2022-09-01

An empirical comparison of the security and performance characteristics of topology formation algorithms for Bitcoin networks

Muntadher Sallal
University of Bournemouth, msallal@bournemouth.ac.uk

Ruairí de Fréin
Technological University Dublin, ruairi.defrein@tudublin.ie

Ali Malik
Technological University Dublin, ali.malik@tudublin.ie

See next page for additional authors

Recommended Citation

Muntadher Sallal, Ruairí de Fréin, Ali Malik, Benjamin Aziz, An empirical comparison of the security and performance characteristics of topology formation algorithms for Bitcoin networks, Array, Volume 15, 2022, 100221, ISSN 2590-0056, DOI: 10.1016/j.array.2022.100221

Funder: SFI

Authors
Muntadher Sallal, Ruairí de Fréin, Ali Malik, and Benjamin Aziz

An empirical comparison of the security and performance characteristics of topology formation algorithms for Bitcoin networks

Muntadher Sallal and Ruairí de Fréin and Ali Malik and Benjamin Aziz
Ollscoil Teicneolaíochta Baile Átha Cliath,
Campas na Cathrach,
Ireland.
web: https://robustandscalable.wordpress.com

in: Array. See also BibTeX entry below.

BibTeX:

```
@article{Sallal22Empirical,
author={Muntadher Sallal and Ruair\'{i} {de Fr\'{e}in} and Ali Malik and Benjamin Aziz},
journal={Array},
title = {An empirical comparison of the security and performance
characteristics of topology formation algorithms for Bitcoin
networks},
year = {2022},
volume = {15},
pages = {100221},
issn = {2590-0056},
doi = {https://doi.org/10.1016/j.array.2022.100221},
url = {https://www.sciencedirect.com/science/article/pii/S2590005622000613},
keywords = {Bitcoin, Blockchains, Clustering, Information propagation, Security, Performance}
}
```


document created on: January 13, 2023
created from file: main.tex
cover page automatically created with CoverPage.sty
(available at your favourite CTAN mirror)

An Empirical Comparison of the Security and Performance Characteristics of Topology Formation Algorithms for Bitcoin Networks

Muntadher Sallal[a], Ruairí de Fréin[b], Ali Malik[b], Benjamin Aziz[c]

[a]*Department of Computing and Informatics, Bournemouth University, United Kingdom*
[b]*School of Electrical and Electronic Engineering, Technological University Dublin, Ireland*
[c]*School of Computing, University of Portsmouth, United Kingdom*

Abstract

There is an increasing demand for digital crypto-currencies to be more secure and robust to meet the following business requirements: (1) low transaction fees and (2) the privacy of users. Nowadays, Bitcoin is gaining traction and wide adoption. Many well-known businesses have begun accepting bitcoins as a means of making financial payments. However, the susceptibility of Bitcoin networks to information propagation delay, increases the vulnerability to attack of the Bitcoin network, and decreases its throughput performance. This paper introduces and critically analyses new network clustering methods, named Locality Based Clustering (LBC), Ping Time Based Approach (PTBC), Super Node Based Clustering (SNBA), and Master Node Based Clustering (MNBC). The proposed methods aim to decrease the chances of performing a successful double spending attack by reducing the information propagation delay of Bitcoin. These methods embody proximity-aware extensions to the standard Bitcoin protocol, where proximity is measured geographically and in terms of latency. We validate our proposed methods through a set of simulation experiments and the findings show how the proposed methods run and their impact in optimising the transaction propagation delay. Furthermore, these new methods are evaluated from the perspective of the Bitcoin network's resistance to partitioning attacks. Numerical results, which are established via extensive simulation experiments, demonstrate how the extensions run and also their impact in optimising the transaction propagation delay. We draw on these findings to suggest promising future research directions for the optimisation of transaction propagation delays.

Keywords: Bitcoin; blockchains; clustering; information propagation; security; performance.

1. Introduction

Bitcoin is the first digital currency to attract the attention of the mainstream business community as well as the private citizen. It is a virtual, decentralised software and cryptography-based system. Its main advantages are that no one is in charge of it and it is not tracked by any hard asset or government [1]. It is operated on a peer-to-peer network where the Bitcoin's value is protected by means of cryptography, which is performed by peers by brute-forcing the double SHA-256 hash function.

Bitcoin relies on a distributed trust mechanism which is achieved by a publicly distributed ledger that is shared across the entire Bitcoin network of nodes [2] [3]. This mechanism acts as a monitoring technique, which tracks the number of available bitcoins. In this paper the term *bitcoin* refers to the actual currency, while *Bitcoin* indicates the whole Bitcoin system. To function successfully, two main requirements need to be fulfilled: (i) transactions verification has to be performed in a distributed manner to ensure the validity of transactions, and (ii) successfully processed transactions have to be quickly announced to everyone to guarantee the state of the blockchain is consistent [4] [5]. As transactions are validated against the blockchain, achieving a consistent state over the blockchain is a fundamental requirement for implementing a distributed transaction verification process. Once a transaction has been verified, it needs to be broadcast to all the nodes in the network so that consensus is achieved about the transaction's validity. Eventually, the consensus is reflected on the blockchain. The most pressing concern of the Bitcoin network is to propagate Bitcoin information to the entire network as quickly as possible. Increasing the speed of this process increases the probability of reaching a global state in the blockchain, which is significantly affected by how quickly the Bitcoin information is announced to all nodes. Delay in information propagation experienced during the transaction verification process can result in an inconsistent blockchain and makes Bitcoin vulnerable to attack.

The Bitcoin peer-to-peer network topology does not consider proximity criteria, either in terms of physical separation or communication latency between nodes. Upon

*correspondence author: Muntadher Sallal, Ruairí de Fréin, Ali Malik and Benjamin Aziz

Email addresses: msallal@bournemouth.ac.uk (Muntadher Sallal), ruairi.defrein@tudublin.ie (Ruairí de Fréin), ali.malik@tudublin.ie (Ali Malik), benjamin.aziz@port.ac.uk (Benjamin Aziz)

joining the Bitcoin network, a Bitcoin node randomly connects to other nodes in the network. This can create long-distance links between the nodes in the physical network. A consequence of these long-distance links is that Bitcoin information traverses network hops unnecessarily, which causes a delay in the transaction verification process [2; 6]. This delay introduces the potential for conflict between nodes about what constitutes the *true* transaction history, which may lead to successful double spending attacks which are hard to detect in slow networks. Conflict in relation to the validity of a given transaction reduces the chances of achieving a consensus on the same blockchain header, which may cause blockchain forks.

Blockchain forks are created when two blocks are created simultaneously, where each one can be added to the same sub-chain [7; 8]. In the special case where the Bitcoin is subject to the blockchain forks [9] , attackers might be able to update their own transactions history, possibly to rewrite transactions they sent so as to successfully perform double spending attacks [10]. Attackers can secretly mine a branch which contains a transaction that reverses the payment to themselves whilst propagating the merchants transaction. Because blockchain forks are caused by delays [2], reducing propagation delay in the Bitcoin network is crucial, even though in many cases, an agreement between parties on the true transaction history can be achieved with a high probability [11; 12].

This paper aims to address the propagation delay problem by investigating the hypothesis that a network overlay that considers geographical displacement and latency between nodes will reduce information propagation delay. Specifically, this paper contributes and critically analyses new network clustering methods, which are named as follows: Locality Based Clustering (LBC), Ping Time Based Approach (PTBC), Super Node Based Clustering (SNBA), and Master Node Based Clustering (MNBC). We demonstrate that the proposed protocols mitigate the information propagation delay issue which reduces the chances of successful double spending attacks occurring. To complete our security evaluation of these protocols, we investigate the inherent tension between forming organized, low information propagation delay networks and providing robustness to partition-style attacks. We present an analysis of the security implications of these protocols, and show that these protocols can be applied in the Bitcoin network without significantly compromising security.

The rest of the paper is organised as follows. In Section 2, strategies for speeding-up information propagation are discussed. Section 3 presents the problem and lists the contributions. We describe the Bitcoin network and the proposed clustering protocols in Sections 4 and 5. The experimental setup and the performance evaluation results are presented in Section 6. In Section 7, a security evaluation of the proposed protocols is presented. We conclude in Section 8 and outline future research directions.

2. Related Work

We discuss related work on mitigating information propagation delay in the Bitcoin network under four headings: minimising verification time, pipelining information propagation, increasing connectivity and double-spending attack mitigation.

2.1. Minimising Verification Time

Several works have considered reducing the information propagation delay by minimising the time taken to complete information (transactions or blocks) verification. When a node receives a transaction, it verifies whether it is valid or not. If the transaction is valid, the node forwards it to its neighbours. Alternatively, invalid transactions are discarded. The idea of reducing block verification time was adopted in [2]. The authors proposed the *minimise verification* protocol as a way to speed up information propagation. The protocol changes the behaviour of Bitcoin nodes. Only the first part of the block verification process is performed by each note. When a node receives a block, it checks the "proof-of-work difficulty" and forwards the block to its neighbours, rather than suspending the relay until the validation of all transactions in the block has been completed. However, this behaviour change is likely to introduce security risks, for example, discarding the transaction validation process would allow an attacker to flood the network with invalid transactions. This type of attacks is commonly known as a Distributed Denial of Service (DDoS) attack. The change in the nodes' behavior does not take into account the transaction propagation delay, which means the transactions would be propagated following the original information broadcasting scenario. As a result, the change does not have a significant positive impact on the overall information propagation delay.

The approach proposed by [13] focused on the blockchain as a main factor in reducing the transaction verification time. As transactions are validated against the blockchain, which contains a history of all transactions and which grows in the size with each new transaction added, the authors claim that by reducing the transactions history at each node, this would play an important role in reducing the transaction verification time. An algorithm, known as BASELINE, was proposed in [13], in which the blockchain is divided at each node in the Bitcoin network and into n parts. These parts are then distributed on several local computers at each node. As all parts represent the same user, the used public/private keys will be the same for all those parts. On the other hand, each part has a different portion of the public ledger. Results in [13] demonstrated that the verification time can be reduced by 71.42% if the blockchain is divided at a given node on five computers. It is hypothesised that an improvement in the information propagation delay could be achieved when the number of divisions at each node, n, was increased. The proposed *BASELINE* algorithm is unlikely to be adopted as a deployed solution due to the expensive requirement

that every node in the network should maintain several local computers.

Research that focused on speeding-up information propagation in conjunction with minimising the blockchain size was proposed in [8]. This approach improved the scalability of the blockchain by increasing the security for off-chain blocks using the miners. In this approach miners are responsible for keeping track and protecting the soft forks that are linked to the main blockchain. Miners are considered to be a trusted third party and the approach provides them with more control over the Bitcoin network. This approach is contrary to the decentralisation concept of Bitcoin; it results in a reduction in security awareness. Such soft forks are subject to the so-called 51% attacks due to their reduced hash rates.

The Blinkchain approach, which focused on minimising the transaction verification time, with the aim of decreasing the consensus latency, was introduced in [14]. The Blinkchain approach was based on splitting the blockchain into localised *shards*, one blockchain per geographical location. Each blockchain was associated with a number of nearby validators. This reduced the transaction history at each blockchain, which resulted in a speed-up in the transaction verification time. This approach reduced the resistance of the blockchain against 51% attacks as these blockchains offer a reduced hash rate. It did not support interoperability, which meant that shard blockchains could not interact with each other. A sharding approach called Rapidchain was also introduced by the authors of [15] to scale-up a blockchain. In Rapidchain, the blockchain network is divided into a random number of shards, where each shard randomly selects a leader node. Shards in Rapidchain are not defined based on proximity, and network information is still needed to travel long distances. Shard leaders in Rapidchain are not forced to fulfill specific requirements, which is a significant shortcoming of the network from a security point of view.

