the bandwidth between two nodes in the test network was measured using the *iperf* utility. The latency was measured with a simple 'ping' test of 15 packets. Each test was repeated 10 times with and without the BTMA running. The results are provided in Table II.

TABLE II
BANDWIDTH AND LATENCY WITH AND WITHOUT BTMA

	Bandwidth (Mbps)		Latency (ms)
	H1 $\rightarrow$ H2	H2 $\rightarrow$ H1	Min/Max/Mean
With BTMA	17.7	21.0	2.012/4.371/3.056
Without BTMA	26.0	26.2	0.034/0.158/0.060

As illustrated in Table II, and, as expected, there is a significant impact on latency and bandwidth when the BTMA is running. Without BTMA, only the first packet of a flow is sent to the controller; the simple learning switch application adds flow entries to the switch flow table to process subsequent packets of a flow. However, the BTMA is processing each packet in order to monitor Bitcoin transactions. For a scalable implementation, the BTMA logic can be adapted to only redirect Bitcoin transaction packets to the controller.

B. CPU and Memory Usage

The CPU and Memory usage were also measured with and without the overhead of the BTMA. Given the additional processing power required to handle and parse Bitcoin transactions, this test was carried out over 5 different scenarios with different levels of Bitcoin transaction activity.

As shown in Table III, the processing power required increases with the amount of Bitcoin transaction activity. For the small test network, this impact on CPU resources is limited. However, as more nodes join the network, more message propagations will occur for each transaction and more transactions are likely to be generated leading to a greater strain on the CPU while running the BTMA. There is a negligible increase in memory usage with increasing transaction activity. The small increases in memory allocation reflect the increasing size of the BTMA transaction table, which stores the transactions. In a deployed system, the transaction logs could be periodically stored to an external database.

TABLE III
BTMA CPU AND MEMORY USAGE

Test No.	CPU Usage Mean/Max (%)	Memory Usage Mean (MB)
Test 1: No Bitcoin node processes	0 / 1	65.102
Test 2: Nodes connected, no tx	0 / 1	65.102
Test 3: Nodes connected, 1 tx/30 s	0.7 / 8	65.234
Test 4: Nodes connected, 5 tx/30 s	1.9 / 10	65.484
Test 5: Nodes connected, 20 tx/30 s	4.5 / 19	65.734

VI. DISCUSSION AND FUTURE WORK

Monitoring P2P traffic in a distributed network and deducing relationships between IP addresses and network packets is a complex and arduous task. Consequently, existing methods present probabilistic approaches, which require significant time and resources to make predictions on likely transaction owners. Furthermore, the design of these methods depends on the relay behaviour of nodes based on a trickling mechanism that has since been deprecated. To overcome these limitations, we have proposed a deterministic approach to deanonymize bitcoin users by utilizing the P2P network traffic that does not exploit the behaviour of relay patterns and is therefore robust against the changing nature of relay mechanisms.

A. Tracking Illicit Activity - Example

As shown in Section IV, transaction IDs captured at the controller can be directly linked to Bitcoin addresses by looking them up on the Blockchain. This also means that direct links can be made between Bitcoin addresses and their originating IP addresses. To show how this approach and existing clustering techniques can complement one another, suppose that a large cluster of Bitcoin addresses have been linked to illicit financial activity, but after scraping the web for external information, nothing links these addresses to a real identity. However, if one of the transactions linked to this pool of users was captured by this application, the whole ring of addresses could be revealed. Suppose that a particular Bitcoin address *2NEaj9...ET2Gu7* has been associated with fraudulently obtaining over 300 BTC from another user, identified by addresses *2N5TNW...5JnVEp* and *2N31Uo...3Kon2r*. The transaction obtained from the Blockchain is shown in Figure 8. Using the BTMA output, TX_{ID} *d8b131...* can be looked

⁷<https://github.com/bitcoin-abe/bitcoin-abe/>

up to link *2NEaj9...ET2Gu7* to an IP address. As shown in Figure 9, it is linked to IP address 10.0.0.6.

Index	Previous output	Amount	From address
0	4a3fd6df64...0	48	2N5TNWsidfAsyRhmhymeMMVRI0PXk5JnVEp
1	ea96f0ba4...0	250	2N5TNWsidfAsyRhmhymeMMVRI0PXk5JnVEp
2	7fa79173f1...1	4.9999668	2N31UoC2EMZwVVsEQ7rAKcPQly4Yu3Kon2r

Outputs

Index	Redeemed at input	Amount	To address
0	be98f8e78d...0	2.9998974	2MyQgZNI4D19YJuVoQXwAJ94RusEmM5mPA
1	d8b1315fa6...0	300	2NEaj9o23nrGmaf6pFTTrUEEp5ei4ET2Gu7

Fig. 8. Blockchain showing the output address suspected of fraudulently obtaining 300BTC

10.0.0.6	{ "tx": [{"b50a50b18a1cf9e7a80c5027995189c8c8259e6370185cc18d6bb4f0b43: "00000000"}], ["d8b1315fa6b64fd5ed40b70cb8740778a283dcae29c38d8e8d4a2a0d5f11c8": "01000000"]]
10.0.0.3	{ "tx": [{"bf3a4751136b5270607c140ac85f937a94c7ed9605d908cdf5c2aedae78d": "01000000"}], ["d8b1315fa6b64fd5ed40b70cb8740778a283dcae29c38d8e8d4a2a0d5f1f": "00000000"]]
10.0.0.6	{ "tx": [{"d8b1315fa6b64fd5ed40b70cb8740778a283dcae29c38d8e8d4a2a0d5f1f": "00000000"}], ["d8b1315fa6b64fd5ed40b70cb8740778a283dcae29c38d8e8d4a2a0d5f11c8": "01000000"]]

Fig. 9. BTMA Transaction ID/IP address mapping

Following this transaction through the blockchain begins to reveal what could potentially be a peeling chain [13]. Using the BTMA mappings, we find additional TX_{IDs} linked to the same IP address linked to the original transaction in question.