2.2. Pipelining Information Propagation

The model introduced in [6] aimed at achieving faster information propagation, by pipelining information dissemination, which reduces the round-trip latencies between nodes in the network. Specifically, nodes could immediately forward *INV* messages that contained hashes of disseminated transactions to the other nodes instead of waiting for the reception of the actual transaction data. This meant that a received transaction could be immediately propagated to those nodes that asked for it and that had already sent *GETDATA* messages as a reply to the *INV* messages. As a result, the network initialised the idle time typically used by nodes waiting for *GETDATA* messages. The key problem with the pipelining propagation protocol was that the global state of the Bitcoin network could potentially become inconsistent when nodes requested a transaction that was not available. This increased the chances of successful double-spending attacks being performed. The pipelining propagation protocol required unlimited memory at every node with the aim of either keeping transactions until a *GETDATA* message had arrived or keeping a *GETDATA* message until transactions had arrived. The authors suggested that this had minimal impact on the information propagation delay as transactions still needed to pass through random, non-localised connections to be disseminated to the entire Bitcoin network of nodes. Another pipeline method called *Compact-Block Relaying* (CBR) was introduced in [16] to mitigate the propagation delay problem. A compact-block that includes only hashes of transactions in the block is announced to other nodes. Upon receiving the compact-block, only missing transactions are transmitted to the receiver, rather than the whole block. Even though the CBR method improves propagation delays, nodes still require a compact-block on hand before forwarding it on. The CBR method has potential to cause large latency, especially when transmitting compact blocks of large sizes. Finally, Falcon, a propagation protocol proposed in [17], attempts to minimise propagation delays by following the cut-and-forward strategy, in which the reception and forwarding of a compact block is handled in parallel. However, Falcon does not rely on existing Bitcoin nodes, instead, it deploys relay nodes to implement the cut-through forwarding protocol. In addition, Falcon is a commercial protocol, which lacks in-depth publicly available analysis.

2.3. Increasing Connectivity

The network distance between the initiator of a block and the nodes is deemed to be one of the most important causes of the propagation delays in Bitcoin. The study in [2] claimed that information propagation delays could be improved by increasing network connectivity. This can be achieved by creating a star sub-graph topology, which forms a central communication hub between nodes. A novel network topology was proposed in [2], in which each node maintains a connection pool capable of maintaining up to 4000 open connections. In this set-up, nodes are typically connected to every single advertised address. Information traverses smaller number of hops, which explains the reduced information propagation times observed. The Bitcoin protocol allows nodes to maintain up to 8 outgoing connections to prevent the network from being controlled by malicious nodes [18]. Unfortunately, the proposed network topology introduces severe security risks due to the fact that nodes are permitted to maintain many connections to other nodes. This may enable malicious nodes to disturb and control the network.

Maximising proximity when establishing connectivity is the aim of the approach proposed in [6]. This change increases the geographical connectivity of the Bitcoin network by making use of several coordinator nodes, known as CDN Bitcoin clients. These CDN Bitcoin clients are then distributed strategically across the Bitcoin network. Their role is to search and recommend Bitcoin network nodes to each other, based on geographical locations. A CDN client measures the geographical distance between

the discovered nodes and other CDN clients. By doing so, the CDN client can suggest which nodes are closest geographically to other CDN clients. Compared to the protocol proposed in [2], CDN clients are allowed to maintain up to 100 outgoing connections to nodes that are considered to be geographically close. The main disadvantage of this solution is that any node can become a CDN client, which reduces Bitcoin's resistance against some classes of attacks. Malicious nodes can easily impersonate the role of CDN clients, and maintain connections to many nodes in the network. This results in malicious nodes being able to control big portions of the network. The resulting Bitcoin network is vulnerable to DDoS and partition attacks. Another concern raised is that the solution is relatively centralised; any CDN client can be used as a coordinator node, without meeting any requirements or achieving an agreement over network nodes. The idea of recommending closer nodes to other nodes does not have a high impact on the overall network connectivity, if it is implemented by a limited number of nodes that are not well connected.

A transport protocol layer known as FIBER (Fast Internet Bitcoin Relay Engine), was introduced in [19] to reduce the information propagation delay. FIBER focused on the reduction of the delay caused by packet losses incurred in the UDP layer with forward error correction. FIBER reduces network traffic by using data compression. The approach in this paper introduces a Bitcoin network protocol that is easily integrated with FIBER. Finally, an optimisation protocol proposed in [20] increases network connectivity by making use of geographical proximity clustering. The k-means algorithm is used to gather proximity peers into clusters. However, their paper does not carry out security evaluations to test if the proposed clustering protocol compromises security. In contrast, in this paper we evaluate the security of several brand new clustering protocols. As far as we are aware, this is the first work in which such a contribution is presented.

2.4. Double Spending attack mitigation

Mitigating double-spending attacks in two scenarios, 0-confirmation and N-confirmation, has received much attention in literature. In the case of N-confirmation, the probability of performing a successful attack was measured in [2] by developing an analytical model of Bitcoin. The authors in [2] observed some correlation between the propagation delay and the size of a message. As adversarial forks of the blockchain can still introduce the possibility of double spending, the contributions in [8; 9] suggested that reducing the possibility of accidental forks would help avoiding double-spending attacks. For the case of 0-confirmations, the authors of [9; 21] presented modifications of the transaction dissemination protocol as one possible solution for mitigating double-spending attacks in fast payments. A model was proposed in [9], which allows a vendor to receive conflicting transactions, T_K, and honest transactions that are then sent to the vendor, T_v, nearly at the same time. The approach allows a vendor to discover double-spending attacks at the right time before delivering the products. A node adds a transaction to its pool and forwards it to the other nodes if the transaction is received for the first time. In the case where the received transaction has already been seen, the node forwards the transaction without adding it to its pool. This enables the reception of the conflicting transaction, T_k, by the vendor prior to product delivery. The downside of this model however is that the network may become flooded by nonessential traffic, leading to degradation in the performance of Bitcoin.

Finally, a prototype system was proposed by [21] to overcome double-spending attacks in vending machines. This system achieves a fast payment with a 0.088 probability of a double-spending attack occurring, by making use of a server that keeps track of transactions. When a transaction is disseminated to more than 40 nodes, the server issues a signal, which indicates that the transaction has been confirmed by the blockchain. This solution is limited because an attacker's transaction could still be delivered to the majority of nodes.

3. Problem Statement

Information propagation delay is a serious problem in the current Bitcoin network. Several models have been introduced to overcome it. Previous attempts to update the network topology have not taken into account the benefits of a clustering approach. They considered either increasing the network connectivity by maintaining a mesh network topology [2], or relying on several coordinator nodes to increase the connectivity based on the proximity of nodes in the network, which was done without paying attention to the security risks involved [6]. We consider if *"clustering in the Bitcoin network can improve information propagation delays without compromising security"*. The main contributions of this paper can be summarised as follows:

Performance Evaluation: We examine the role of clustering in the Bitcoin network to reduce the average latency of information delivery between peers without compromising security. We propose and evaluate four clustering approaches: (1) Location Based Clustering (LBC), (2) Bitcoin Clustering Based Ping Time protocol (PTBC), (3) Bitcoin Clustering Based Super Node (SNBA) and finally, (4) Master Node Based Clustering (MNBC). The LBC protocol aims to improve the connectivity in the Bitcoin network by prioritising geographically close connections between nodes. The PTBC approach seeks to optimise the overlay topology by creating distinct but connected clusters of peers, which have Peer-2-Peer (P2P) latencies specified under some intra-cluster threshold. The aim of the SNBA approach is to generate a set of geographically diverse clusters. The MNBC protocol relies on several nodes, known as masters, to achieve fully connected clusters based on Internet proximity and random

peer selection.

Security Evaluation: As undertaking clustering in the Bitcoin network is different from clustering within other classes of P2P networks, due to the strict security requirements, this paper examines whether clustering can be done safely, without increasing the likelihood of certain classes of attacks, specifically, partitioning attacks. The impact of partitioning attacks on the proposed protocols as well as on the Bitcoin network are evaluated.

Simulations: To evaluate the proposed clustering protocols, several simulations are developed using the simulation model of [22]. To parameterise the simulation model, large-scale measurements of the real Bitcoin network parameters that have a direct impact on a client's behavior and information propagation in the real Bitcoin network were performed. Measurements of the transaction propagation delay in the Bitcoin network are presented. These measurements are collected using a methodology which ensures that the transaction propagation delays are accurately measured. These measurements offer an opportunity to validate the developed simulator against the real Bitcoin network.

4. Background

The Bitcoin network refers to a group of nodes that support the Bitcoin protocol. We outline this decentralised structure.

4.1. Bitcoin Network Structure

Decentrality is one of the key features of Bitcoin. A distributed protocol is maintained to support the system [23]. Each peer runs the Bitcoin protocol and connects with other peers over a TCP channel [24]. As the Bitcoin network topology is not established based on proximity, selecting which peers to connect with, is undertaken randomly. It is a requirement that every node should maintain a maximum of 8 outgoing connections to peers and accept up to 117 connections [25]. Nodes can join and leave the network at any time. When a node re-joins, it asks other nodes for new blocks to complete its local copy of the blockchain [26] [27]. To mitigate DoS attacks, only valid transactions and blocks are propagated on the network [28]. Bitcoin nodes take different roles in the network based on the functionality that they support such as wallet services, routing, etc. As Bitcoin relies on distributed validation, an essential function is that of validating transactions in a distributed manner. This role is performed by all nodes in the network [25]. To participate in the Bitcoin network, all nodes have to support the routing function. This function includes validation and propagation of transactions, and maintaining connections to other nodes.

4.2. Bitcoin Network Discovery

When a node, n, joins the Bitcoin network for the first time, a discovery mechanism that does not consider any proximity criteria finds other nodes in the network. At least one existing Bitcoin node needs to be discovered by the node, n, for it to discover more nodes [29]. More connections are then established between the node, n, and the nodes that are discovered. Establishing connections to other nodes is done without taking into account node proximity as the Bitcoin network topology is not established based on proximity [2; 6]. To establish a TCP connection, a handshake with a known peer is handled by sending a *version message* which contains basic identifying information. A peer responds to the version message by sending a *verack message*. Each peer caches the IP addresses of peers that are connected to it. To stop peers misbehaving, each node assigns a penalty score to each node connected to it. The score is increased when an unreliable behavior is announced. When the score reaches 100, the node with the associate misbehaving IP address is banned by the node that handles the penalty score. A transactions pool is maintained by each node which includes transactions that wait to be verified and to be relayed to the neighboring nodes [24].

Discovery of the first node in the network is now described. The network contains stable nodes that behave as seed nodes. Their identities are listed in the new Bitcoin client as suggested nodes in the network [25]. Bootstrapping that needs to be handled by the new node, requires at least one node's IP address, which is known as the *DNS seed node*. After establishing a connection to the seed node, further introductions to other nodes are then initiated. Once more connections to other nodes have been established, the new node disconnects from the seed node. Connecting to other nodes helps the new node to discover more nodes. This can be done by sending an *Addr message*, which includes the IP address of the sender node. The newly connected node can advertise its own IP to other nodes by sending an *Addr message* to its neighbours. This helps the new node to be found by other nodes. On the other hand, the new node can get to know other nodes by sending a *Getaddr* message to its neighbours and the neighbour nodes respond by disclosing their IP addresses. Even though each node establishes connections to other nodes, the node should continue discovering more nodes and advertising its existence to new nodes as they join the network [24]. This is because paths can be unreliable as nodes can join and depart the network in an unplanned way. A node that connects to other nodes does not do so with the guarantee that these connections will never be lost. The process of discovering other nodes continues to operate so that diverse paths across the Bitcoin network are available. When a node reboots, it can re-join the network without needing to bootstrap the network again as it remembers the most recent successful node connections; the node tries to reestablish connections to those nodes by sending connection requests. If there are no responses,

the node starts bootstrapping the network again. In terms of dropping a connection, if it does not deliver traffic for more than 90 minutes, the connection is dropped [29].

4.3. DNS Seed Nodes in the Bitcoin Network

A Bitcoin DNS seeder is a server that assists nodes in discovering active peers in the Bitcoin network. The DNS seeder responds to the DNS query by initiating a message that contains a list of IP addresses. The maximum number of IP addresses that can be attached to the message is limited by constraints on DNS. Approximately 4000 messages can be returned by a single DNS query [25]. In the Bitcoin network, there are six DNS seeds that periodically crawl the entire network to obtain active IP addresses. There are two scenarios where DNS seeders are queried by other nodes. The first scenario is when a node that joins the network for the first time, tries to connect to active IP addresses. In the second scenario, the DNS seeder is queried by a node that restarts and attempts to reconnect to new peers. In this case, the DNS query is initialised 11 seconds after the node attempted to reconnect and if it has less than two outgoing connections [25].

4.4. Bitcoin Protocol and Information Propagation

The distributed validation mechanism in the Bitcoin protocol relies on a replicated blockchain which is collectively maintained by network miners. The replicated ledger monitors the address balances of all Bitcoin users. Bitcoin users are able to generate an arbitrary number of addresses to send and receive bitcoins. The ownership of bitcoins associated with these addresses can be proven by an Elliptic Curve Digital Signature Algorithm (ECDSA) key pair. Entries within the public ledger are transactions which are generated by users who sent bitcoins to one or more bitcoin recipients [24]. Public ledger transactions are represented by the public key of the recipient as well as the hash of the previous transaction. Each transaction consists of an input which references the funds from other previous transactions, and an output which indicates the transferred bitcoins as well as the new owner of the transferred bitcoin. The sum of all outputs should be less than or equal to the sum of all inputs [25] [30].

By propagating transactions and blocks, nodes synchronise their replica of the public ledger. To avoid sending the transaction to nodes which have already received it, the transaction availability is announced first to nodes, once the transaction has been verified, as shown in Figure 1. This can be achieved by forwarding *INV* messages that contain hashes of disseminated transactions to the rest of nodes [31]. If the transaction has not been received before, the node responds to the *INV* message by sending a *GETDATA* message, requesting the actual transaction. In response to receiving the *GETDATA* message, the node responds by sending the transaction. Valid received transactions are collected and included in a block by a node that generates blocks. A block's availability is

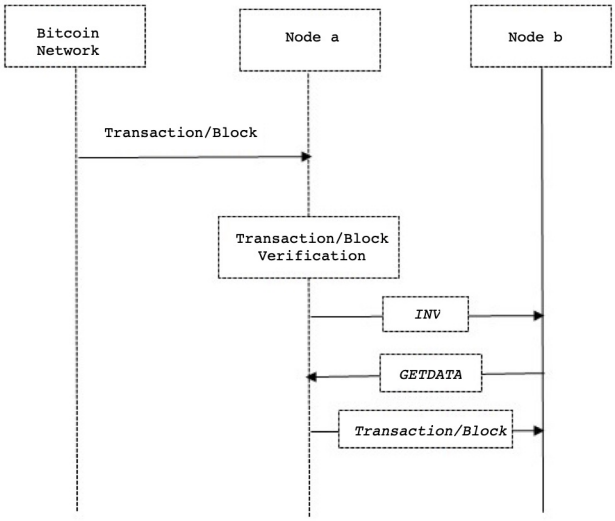


Figure 1: By propagating transactions and blocks, nodes synchronise their replica of the public ledger: The information propagation mechanism between nodes a and b is illustrated.

then announced to other nodes, as explained in Figure 1, following the same mechanism for transaction availability announcement. However, this information broadcasting approach causes a delay in transaction propagation [6].

5. Proposed Clustering Approaches

We introduce clustering techniques to reduce propagation delays, including LBC, PTBC, SNBC and MNBC.

5.1. Locality Based Clustering

Previous approaches that focused on making connections based on the proximity of nodes in the Bitcoin network were vulnerable to significant security implications. Forming networks using this principle went against the decentralisation principle of the Bitcoin architecture. It increased the chances of the network being controlled by allowing each node to maintain more than 8 outgoing connections. In addition, previous approaches were implemented by limited nodes, which were not well connected and which resulted in a low-level impact on information propagation latency. We propose a location-based clustering protocol, named Locality Based Clustering (LBC) that overcomes the security and performance limitations of previous approaches with the aim of maximising the proximity of nodes when establishing connections in the Bitcoin network without compromising security.

To overcome these limitations, a proximity-based network layout is achieved by all nodes using the LBC protocol, which establishes this topology in a distributed manner. This increases the level of security as no single node has full knowledge of the network topology. To evaluate the impact of maximising the geographical proximity when forming connections on the information propagation delay in the Bitcoin network, the LBC protocol groups Bitcoin

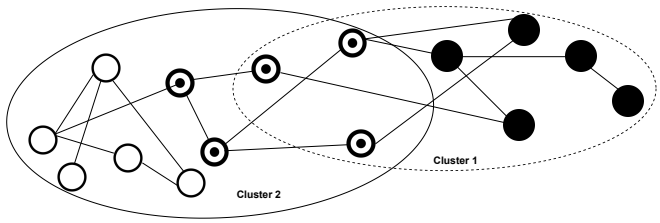


Figure 2: Location-based cluster creation using the LBC protocol: The black dotted nodes represent the border nodes between clusters. The black and white nodes represent the two clusters.

peers based on the geographical closeness of their IP prefixes. This contributes to minimising the network latency between peers, which results in improvements in the information propagation delay. In the LBC protocol, peers' IP addresses are used as basis for defining a local area inside the Bitcoin network. The LBC protocol is measurement based and can dynamically change the network layout and connect geographically closer peers. Every peer in the network connects to other nodes within the same geographical location and forms a cluster. In Figure 2, short-distance links are maintained within each cluster. Clusters are fully connected by their border nodes to support the visibility of the available information from outside the cluster as well as avoiding network partitions. Border nodes between two clusters refer to the two closest nodes belonging to two different clusters.

5.1.1. Localised Cluster Generation

The LBC protocol is run independently by each node using information about discovered nodes and local neighbours. The network is divided into clusters. Nodes in the same location belong to the same cluster. It requires that an extra function is available on each node, which is responsible for recommending proximity nodes to their neighbours. Proximity is defined based on the geographical location of two nodes. It relies on a distance threshold, which identifies the number of clusters and the size of a cluster. Nodes calculate the network geographical distance between their neighbours and the newly discovered nodes. Consider two Bitcoin network nodes i and j. These nodes are geographically close if:

$$D_{i,j} < D_{th} \tag{1}$$

where $D_{i,j}$ is the distance between node i and node j, and D_{th} is the distance threshold. To measure the geographical distance, the LBC protocol uses the haversine formula [32], which calculates the real-world distance between two nodes by making use of their longitude and latitude. The longitude and latitude of nodes i and j are λ_i and ϕ_i, and λ_j and ϕ_j, respectively. For convenience we define the difference in the longitude and latitude between nodes i and j to be $\Delta\lambda_{ij} = \lambda_i - \lambda_j$ and $\Delta\phi_{ij} = \phi_i - \phi_j$. The harversine formula is:

$$a_{ij} = \sin^2\left(\frac{\Delta\phi_{ij}}{2}\right) + \cos(\phi_i)\cos(\phi_j) \times \sin^2\left(\frac{\Delta\lambda_{ij}}{2}\right) \tag{2}$$

where, R, is the earth's radius (mean radius $= 6,371$km [33]). The distance in meters is then calculated using:

$$D_{i,j} = \text{distance}(i,j) = 2R \times atan2\left(\sqrt{a_{ij}}, \sqrt{1-a_{ij}}\right) \tag{3}$$

The MaxMind GeoLite City database is used to retrieve the latitude and longitude of a particular node's IP [34]. For example, when a node, n, discovers another node, k, that is close to k's neighbour, m, the node n sends the IP address of the discovered node k to its neighbour node m as a recommended node to connect with. On receiving the IP address, the node, m, connects to the node, k, and then verifies whether the node, k, is also close to its neighbours. The LBC protocol requires that this process is repeated by the entire set of Bitcoin nodes when recommended nodes are received from their neighbours. It ensures that generated clusters are fully connected by making use of border nodes. This is achieved by selecting border nodes between every pair of clusters. Border nodes are selected to be the closest pair of nodes that belong to two separate clusters. This ensures efficient information dissemination between clusters is achieved, as many transmission channels between clusters are available. Increasing the number of border nodes between clusters increases the difficulty in achieving a partitioning attack on the network. Let $K = \{k_1, k_2, ..., k_m\}$ and $Q = \{q_1, q_2....., q_n\}$ represent the members of two clusters, and let k_b and q_b denote their border nodes, where $k_b \in K$ and $q_b \in Q$, then for all other pairs of clusters, we have:

$$\text{distance}(k_i, q_j) \geqslant \text{distance}(k_b, q_b) \quad \text{such that } k_i \neq k_b, q_j \neq q_b, k_i \in K, q_j \in Q \tag{4}$$

5.1.2. Localised Cluster Maintenance

The Bitcoin network structure exhibits some degree of churn; peer nodes enter and exit the network at arbitrary times. Existing clusters of nodes in the network are influenced by the dynamics in the Bitcoin network structure. A mechanism that handles the node dynamics is required to avoid re-clustering the entire network in response to each node entry and/or exit. As nodes frequently join and leave the network, re-clustering is impractical as the clusters do not get the opportunity to stabilise [35].

Once a node z joins the Bitcoin network, it receives a list of the available Bitcoin nodes from DNS services. Upon receiving a query from the node, z, DNS services probe the node, z, to determine its geographical location, by making use of the same methodology described in Section 5.1.1 to calculate the distance. Based on the probe's results, DNS services check the network and return any known peers close to node z. If none are found, random peers are returned. If the DNS service is close to the node z, it returns all peers that are close to itself. Based on a distance threshold, the node z determines the location-based order of the discovered node by measuring the distance to each discovered node. After that, a *JOIN request* message is sent by the node z to the closest node c, in the set of discovered nodes. After connecting to the node c, the node,

z, connects to the nodes that belong to c's cluster only, as it receives a list of IP addresses of nodes that belong to the same cluster as the node c. No further action is required when the node z leaves the network. From a security point of view, DNS nodes do not impose a significant security risk, even when a newly joined node is forced to connect to attacker nodes. The reason for this is that newly joined nodes normally learn one peer from Bitcoin DNS nodes, and then nodes can use the normal discovery mechanism of the Bitcoin network to find more nodes to connect with.

5.2. Ping Time Based Approach

Nodes that are geographically close might actually be quite far from each other on the Internet and vice versa. For instance, hosts that are directly connected by optical fiber are most likely very "close" when the proximity only takes into account the link latency between network nodes, even if they are physically placed far away from each other. Proximity can be measured using different criteria, such as the physical location and the link latency between peers [36]. We propose a proximity-based latency-awareness protocol, named as PTBC. We evaluate the security and performance impact of connection establishment based on the proximity of the nodes, which is measured using ping latencies, on the Bitcoin network. Based on round trip ping latencies, nodes detect and disconnect most of the inefficient and redundant logical links, and select closer nodes as their direct neighbours. Consequently, peers within each cluster are highly connected via short link latencies. This offers faster information propagation, resulting in a better distribution of Bitcoin information over the network, which helps the Bitcoin network to achieve a consistent state. To maximise security awareness with respect to network partitions as well as ensuring efficient information distribution between clusters, clusters in PTBC are fully connected using border nodes. Border nodes are selected using the same strategy for border node selection in the LBC protocol described in Section 5.1.1 with one difference. Instead of using the distance, distance(x, y), between two nodes, x and y, the distance between two nodes x and y is measured by the link latency, $L_{x,y} = \text{latency}(x, y)$.

5.2.1. Distance calculation

In the PTBC protocol, the distributed algorithm principle is followed. Each node runs the protocol independently based on proximity information collected from local neighbours and discovered nodes. Each node gathers proximity knowledge about the discovered nodes by calculating the Internet distance between itself and the Bitcoin nodes that it has discovered. This can be done by measuring the round-trip latency between two nodes. Two nodes i and j are considered close on Internet if the latency measured between them, $L_{i,j}$ is less than a threshold, L_{th}:

$$L_{i,j} < L_{th} \tag{5}$$

The latency between i and j is measured by the round-trip latency. It is measured using a utility function that calculates the latency between two nodes. When the overlay changes, the node latency information is updated by re-running the latency function. The latency is calculated as follows:

$$L_{i,j} = \frac{M_{ping}}{r} + 2P + q \tag{6}$$

where i and j represent two Bitcoin network nodes and M_{ping} represents the ping message length in bytes. The transmission rate, r, is the total amount of data that can be transferred between two nodes in a given time frame, ≈ 100 KB/hour. The transmission time is denoted P. It is the time taken for a signal to traverse a propagation medium which connects two points. We make the simplifying assumption that multiplying the propagation time by 2 yields a reasonable estimate of the round-trip time. The propagation speed is:

$$P = \frac{D_M}{S} \tag{7}$$

The propagation medium length between nodes i and j is D_M. The speed of light is S. We use the approximation, $3 \times 10^8 m/s$ for free-space propagation (using Wi-Fi internet) and $2/3 \times 3 \times 10^8 m/s$ for guided propagation media, for example copper cable. The queueing time is:

$$q = \frac{M_{ping}}{r - \lambda M_{ping}} \tag{8}$$

where λ is the arrival rate in units of pings per second.