Therefore, using the BTMA with existing clustering techniques may not only enable law enforcement to deduce relationships between the perpetrators' Bitcoin address(es) and IP address(es), but provide further evidence of attempts to deliberately obfuscate transaction flows, which could assist in identifying criminal activity.

B. Future Work

The performance limitations of this approach are clear from the results presented in Section V. In our future work, these will be addressed through adaptation of the monitoring logic and extension of the network design to include multiple controller instances to monitor a larger network. Another important factor is consideration of the impact of anonymity enhancing services on the IP traffic. Generally, methods that analyse IP traffic to deanonymize users are not impacted by mixing services designed to obfuscate the input addresses of transactions as seen on the Blockchain. However, this solution does not take into account the extensive use of services such as TOR to obfuscate IP traffic. This will be considered in future work. Finally, the privacy and ethical considerations of such a monitoring technique must be considered. Although cybercriminals are using cryptocurrencies for illicit activity, the privacy of legitimate users should not be compromised. This is a significant consideration for our future work.

VII. CONCLUSION

The aim of this research was to find a solution to address the limitations of existing techniques to deanonymize Bitcoin users by presenting a method to analyse the P2P network traffic in an SDN. An SDN Bitcoin transaction monitoring application has been presented that is capable of capturing and analysing the transaction traffic generated by every Bitcoin node in the network. The results demonstrate that it is possible to map transactions to generate Bitcoin address and IP address

pairings. The solution is robust against the changing nature of relay mechanisms, and has the potential to assist in tackling the challenge of Bitcoin-related crime.

REFERENCES

- [1] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.
- [2] M. C. K. Khalilov and A. Levi, "A survey on anonymity and privacy in bitcoin-like digital cash systems," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2543–2585, 2018.
- [3] "History of bitcoin," https://en.wikipedia.org/wiki/History_of_bitcoin, Bitcoin Wiki.
- [4] "How bad is bitcoin crime?" <https://www.worldgovernmentsummit.org/observer/articles/how-bad-is-bitcoin-crime>, World Government Summit.
- [5] "Internet Organised Crime Threat Assessment (IOCTA) 2019," https://www.europol.europa.eu/sites/default/files/documents/iocta_2019.pdf, Europol 2019.
- [6] A. M. Aranda, "Software-defined networking: Current state, adoption factors and future impact on network engineers," 2016.
- [7] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A survey of security in software defined networks," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 623–654, 2015.
- [8] S. Shin, L. Xu, S. Hong, and G. Gu, "Enhancing network security through software defined networking (sdn)," in *2016 25th international conference on computer communication and networks (ICCCN)*. IEEE, 2016, pp. 1–9.
- [9] "Securechain," <https://www.reply.com/en/content/securechain>, Reply.
- [10] M. Steichen, S. Hommes, and R. State, "Chainguard—a firewall for blockchain applications using sdn with openflow," in *2017 Principles, Systems and Applications of IP Telecommunications (IPTComm)*. IEEE, 2017, pp. 1–8.
- [11] Z. A. El Houada, L. Khroukhi, and A. Hafid, "Chainsecure—a scalable and proactive solution for protecting blockchain applications using sdn," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.
- [12] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *Security and privacy in social networks*. Springer, 2013, pp. 197–223.
- [13] S. Meiklejohn *et al.*, "A fistful of bitcoins: characterizing payments among men with no names," in *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 2013, pp. 127–140.
- [14] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Evaluating user privacy in bitcoin," in *International Conference on Financial Cryptography and Data Security*. Springer, 2013, pp. 34–51.
- [15] S. Lee, C. Yoon, H. Kang, Y. Kim, Y. Kim, D. Han, S. Son, and S. Shin, "Cybercriminal minds: an investigative study of cryptocurrency abuses in the dark web," in *Network and Distributed System Security Symposium*. Internet Society, 2019, pp. 1–15.
- [16] P. Koshy, D. Koshy, and P. McDaniel, "An analysis of anonymity in bitcoin using p2p network traffic," in *International Conference on Financial Cryptography and Data Security*. Springer, 2014, pp. 469–485.
- [17] A. Biryukov, D. Khovratovich, and I. Pustogarov, "Deanonymisation of clients in bitcoin p2p network," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 15–29.
- [18] P. L. Juhász, J. Stéger, D. Kondor, and G. Vattay, "A bayesian approach to identify bitcoin users," *PloS one*, vol. 13, no. 12, p. e0207000, 2018.
- [19] G. Fanti and P. Viswanath, "Deanonymization in the bitcoin p2p network," in *Advances in Neural Information Processing Systems*, 2017, pp. 1364–1373.
- [20] "Coinjoin: Bitcoin privacy for the real world," <https://bitcointalk.org/index.php?topic=279249.0>, bitcointalk.
- [21] D. Kreutz, F. Ramos, P. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan 2015.
- [22] P.-W. Tsai, C.-W. Tsai, C.-W. Hsu, and C.-S. Yang, "Network Monitoring in Software-Defined Networking: A Review," *IEEE Systems Journal*, 2018.
- [23] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*. ACM, 2010, p. 19.