5.2.2. PTBC Cluster Maintenance

Different types of proximity criteria may be applied to influence the node discovery mechanism when a new node interacts with DNS services. A newly joined node, n, learns about the available Bitcoin nodes in the network from the DNS services. The node discovery mechanism takes into consideration that the DNS service might provide sub-optimal peers. Nodes should rank peers in the received list and the decision about which node to connect to should be taken based on this ranking. DNS service nodes take into consideration the proximity in the physical geographical location while recommending peers to the newly joined node n. Relying on the geographical distance calculation methodology that is used in the LBC protocol, DNS services recommend the closest available nodes to the node n. To get the proximity ordering for the discovered nodes based on a link latency threshold, the node n calculates the distance to each discovered node. After determining the ordering based on proximity, the node, n, connects to the closest node, k, in the set of nodes supplied by the DNS service. Upon connecting to the node, k, the node, n, uses the Bitcoin network discovery mechanism to periodically discover other nodes in the network without relying on the DNS service anymore [25]. When discovering new nodes, the node n decides whether these nodes are physically close, by making use of the distance calculation mechanism described in Section 5.2.1. No further action is required when the node n leaves the network.

Similar to the LBC protocol, the Bitcoin DNS service does not pose a serious security risk because the newly joined nodes normally use the Bitcoin network discovery mechanism after connecting to at least one node supplied by the DNS service.

5.3. Super Node Based Approach

The number of hops between peers is one of the factors that influences the measurement of node proximity in P2P networks [36]. Approaches that use the idea of super-peers can contribute to minimising the number of intermediate hops between peers. As the Bitcoin network is a financial instrument that needs to be resilient against active attacks, the super-peer approach introduced in this work enhances previous super-peer solutions [37; 38] whilst also considering security awareness. Firstly, it does not require any network node to have full knowledge of the entire network topology. This property supports the decentralised concept of Bitcoin. Secondly, super-peers are selected based on achieving several conditions in a distributed manner. If a malicious node attempts to impersonate a super-peer, it must overcome the challenge posed by these conditions. In this paper, we propose a super-peer approach, named SNBA, in which the design of the overlay network is composed of several clusters of peers. It selects a peer to be a super-peer and this super-peer becomes a cluster head that propagates network information to other super-peers in different clusters. Super-peers in SNBA can be given an extra function, such as, grouping peers based on a specific criteria. By grouping peers according to their geographical proximity, we can further speed-up information broadcasting in the network. A hierarchical Bitcoin overlay network that clusters nearby peers might achieve faster information propagation than the original Bitcoin system. In this paper, the concept of super-peers is applied to the Bitcoin network to increase connectivity between peers that are close in a geographical sense.

SNBA combines two properties: (1) a reduction in the number of intermediate hops between any two peers and, (2) an increase in the connectivity between geographically close peers. The ultimate goal of the SNBA protocol is to randomly split the Bitcoin network into several geographically diverse clusters by making use of super-peer technology. In SNBA, each cluster elects a node to act as a super-peer, a role that maintains the cluster and broadcasts information in the Bitcoin network. This is the first paper that applies clustering-based super-peer technology in the Bitcoin network. In Figure 3, the SNBA protocol selects several nodes as super peers. Each super-peer connects to the geographically closest nodes and forms a cluster. All super-peers in the network are fully connected and known to each other. SNBA reduces the number of hops that the transaction passes through such that the propagation delay is reduced.

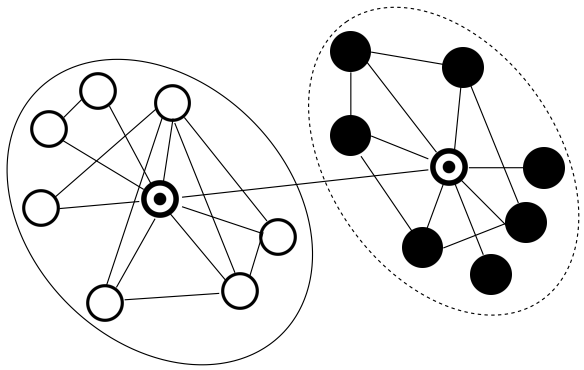


Figure 3: Super-peer cluster creation in SNBA: black dotted nodes represent super-peers in each cluster. Black and white nodes indicate different clusters.

5.3.1. Super-Peer Selection Algorithm

Due to security requirements, the super-peer selection algorithm in SNBA relies on selection criteria which are different from the selection criteria proposed in previous work. The super-peer selection approach in [39] relies on the node unique ID. A node with the lowest ID is more likely to be elected as a super-peer. The super-peer selection approach in SNBA is based on a weight, a positive real number, which is assigned to each node. The weight is computed based on two features: how long each node has been online and how many bitcoins are spent by each node. Using these inputs for the selection criteria makes impersonation of a super-peer challenging. A node with the highest weight is more likely to be elected as a super-peer. A reward is used in the SNBA approach to encourage information propagation in the Bitcoin network. Super-peers that propagate a valid transaction and behave honestly are given a reward. This reward acts as an incentive for nodes to win the super-peer's role. When a super-peer goes offline, each cluster selects a backup peer, which copies the entire cluster state information periodically from the super-peer. The backup peer is selected using the same mechanism and criteria as that for super-peer selection.

Node stability is one of the key parameters when calculating a weight for each node. A penalty score which is based on how long a node has been online, is calculated for each node by its connected nodes. The penalty score for a node is increased by 1 by its connected nodes when the node goes offline. After that, the super-peer is sent the updated score from those nodes that increased the score. The super-peer circulates the updated score to all of its connections once the super-peer's record is updated.

Super-peer selection relies on two types of message *SupINV* and *AcceptINV*. A node, k, that is willing to be a super-peer, invites its connected nodes by sending a *SupINV* message, which includes the node's ID and

Algorithm 1: SupINV handler function

```
Let k as: nearest superpeer() with Bigger weight
Let s as: current superpeer
if s ≠ k then
    s = k
    connectTo (k)
    Forward (SupINV)
else
    Forward (SupINV)
end
```

Algorithm 2: Peer joining algorithm

```
Let P as: Super-peers set
Let z as : new peer to join the network
while P ≠ 0 do
    d ← distance(z, s_l) where ∀s_l ∈ P
    d_1 ← distance(z, s_j) where ∀s_j ∈ P
    if d < d_1 then
        z ← connectTo s_l
    else
        z ← connectTo s_j
    end
end
```

weight. In Algorithm 1, the invitation is accepted by the node m if the node k is geographically closer and has a bigger weight than the current super-peer. Node m decides whether or not k is geographically close to it by calculating the geographical distance. Node m sends an *AcceptINV* message when accepting k's invitation. Node m should forward the *SupINV* message to its neighbour nodes. They which in turn disseminate the *SupINV* message further.

5.3.2. Peer Joining Algorithm

The second phase of the SNBA protocol is a cluster maintenance protocol which handles the entry and exit of nodes in the Bitcoin network. Let $M = \{k_1, k_2, \ldots, k_i, \ldots\}$ be a set of peers in the Bitcoin network, where $|M|$ is the number of peers. Let $P = \{s_1, s_2, \ldots, s_j, \ldots\}$ be a set of super-peers, where $|P|$ is the number of super-peers and $P \subseteq M$. Let $Sp_l = \{s_l, b_1, b_2, ..., b_n\}$, be the set of nodes in the lth cluster. We have $Sp_l \subseteq M$ and $M = Sp_1 \cup Sp_2 \cup \ldots \cup Sp_{|P|}$. When a node z joins the network for the first time, it first uses the DNS service to contact a random node k which helps by introducing the available super nodes in the network. The node z is then sent a list of the known super-peers by the node k. According to the peer joining algorithm described in Algorithm 2, the node z selects a super-peer s_l, such that $\forall q \in P, \text{distance}(z, s_l) \leq \text{distance}(z, q)$. Then, a *JoiningRequest* message is sent to the selected super-peer by the node z. Note that distance(x, y) refers to the geographical distance between the nodes in the network. This distance is calculated using the method in the LBC protocol, in Section 5.1. To allow the node z to connect to the nodes that belong to the Sp_l cluster only, an *Acceptance* message which includes a list of node addresses that the cluster Sp_l connects with, is sent to the node z via the super-peer s_l. When the node z leaves the network, it sends a disconnect message to its super-peer, which requires no reply. Once the node z joins the Bitcoin network, it sends metadata over its connections to its super-peer. At the same time the super-peer adds the node z to its index.

5.4. Master Node Based Clustering

The master node based clustering approach, known as MNBC, extends the SNBA protocol that was proposed in [27], with the aim of addressing security and performance limitations of the BCBSN protocol [40]. As discussed in [27], the SNBA protocol aims to generate a set of geographically diverse clusters in the Bitcoin network by exploiting super-peer technology. Within each cluster, the SNBA protocol assigns one node to be a super-peer. This node is responsible for maintaining the cluster and broadcasting information in the Bitcoin network. In the SNBA protocol, clusters are fully connected via super-peers only. Due to this, the information flow between clusters in the SNBA protocol is only supported by super-peers. Furthermore, super-peers in the SNBA protocol group peers based on their geographical location to increase the number of connections between nodes which are close in the network. However, a long-link distance might exist between any two peers even though they are in the same geographical location. The node selection approach used by SNBA protocol is not random. Instead, the node is forced to connect to the list of nodes that was supplied by the super-peer that the node connects to. From a security point of view, the level of security awareness in the SNBA protocol can be improved if more links between clusters are maintained as well as the random process of peer selection. This improves the network resistance against partitioning attacks as well as eclipse attacks. What is meant by an eclipse attack is a scenario where an attacker creates an artificial environment around a target node so that the target node can be manipulated into performing an incorrect action. Isolation of the target nodes in this way from legitimate neighbours can be used to cause the target to produce illegitimate transaction confirmations.

The limitations of the SNBA protocol motivate the development of a new protocol that overcomes the lack of connection channels between clusters. This new protocol also considers the random selection of peers based on the Internet distance rather than the geographical location. Specifically, MNBC relies on several nodes, known as master nodes, to achieve fully connected clusters based on Internet proximity and random peer selection, where information can be exchanged between clusters via master nodes as well as normal nodes. The MNBC protocol is inspired by the Master node technology that was originally adopted in [41]. Master nodes in Darkcoin were responsible for propagating the network information to the majority of nodes. This was done without taking into account whether these nodes were close. Selecting master

Algorithm 3: Master node score calculation.

```
Let M as: Master nodes set in the network
Let z as : Best master node score to achieve
while M ≠ 0 do
    for master node in M do
        n ← masternode.CalculateScore()
        if n > z then
            z = n
            winning − node ← masternode
        end
    end
end
```

nodes in Darkcoin did not require conditions to be fulfilled to preserve security. Master nodes in the MNBC protocol connect to other nodes based on a proximity criteria. Master nodes in the MNBC protocol are selected by applying a selection phase that requires several conditions to be fulfilled to cover the role of master nodes.

Clusters in the MNBC protocol are fully connected via master nodes. Typically, this improves information propagation and security awareness. Clusters are also connected by several nodes, known as edge nodes, that represent the closest nodes belonging to different clusters. Master nodes are normally Bitcoin full nodes that can offer a level of additional functions, such as (1) creating a set of clusters in the Bitcoin network, and (3) supporting a propagation scenario, in which messages are propagated to a list of all of the known master nodes across the network as well as nodes that belong to the master nodes cluster. In addition, information can also be propagated to outside a cluster by edge nodes that are connected to other nodes in different clusters.

5.4.1. Master Node Selection

Master node selection is based on a set of rules and conditions that should be fulfilled by any node willing to take-on the role of a master node in the network. Achieving a score, which is calculated based on how much each node burns bitcoins and how long a node has been online, is required. The main advantage is that impersonation of a master node by a malicious node is challenging. This score helps to elect master nodes that are better suited to that role. To encourage nodes to compete to win the master node's role, a reward is given to a master node when it propagates a valid transaction and behaves honestly. This process is described in [42]. When a node achieves the best score, the node is elected to be a master node. This is described in Algorithm 3. When a peer wants to occupy the role of the master node, the peer invites other peers that connect to it by propagating two types of messages: a $masterINV$ message and an $AcceptINV$ message. Consider a node m that decides to be a master node and a peer p that receives a $masterINV$ message from m. When it receives the $masterINV$ message, the node p accepts m's invitation if it finds the node m to be closer in the Internet and it has a bigger weight than the master node that p is connected to. Node p decides whether m is close in the Internet by calculating the Internet distance based on ping latencies. This is the same methodology that is described in Section 5.2.1 to measure the Internet distance. Node p accepts m's invitation by sending an $AcceptINV$ message. Node p keeps forwarding the $masterINV$ to all its connected nodes, which propagates the $masterINV$ message further.

5.4.2. MNBC Cluster Maintenance

The second phase of the MNBC protocol is a cluster maintenance protocol. To increase the network's resistance to an eclipse attack or a partition attack, peer selection in MNBC preserves the idea of random selections of peers, which is important in the Bitcoin network. Peers in MNBC protocol select other peers based on a combination of factors, such as physical proximity (link latency) and random selection. Let $R = \{n_1, n_2, ..., n_{|R|}\}$ be a set of peers in the Bitcoin network, where $|R|$ is the total number of peers. Let $M = \{m_1, m_2, ..., m_{|M|}\}$ be a set of master nodes, where $|M|$ is the number of master nodes and $M \subseteq R$. Let $Mp_l = \{m_l, b_1, b_2, ...\}$, where the cluster indexes are $l = 1, 2, ..., |M|$ and let Mp_l be a set of peers in the lth cluster. Therefore, we have $Mp_l \subseteq R$ and $R = Mp_1 \cup Mp_2 \cup ... \cup Mp_{|M|}$. When a node z wants to join the Bitcoin network, it first learns about the available master nodes by contacting an arbitrary node t which it has already learned from the DNS service. The node t responds with a list of the master nodes it knows about in the network. When a node z wants to join the Bitcoin network, it first selects a master node m_i such that $\forall m_j \in M, \text{latency}(z, m_i) \leq \text{latency}(z, m_j)$. The node z sends a $JoiningRequest$ message to the selected master node. Note that the distance is also calculated based on the link latency (cf. Section 5.1.1).

Clusters are fully connected by their edge nodes and master nodes with the aim of improving the security and performance of the MNBC protocol. Edge nodes are selected between every pair of clusters. They are selected to be the closest pair of nodes in the Internet that belong to two clusters and are selected using the same strategy of border node selection that is used by the LBC protocol in Section 5.1. The one difference is that the distance between the two nodes is a measure of the link latency.

6. Performance Evaluation

Information propagation delay is used as the performance metric in our evaluation of the proposed protocols. Estimates of the reductions achieved in the transaction propagation delay may be generalised to other forms of information dissemination in the Bitcoin network and so we focus on information propagation delay measurement. We develop several simulations based on an event-based simulator that was introduced in [27]. This simulator, and the parameters which guide its operation, are described first.

These evaluations look to determine the gains achieved by the different delay reduction hypotheses investigated in this paper. Headings for these hypotheses and our conclusions are listed below to outline the structure of the rest of this section.

(1) **Proximity Aware Topology Formation:** Protocols which consider proximity awareness, reduce the propagation delay variance compared to the Bitcoin protocol. This reduction is significant when the number of nodes increases from 7 to 10. This performance gain is explained by the fact that the Bitcoin protocol does not consider the structure of the topology, whereas all of the proposed protocols look to create connections between nodes using a set of properties which are based on node proximity.

(2) **Super-Peers vs Master Nodes:** The SNBA protocol seeks to use super-peers to reduce the numbers of hops between peers. The SNBA protocol exhibits the largest delay variation out of all protocols contributed in this paper for node counts greater than 7. This increase in delay is explained by the fact that information flow is only achieved by super-peers in the SNBA protocol. The extra connection channels introduced by the MNBA protocol achieve faster information propagation.

(3) **Geographical Distance vs Latency:** Protocols that attempt to form an overlay based on link latencies yield smaller information propagation delays. The PTBC protocol has a smaller delay variation compared to the LBC protocol. This is explained by the fact that geographically close nodes might in fact be far away when this distance is measured using Internet distance.

(4) **Latency-based Proximity Measurement and Increased Connectivity:** Protocols that use the physical Internet distance (latency) as a measure of proximity for both edge node formation and cluster formation achieve the smallest information propagation delays. The MNBC protocol achieves the best improvement in propagation delay out of all protocols evaluated in this paper, because it benefits from the use of extra channels.

(5) **Consistency of the public ledger:** The larger the network, the greater the resistance to partition attacks. The Bitcoin protocol achieves the largest minimum vertex cut, which is a measure of its resistance to partition attacks; however attackers would need significant computational resources to split the network topologies generated by the protocols proposed in this paper.

6.1. Simulation Structure

We use a lightweight, event-based simulator which is abstracted from cryptography aspects of Bitcoin to interrogate the hypotheses formulated in this paper. Its focus is on the Bitcoin overlay network and the transaction round-trip delay. The simulation model is developed in Java for object oriented structure and modularity. It implements a discrete event simulation environment, where the behavior of the Bitcoin client is modelled as an ordered sequence of well-defined events. These events, which take place at discrete points in simulation time, correspond to changes in the systems state. Two notions of time are taken into account, simulation time and run time. Simulation time reflects the virtual time or logical time in the simulation world. The run time refers to the time that is consumed by a processor that is contending with a particular thread. Simulation time has a direct impact on how the simulation events are organised and on how accurate results are gained. When an event E_1, is executed by a thread A, E_1 should schedule another event $E_{1,Return}$, which represents a successful return from E_1. The successful return $E_{1,Return}$, must be scheduled at a specific point in the simulation time which is calculated after adding an appropriate delay. This delay is collected from the time distributions that are passed to the model. Details about how these distributions are approximated are given in Section 6.1.2. During the time that elapses betweenE_1, and $E_{1,Return}$, the simulator can execute any number of events for the same or another client. The simulator is based on a priority queue that includes all events which are ranked

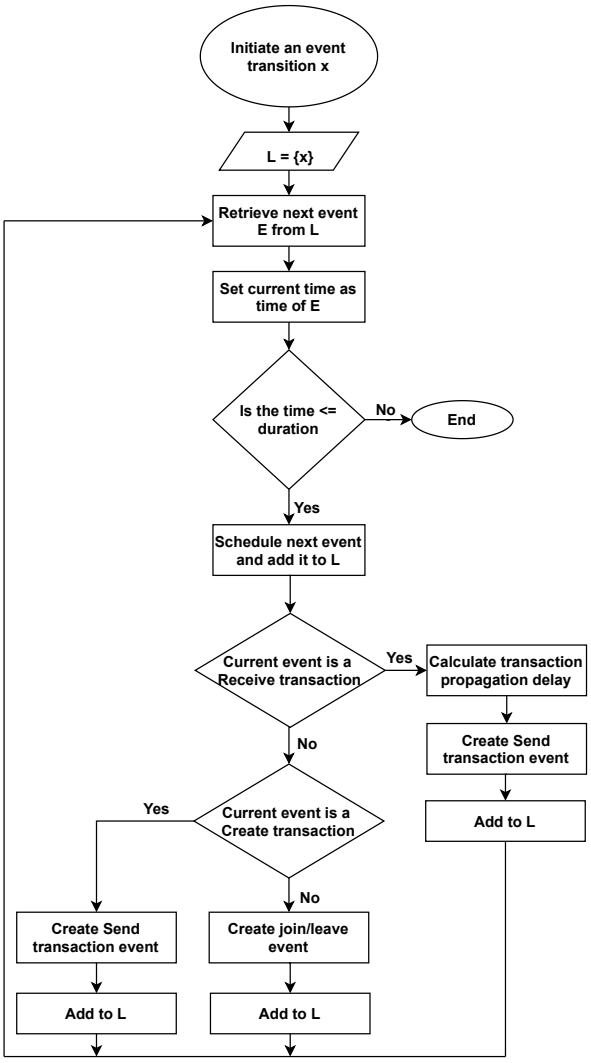


Figure 4: Bitcoin simulator structure is based on a priority queue.

based on its Expected Time of Schedule (ETS) in Figure 4. The ETS is calculated for each event based on time distributions which are measured in the real Bitcoin network and passed as an input to the simulator. Based on the ETS, the first event is scheduled and removed from the queue. An individual node's behavior such as joining or leaving the network, creating transactions and forwarding transaction, is implemented by inheritance from given generic java classes.

Different measurements of the most influential parameters that have a direct impact on a clients behavior and information propagation in the real Bitcoin network (cf. [27]) are attached to the developed simulator to ensure that information propagation is well modelled. These measurements include the number of reachable nodes, link latencies, and the lengths of the sessions nodes participate in. We now describe how these measurements are made.

6.1.1. Session Length

The session lengths in the real Bitcoin network were calculated by implementing a Bitcoin client which was used to crawl the entire Bitcoin network by establishing connections to all reachable peers in the network. Periodically, the client attempted to discover Bitcoin network peers with the aim of maintaining connections to the majority of them. This was done by sending an *Addr* message to the client's neighbours. By getting a list of IP addresses from its neighbours, the client started connecting to each of the IP addresses in the received list of IP addresses. As crawlers require time to capture a complete snapshot that accurately reflects the topological properties and dynamics of unstructured P2P networks [43], the developed client crawled the Bitcoin network for one week. During this week, snapshots of IP addresses of reachable peers were published every 3 hours to avoid a situation where the captured snapshots became more distorted due to the gap between consecutive snapshots. By using data that was gathered by running the developed crawler for one week, points in time in which peers left or joined the network were available. An example of an incidence that might happen during snapshot gathering, is losing the network connectivity or that the observation software crashes. This results in a gap in the data captured during the overall gathering time. During this gap, important data maybe missing. To overcome this challenge, measurements were composed from a series of snapshots that were maintained by the crawler. Each snapshot included the start time of the crawl. Therefore, it was possible to identify whether or not some data was missing by examining the series of times in which the captured snapshots started. By following this data verification procedure, we determined that significant gaps in the collected data were not experienced, and thus the data was usable for our experiments.

The distributions of session lengths in the real Bitcoin network are shown in Figure 5. Even though the distributions of session lengths reveal a considerable churn in the data, 1400 peers did not leave the network during the observation time. We conclude that the stability of the network fluctuates. This might lead to substantial changes in the topology during experimentation.

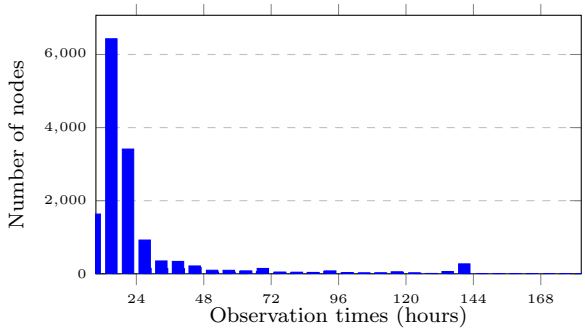


Figure 5: Session lengths of peers in the Bitcoin network.

6.1.2. Link Latencies

Measurements of the network latency between peers in the Internet play a significant role in the development of any P2P network model as these measurements control the accuracy of conclusions produced by network models [44]. One focus of this research is on information propagation latency in the Bitcoin network. The accurate measurement of link latencies between peers is a fundamental requirement. Measurements of link latencies between peers were collected by setting up a Bitcoin client that crawled the entire Bitcoin network. The developed client utilised a list of IP addresses to connect to the majority of peers in the network. Also, the client considered the advantage of ping messages to measure the round trip latency between the discovered peers and developed client. The client attempted to maintain connections to several peers. After that, the client began an iterative process of sending ping messages to each peer of the connected peers. The link latency between the client and a particular connected peer was calculated when the client heard back from the peer (reception of a pong message). The link latency was measured by calculating the time difference between sending a ping message to the peer, and receiving a pong message back. To maintain large scale and distributed measurements, the client periodically scanned the network and applied the same scenario of measuring the link latencies.

The distributions of latencies between a client that was located in Portsmouth in the UK, and peers in the real Bitcoin network are shown in Figure 6. These distributions were collected by running the crawler, which was connected to approximately 7000 network peers and observed a total of 27000 ping/pong messages. The distribution of latencies reveals that around 75% of the collected latencies were below 800ms, while 25% of distributions are over 800ms. Some of them lasted up to 2500ms. Note that these empirical distributions indicate the latency between the crawler and the other network peers.

Although the link latency between two peers relies on the location of the host from which the latency is mea-

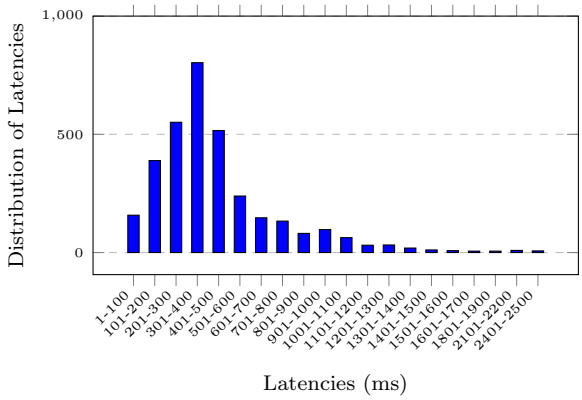


Figure 6: Link latency values between the measurement node (located in Portsmouth, UK) and other Bitcoin peers.

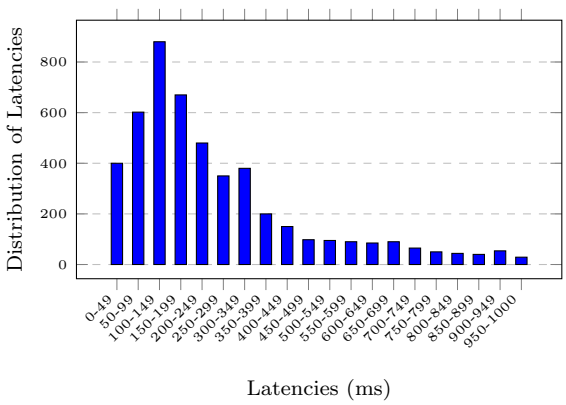


Figure 7: Link latencies between the measurement node (located in Los Angeles) and other peers in the Bitcoin network.

sured, a similar distribution of latencies over the entire set of peers might be obtained from two different hosts, where each host is in a different location. To investigate this, the crawler was run in a different location. Figure 7 shows the distribution of the round-trip latencies between peers that were collected by running the crawler in Los Angeles. The shape of the distribution in Figure 7 is similar, up to a dilation factor, to the previous distribution in Figure 6. We conclude that inputting the obtained link latencies distributions to the developed simulation model gives a reasonable estimate of the time delay taken by a transaction to reach different peers in the network.

6.1.3. Size of the Bitcoin Network

As the developed model simulates information propagation in the Bitcoin network, the size of the network matters because the number of nodes has a direct impact on the range of propagation delays that will be observed. The size of the Bitcoin network was measured using the same crawler in the Section 6.1.1. The crawler was able to measure the size of the network by discovering the available IP addresses in the network and by trying to connect to them. The size of the Bitcoin network was observed to be approximately 8000 nodes, because the crawler learned 313676 IP addresses but was only able to connect to 7834 peers.

6.1.4. Model Validation

The developed model was validated by comparison with real Bitcoin network transaction propagation delays. Several aspects of the real Bitcoin network such as client behavior, processing delay, and network topology have a direct impact on transaction propagation delay. In previous research, transaction propagation delay measurements were presented in the real Bitcoin network based on the propagation of *INV* messages. The transaction propagation delay was measured in [2; 44] by setting up a Bitcoin client that kept listening for *INV* messages. The client calculated the time difference between the first reception of an *INV* message and subsequent receptions of *INV* messages, where all of the received *INV* messages belonged to the same announcement of a transaction. The collected measurements did not indicate when transactions were received, and so these measurements did not represent the actual transaction propagation delay. We measure transaction propagation delay in the real Bitcoin network in a way that the transaction propagation delay is indicated when peers receive transactions.

To measure how fast a transaction was propagated in the Bitcoin network, the Bitcoin protocol was implemented and used to establish connections to many points in the network, to measure the time that a transaction took to reach each point. A measuring node was implemented, which behaved exactly like a normal node with the following functionalities. The measuring node connected to 10 reachable peers in the Bitcoin network. It was capable of creating a valid transaction and propagating it to one peer of its connections, and then tracking the transaction to record the time each peer of its connections announced the transaction. For example, suppose the client c, in Figure 8, has connections with nodes $1, 2, 3, \ldots, n$, the node c propagated a transaction at time T, and it was received by the nodes it was connected to at different times, $T_1, T_2, T_3, ..., T_n$, as illustrated in Figure 8. The time differences between the initial transaction propagation and subsequent receptions of the transaction by connected nodes was denoted, $\Delta t_{c,1}, ..., \Delta t_{c,n}$, where:

$$\Delta t_{c,n} = T_n - T_c \tag{9}$$

The transaction reception times were ordered from largest to smallest, $T_n > T_{n-1} >, ..., T_2, T_1$. The timing information was collected by running the experiment 1000 times as oneoff style events, so that networking delays, for example, were averaged out. At each run, the measuring node randomly connected to 10 nodes. The number of connected nodes represented the sequence of the random nodes that the measuring node connected with at each run. In terms of measuring the transaction propagation delay in the simulation world, the aforementioned measuring method in the real Bitcoin network was used in the

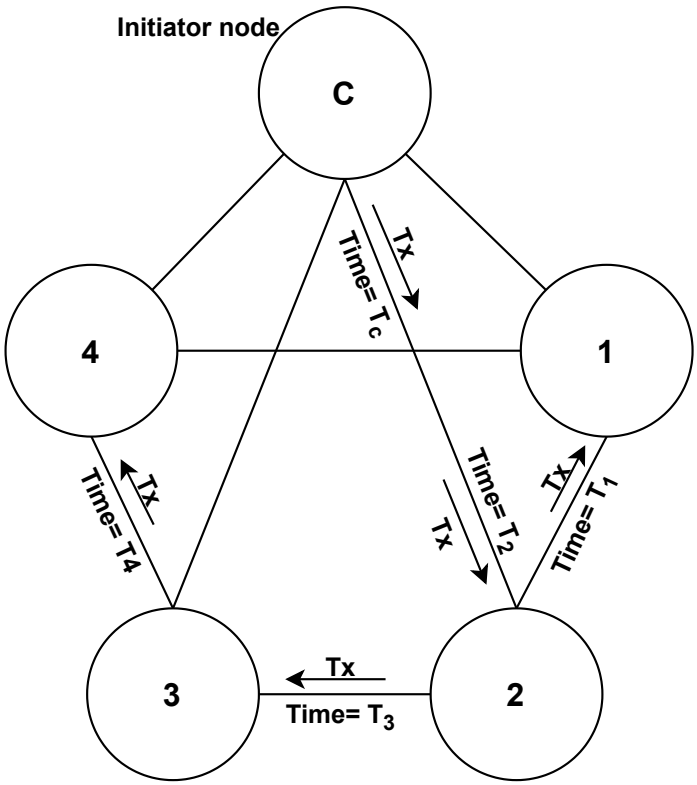


Figure 8: Illustration of propagation experimental setup.

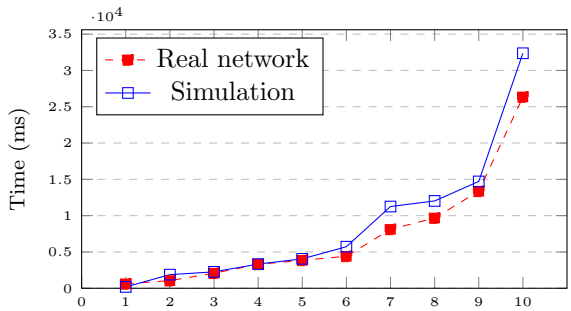


Figure 9: Comparison of the distributions of $\Delta t_{c,n}$ measured in the real Bitcoin network and via the simulation.

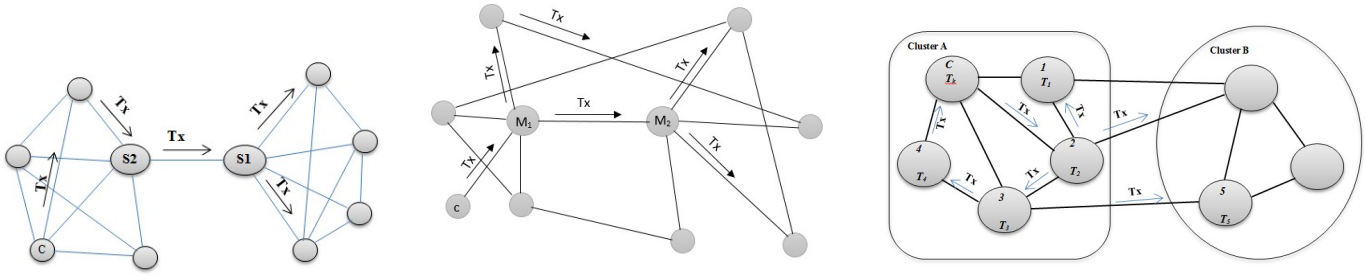
simulation. By doing this, the simulation model was validated by comparing the propagation delay measurements that were collected from the Bitcoin simulator to the measurements that were collected from the real Bitcoin network. As the measurements are taken when peers received transactions, the distribution of these measured time differences, $\Delta t_{c,1}$, represents the real transaction propagation delay. The average distributions of $\Delta t_{c,n}$ for the real Bitcoin network and the simulated network are shown in Figure 9.

Results demonstrate that during the first 13 seconds the transaction was propagated fast, and 6 nodes received it with low variance of delays. It should be noted that the transaction propagation delays increased dramatically as the number of nodes increased to 9 and 10 nodes, which means that the transaction was received by these nodes with a significantly larger delay variance. These results reveal that the propagation delay increases with the number of nodes. This is because the total duration of subsequent announcements of the transaction by the remaining nodes increases as the numbers of connected nodes increases. This happens due to each node being connected to large segments of the network, while the connected nodes were not geographically localised. We conclude that the simulation model closely approximates the behaviour of the real Bitcoin network.

6.2. Experimental Setup

The experimental setup that is used to evaluate the performance of the LBC, PTBC, SNBA, and MNBC protocols is now explained. We consider four different simulation scenarios, one for each of the proposed protocols. In each simulation, the size of the network matters as the evaluation is based on the transaction propagation delay. The size of the network in each simulation matches the size of the real Bitcoin network which was measured in our previous work [22]. Each node in the overlay is allowed to discover new nodes every 100ms. Several proximity based clusters are generated at times which depend on the protocol under consideration. As the performance evaluation is based on measuring how fast a transaction is propagated in the network after applying the clustering approaches, the transaction propagation delay in each approach is measured using the same methodology that was used in [22], to measure the transaction propagation delay in the real and simulated Bitcoin network. Figure 10(a) gives an illustrative example of how the simulation experiment works for the SNBA protocol, while Figure 10(b) illustrates the simulation setup for the MNBC protocol. Figure 10(c) shows an example of the simulation setup for PTBC and LBC.

Before applying the proximity cluster generation algorithms of the proposed techniques, it is assumed that the network nodes belong to one cluster. Based on the PTBC and the LBC protocol, proximity based clusters are generated at times which depend on the ping latency threshold in the PTBC protocol, and a geographical distance threshold in the LBC protocol. For the PTBC protocol, two nodes are close to each other if the measured latency is lower than the suggested distance threshold, L_{th} = 25ms. In the LBC protocol, if the geographical distance between two nodes is lower than the suggested threshold, D_{th} = 50km, then those nodes are close to each other. Regarding the SNBA protocol, super-peers are selected by running the super-peer selection algorithm that is described in Section 5.3.1. After that, every super-peer of the selected super-peers constructs a cluster by recruiting geographically close nodes. Similarly, the master node selection algorithm in the MNBC protocol (in Section 5.1.1), is launched at a certain point in the experiment time to select master nodes. The selected master nodes group peers



(a) SNBA simulation setup. (b) MNBC simulation setup. (c) LBC & PTBC simulation setup.

Figure 10: Performance evaluation: experiment setup.

that are close in the physical Internet. The link distance between nodes is modelled using the real-world measurements in Section 6.1.2.

Once the proximity based clusters have been formed in each simulation scenario, normal Bitcoin simulator events are launched. For each of the proposed protocols, a measurement node, c, is implemented, which creates a valid transaction, T_x, and sends it to one node of its connected nodes. It then tracks the transaction to record the time each node of its connections announces the transaction. Suppose the client c, has proximity based connections with nodes $1, 2, 3, \ldots, n$, the client c propagates a transaction at time, T, and it is received by its connected nodes at different times $(T_1, T_2, T_3, ..., T_n)$. The time differences between the transaction transmission and the subsequent reception times for the transactions at connected nodes are calculated, $(\Delta t_{c,1}, ..., \Delta t_{c,n})$. The latency value is determined by taking an average of measurements from approximately 1000 experimental runs to increase the accuracy of the collected latencies, which might be affected due to data corruption and loss of connection.

6.3. Results and Analysis: Propagation Delay

Simulation results show that the proposed protocols offer an improvement in propagation delay compared to the Bitcoin protocol. Figure 11 compares the distributions of $\Delta t_{c,n}$ for the simulated Bitcoin protocol and the proposed protocols SNBA, LBC, PTBC, and MNBC.

The number of connected nodes represents the sequence of the random nodes that the measuring node connects with at each run. In all protocols, the distributions of delays increase gradually as the simulation time moves forward and the number of connected nodes increases. It should be noted that the transaction propagation delays are larger in the simulated Bitcoin protocol over nodes $7, 8, 9$ and 10. The observed delays for the SNBA, LBC, PTBC, and MNBC protocols are much smaller for the same nodes sequences. This means that the transaction was received by the connected nodes in the SNBA, LBC, PTBC, and MNBC protocols with lower variances of delays compared to the simulated Bitcoin protocol. The

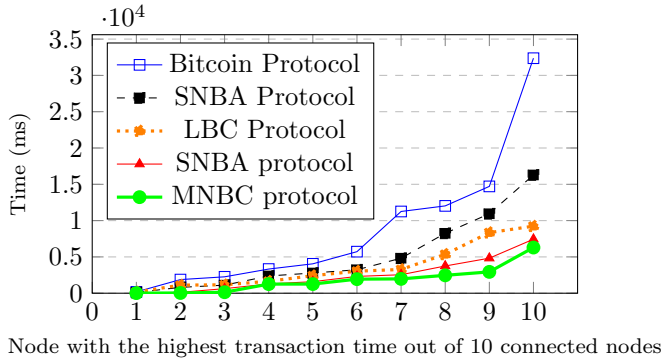


Figure 11: Comparison of the empirical distribution of $\Delta t_{c,n}$ measured in the simulated Bitcoin protocol with the empirical distribution of $\Delta t_{c,n}$ measured for the PTBC, LBC, SNBA, and MNBC protocols. The thresholds used are: $L_{th} = 25$ms for the PTBC protocol and $D_{th} = 50$km for the LBC protocol.

reduction of the transaction propagation time variances achieved by the proposed protocols occurs because the Bitcoin network layout, where nodes connect to other nodes without taking advantage of any proximity correlations, results in a high communication link cost, which is measured here by the distance between the nodes. Consequently, the average delay to get transactions delivered is also increased. This has direct implications on the consistency of the public ledger, whose consistency becomes vulnerable when delays are large. Contrary to what was previously thought, this result demonstrates that reconstructing a Bitcoin network topology, so that proximity is considered, yields faster transmission times.

We now compare the PTBC, LBC, SNBA and MNBC protocols. In Figure 11, the proposed protocols show similar delay variances over nodes in the range, $1, 2, \ldots 6$. From node 7, variances of delays in the SNBA protocol started climbing steadily and reached a peak at for 10 nodes, where the recorded transaction propagation delay was nearly 18000ms. In contrast, the trend of the variances of delays for the LBC protocol flattened off at a level of 2000ms for 6 nodes but then reached a peak of 2500ms for 7 nodes. After that, it quickly increased and reached 9000ms for 10 nodes. On the other hand, the variances of

delays were improved in the PTBC protocol over the LBC and SNBA protocol, especially for 8, 9 and 10 nodes. Regarding the MNBC protocol, it achieved faster transaction propagation delays regardless of the gradually increasing delays when the number of nodes increased.

The most likely cause of the higher variances of delays in the SNBA protocol is the fact that the information flow between clusters in the SNBA protocol can only be achieved by supers peers. This causes a shortage of transmission channels between clusters which results in inefficient information distribution over the network. The lack of connections between clusters in the SNBA protocol was tackled in the MNBC protocol by considering the edge nodes technology, which added an extra connection channel between clusters. Faster information propagation was achieved by the MNBC protocol compared to the SNBA protocol. Even though the LBC protocol delivered faster transaction propagation compared to the SNBA protocol, the lowest variances of delays were achieved by the PTBC protocol over the LBC and SNBA protocol. It is possible that the cause of the lower variances of delays in the PTBC protocol compared to the LBC protocol, is that two geographically close nodes may actually be quite far from each other in the physical Internet. Somewhat counter-intuitively, physical distance may lead to smaller delays. This leads to a different conclusion, proximity awareness in the physical Internet improves delivery latencies with a higher probability because transactions may traverse fewer hops and use shorter links. However, comparison of the MNBC protocol's results with those of other the proposed protocols confirms that the MNBC protocol achieves the best reduction of delay for information propagation. A possible explanation for this improvement is that it adopts the physical Internet distance as a proximity metric in both edge nodes technology and clusters creation. Furthermore, the MNBC protocol provides extra transformation channels by which faster information distribution is achieved.

As the PTBC and LBC protocols are based on a suggested threshold, we investigated the PTBC and LBC protocols' performance as a function of the latency and geographical distance thresholds L_{th} and D_{th} respectively to determine which threshold yielded the biggest reduction in information propagation delay. In the PTBC protocol, the comparison among three variances of delays was undertaken using three different latency thresholds: 30ms, 60ms, and 90ms. The comparison for the LBC protocol used the geographical thresholds 20km, 50km, and 100km. The results shown in Figure 12 reveal that the lower the latency of the distance threshold for PTBC protocol, the smaller the resulting variance is for delays.

Based on these results, there is a negative correlation between the propagation delay and the latency threshold, as the total duration of subsequent announcements of the transaction by the remaining nodes increases with a larger latency threshold. The key reason for variances of delays declining when the threshold value is reduced is that

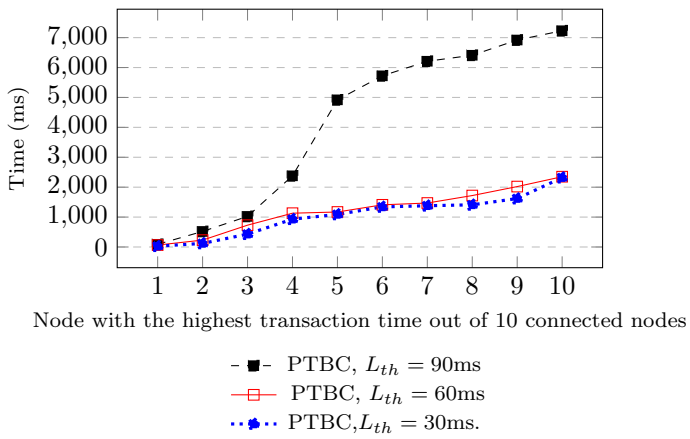


Figure 12: Distributions of $\Delta t_{c,n}$ measured for the PTBC protocol with three thresholds ($L_t = 30, 60, 90$ms)

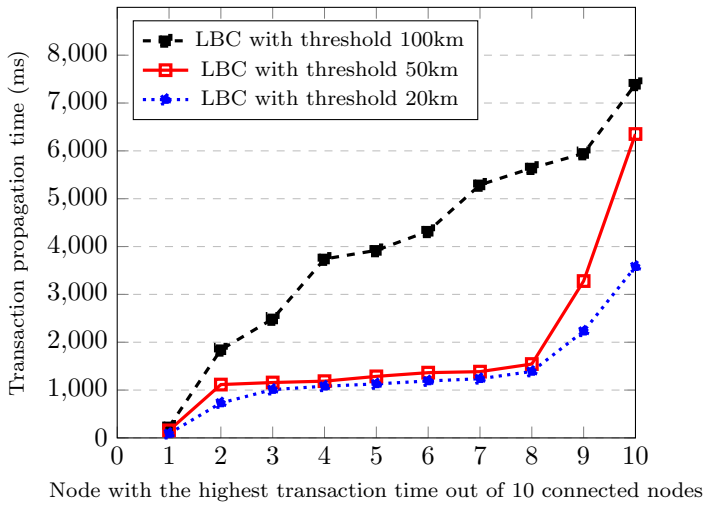


Figure 13: Comparison of the distribution of $\Delta t_{c,n}$ as measured for LBC with three thresholds ($D_{th} = 20, 50, 100$km).

the number of nodes at each cluster is minimised due to the limited coverage of the physical topology. Similarly, reducing the geographical distance threshold in LBC, as illustrated in Figure 13, yields smaller variances of delays. The most likely cause for the reduction in variances of delays when the threshold value is minimised is that the limited coverage of geographical location results in fewer nodes being members of each cluster, which results in the hop-count for the transaction being reduced.

7. Security Evaluation

We evaluate the potential for partition attacks occurring on the proposed protocols as well as on Bitcoin. Partition attacks split the network into a number of sub-partitions and block the data flow among them [45]. In the Bitcoin network, partition attacks affect the main system functions, which in turn, negatively impact user trust. We adopt an attack model, which consists of three steps:

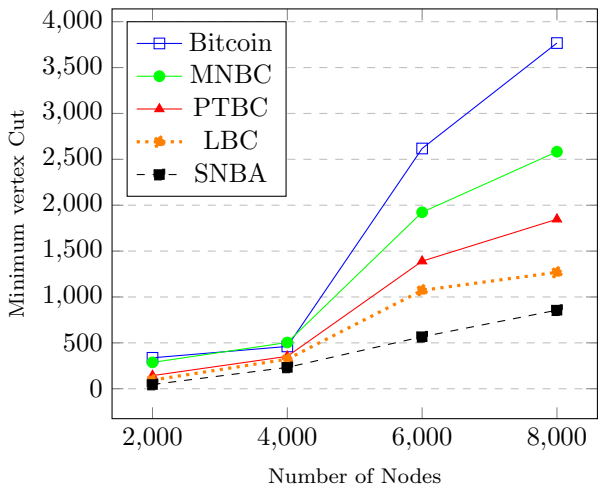


Figure 14: Number of non-compromised peers on the minimum vertex cut. Bitcoin achieves the largest minimum vertex cut.

(1) The attacker injects a number of compromised nodes into the P2P Bitcoin network. Each compromised node announces the IP of the other compromised nodes so that the probability of connecting to non-compromised nodes is increased.
(2) Once the connection between compromised and non-compromised nodes is complete, the attacker predicts the network topology. For example, this can be accomplished using the probabilistic techniques describe in [24], which allow the attacker to expose the topology by sending marker addresses and observing the flow of these addresses.
(3) At this stage, attackers detect the *minimum vertex cut*, that is the least number of non-compromised nodes whose removal partitions the network into 2-parts or more [46].

We use the *minimum vertex cut* to evaluate the cost of performing partition attacks in Bitcoin networks. Two platforms were utilised to evaluate partition attack, these are: (i) the developed simulator (Section 6.1) and (ii) the Metis toolkit [47] for graph partitioning. The application of Metis results in balanced partitions [48]. In this paper, we assume that the attacker is aiming to gain a number of well-sized partitions. We do not require that the partitions are balanced. We verify the security performance using 1000 runs for each scenario.

7.1. Results and Analysis: Security

We analyse the experimental results produced using the simulator described in Section 6.1. Figure 14 illustrates the performance of the PTBC, LBC, SNBA, and MNBC protocols in response to attacks that were conducted on a real-world Bitcoin network.

Four networks of size 2000, 4000, 6000 and 8000 nodes were constructed for these experiments. In small-scale networks –the case where the number of nodes was either 2000 or 4000 nodes– we observed that the number of the non-compromised nodes remained less than 500 after the partition attack had been launched. For large-scale networks –the number of nodes was either 6000 or 8000 nodes– we observed that networks exhibited more resistance to attacks. In summary, the larger the network, the greater the resistance to partition attacks. Crucially, we report that the Bitcoin protocol achieves the largest minimum vertex cut out of all evaluated protocols. The SNBA protocol has the minimum vertex cut. Both, the PTBC and LBC protocols have low resistance to the attack; the minimum vertex cut is below 2000, even in large scale scenarios. We also report that MNBC protocol has a higher resistance to attack compared to the LBC and PTBC protocols, where the minimum vertex cut is approximately 2500 in large scale scenarios. However, this is approximately 1000 smaller than the the minimum vertex cut for the Bitcoin protocol for networks of the same size. In conclusion, the results show that the Bitcoin protocol has the highest minimum vertex cut. This makes it the most resistant to partition attacks compared to the other protocols. The SNBA protocol has the lowest minimum vertex cut, which makes the launching of partition attacks easier. Even though the proposed protocols have a lower minimum vertex cut compared to the Bitcoin protocol, they still require a very large number of non-compromised nodes to perform the cut. This form of attack requires massive computational resources. As expected, clusters in the MNBC protocol, which are fully connected via master nodes and edge nodes, and clusters in the LBC and PTBC protocols, which are connected via border nodes, have fewer numbers of non-compromised nodes in the minimum vertex cut. However, clusters formed by the SNBA protocol, which are connected via super-peers, result in the number of nodes in the area of the minimum vertex cut decreasing.

To determine the relationship between the resistance to partition attacks and the session length of the attacker, we run another experiment and evaluate Bitcoin and the proposed protocols. The result in Figure 15 shows the direct impact of the attacker's session length on launching the attack successfully.

In this experiment, the attack is launched over 24 hours. We observe that the number of nodes in the minimum vertex cut decreases for all protocols with the passage of the experiment time. Note that the number of minimum vertex cut nodes dropped from 3700 to 1500 in the real Bitcoin network scenario in Figure 15. The minimum vertex cut nodes dropped from 2500 to 1150, from 1800 to 930, from 1200 to 430 and from 850 to 290 for the MNBC, PTBC, LBC and SNBA protocols respectively. An important finding that emerges from this experiment is that the simulated Bitcoin network outperformed the proposed protocols in terms of the resistance to partition attacks. The more patience an attacker (with a high number of peers) has, the better the attacker's chances of splitting the network are. To find the correlation between the number of clusters and the difficulty of successfully carrying out a partitioning attack, the results of another experiment are shown in Figure 16. These results reveal that

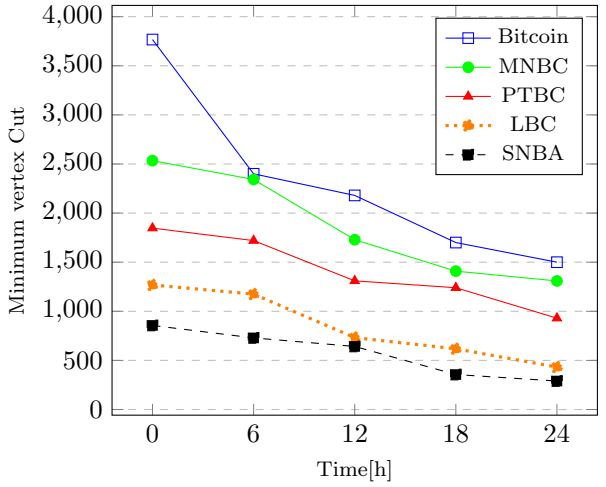


Figure 15: Number of non-attacker peers on the minimum vertex cut during an attack with 7000 non-compromised peers. This experiment is parameterised as in the real-world network an the attackers session length is 6 hours.

the number of clusters is directly proportional to the minimum vertex cut nodes. This means that more proximity clusters would result in increasing difficulty in achieving a partition attack.

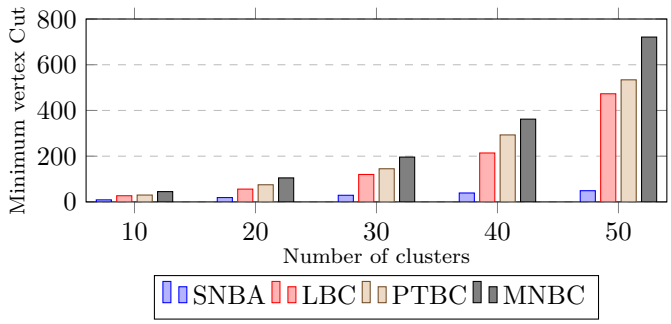


Figure 16: Number of non-compromised peers on the minimum vertex cut based on number of partitions.

8. Conclusion and Future Work

In this paper, we proposed several network clustering methods which aim to decrease the chances of performing a successful double spending attack through alleviating the information propagation delay of Bitcoin. Furthermore, we critically analysed the performance and security impact of the proposed clustering methods. Specifically, we evaluated the performance and security properties of these clustering approaches in terms of (1) their transaction propagation speeds and (2) their ability to resist partition attacks. The results show that significant improvements in the transaction propagation delay over the Bitcoin network protocol are possible. The MNBC protocol achieves the lowest variance of delays over the PTBC, LBC, and SNBA protocols. Experiments with different latency thresholds in the PTBC protocol, as well as different geographical distance threshold values in the LBC protocol were conducted to identify the distance threshold that gave the best improvement in the transaction propagation delay. Reducing the latency and geographical distance thresholds improved the transaction propagation delay. Security evaluations revealed that the Bitcoin network is more resistant to attackers than the proposed protocols. Maximising the number of clusters in each approach improved the network's ability to resist partition attacks. Attackers would need significant resources to split the network generated by the proposed protocols, especially large networks. These findings suggest the proposed protocols are a good starting-point for future research investigations into transaction propagation delay optimisation. We propose the following conclusions should be adopted as avenues for exploration in future work:

- Bitcoin does not consider the structure of the topology. Protocols which consider proximity awareness, reduce the propagation delay variance compared to the Bitcoin protocol.
- Super-peers may be used to to reduce the numbers of hops between peers, however, in this paper the largest delay variation out of all protocols in this paper (for node counts greater than 7) was observed for the super-peer approach. In comparison, the extra connection channels in the MNBA protocol helped it to achieve faster information propagation.
- In terms of adopting a geographical or Internet distance in protocol design, protocols that form an overlay based on link latencies yield smaller information propagation delays.
- Taking this one step further, protocols that use the physical Internet distance (latency) as a measure of proximity for both edge node formation and cluster formation achieve the smallest information propagation delays.
- Robustness to partition attacks and double-spending attacks are achieved by different mechanisms. The larger the network, the greater the resistance to partition attacks. The faster the information propagation time, the greater the resistance to double-spending attacks. The Bitcoin protocol achieves the largest minimum vertex cut, which is a measure of its resistance to partition attacks, however, attackers would need significant computational resources to split the network topologies generated by the protocols proposed in this paper.

In summary, numerical results demonstrate how the extensions run and also their impact on optimising the transaction propagation delay.

Acknowledgement

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under the grant number 15/SIRG/3459.

[1] P. Nerurkar, D. Patel, Y. Busnel, R. Ludinard, S. Kumari, M. K. Khan, Dissecting bitcoin blockchain: Empirical analysis of bitcoin network (2009–2020), Journal of Network and Computer Applications 177 (2021) 102940.

[2] C. Decker, R. Wattenhofer, Information propagation in the bitcoin network, in: Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on, IEEE, 2013, pp. 1–10.

[3] G. Owenson, M. Adda, et al., Proximity awareness approach to enhance propagation delay on the bitcoin peer-to-peer network, in: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), IEEE, 2017, pp. 2411–2416.

[4] M. Conti, C. Lal, S. Ruj, et al., A survey on security and privacy issues of bitcoin, arXiv preprint arXiv:1706.00916.

[5] M. Fadhil, G. Owenson, M. Adda, Locality based approach to improve propagation delay on the bitcoin peer-to-peer network, in: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), IEEE, 2017, pp. 556–559.

[6] C. Stathakopoulou, C. Decker, R. Wattenhofer, A faster bitcoin network, Tech. rep., ETH, Zurich,. Semester Thesis.

[7] B. Bitcoin Wiki, Block (2008). URL "http://www.bitcoin.org/bitcoin.pdf"

[8] Y. Sompolinsky, A. Zohar, Accelerating bitcoin's transaction processing. fast money grows on trees, not chains., IACR Cryptology ePrint Archive 2013 (881).

[9] G. O. Karame, E. Androulaki, M. Roeschlin, A. Gervais, S. Čapkun, Misbehavior in bitcoin: A study of double-spending and accountability, ACM Trans. Inf. Sys. Sec. 18 (1) (2015) 2.

[10] M. Rosenfeld, Analysis of hashrate-based double spending, arXiv preprint arXiv:1402.2009.

[11] J. A. Garay, A. Kiayias, N. Leonardos, The bitcoin backbone protocol: Analysis and applications., in: EUROCRYPT (2), 2015, pp. 281–310.

[12] A. Miller, J. J. LaViola Jr, Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin, Available online: http://nakamotoinstitute. org/research/anonymous-byzantine-consensus.

[13] J. E. Pazmiño, C. K. da Silva Rodrigues, Simply dividing a bitcoin network node may reduce transaction verification time, The SIJ Transactions on Computer Networks and Communication Engineering (CNCE) 3 (2) (2015) 17–21.

[14] C. Basescu, E. Kokoris-Kogias, B. A. Ford, Low-latency blockchain consensus, Tech. rep., EPFL (2017).

[15] M. Zamani, M. Movahedi, M. Raykova, Rapidchain: Scaling blockchain via full sharding, in: Proceedings of the 2018 ACM/SIGSAC Conf. on Computer and Communications Security, 2018, pp. 931–948.

[16] M. Corallo, Compact block relay. bip 152 (2017).

[17] F. falcon, fast bitcoin backbone falcon (2015). URL "https://www.falcon-net.org".

[18] A. Biryukov, I. Pustogarov, Bitcoin over tor isn't a good idea, in: Security and Privacy (SP), 2015 IEEE Symposium on, IEEE, 2015, pp. 122–134.

[19] M. Corallo, Fibre: Fast internet bitcoin relay engine (2017).

[20] Y. Shahsavari, K. Zhang, C. Talhi, A theoretical model for block propagation analysis in bitcoin network, IEEE Transactions on Engineering Management.

[21] T. Bamert, C. Decker, L. Elsen, R. Wattenhofer, S. Welten, Have a snack, pay with bitcoins, in: IEEE P2P 2013 Proceedings, IEEE, 2013, pp. 1–5.

[22] M. Fadhil, G. Owenson, M. Adda, A bitcoin model for evaluation of clustering to improve propagation delay in bitcoin network, in: 2016 IEEE Intl Conf. on Computational Science and Engineering (CSE), IEEE, 2016, pp. 468–475.

[23] J. A. D. Donet, C. Pérez-Sola, J. Herrera-Joancomartí, The bitcoin p2p network, in: International Conference on Financial Cryptography and Data Security, Springer, 2014, pp. 87–102.

[24] A. Biryukov, D. Khovratovich, I. Pustogarov, Deanonymisation of clients in bitcoin p2p network, in: Proc. of the 2014 ACM/SIGSAC Conf. on Computer and Communications Security, ACM, 2014, pp. 15–29.

[25] E. Heilman, A. Kendler, A. Zohar, S. Goldberg, Eclipse attacks on bitcoin's peer-to-peer network., in: USENIX Security Symposium, 2015, pp. 129–144.

[26] D. Kondor, M. Pósfai, I. Csabai, G. Vattay, Do the rich get richer? an empirical analysis of the bitcoin transaction network, PloS one 9 (2) (2014) e86197.

[27] M. Fadhil, G. Owenson, M. Adda, A bitcoin model for evaluation of clustering to improve propagation delay in bitcoin network, in: 2016 IEEE intl conference on computational science and engineering (CSE) and IEEE intl conference on embedded and ubiquitous computing (EUC) and 15th intl symposium on distributed computing and applications for business engineering (DCABES), IEEE, 2016, pp. 468–475.

[28] S. Feld, M. Schönfeld, M. Werner, Analyzing the deployment of bitcoin's p2p network under an as-level perspective, Procedia Computer Science 32 (2014) 1121–1126.

[29] A. M. Antonopoulos, Mastering Bitcoin: unlocking digital cryptocurrencies, " O'Reilly Media, Inc.", 2014.

[30] M. Sallal, G. Owenson, M. Adda, Security and performance evaluation of master node protocol in the bitcoin peer-to-peer network, in: 2020 IEEE Symposium on Computers and Communications (ISCC), IEEE, 2020, pp. 1–6.

[31] G. Karame, E. Androulaki, S. Capkun, Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin., IACR Cryptology ePrint Archive 2012 (248).

[32] C. Veness, Calculate distance and bearing between two latitude/longitude points using haversine formula in javascript, Movable Type Scripts.

[33] J. D. Anderson, J. K. Campbell, J. E. Ekelund, J. Ellis, J. F. Jordan, Anomalous orbital-energy changes observed during spacecraft flybys of earth, Physical Review Letters 100 (9) (2008) 091102.

[34] Maxmind - geolite legacy downloadable databases. (2013). URL "http://dev.maxmind.com/geoip/legacy/geolite/"

[35] L. Ramaswamy, B. Gedik, L. Liu, A distributed approach to node clustering in decentralized peer-to-peer networks, IEEE Trans. on Parallel and Distributed Sys. 16 (9) (2005) 814–829.

[36] C. C. Miers, M. A. Simplicio, D. S. Gallo, T. C. Carvalho, G. Bressan, V. Souza, P. Karlsson, A. Damola, A taxonomy for locality algorithms on peer-to-peer networks, IEEE Latin America Transactions 8 (4) (2010) 323–331.

[37] A. T. Mizrak, Y. Cheng, V. Kumar, S. Savage, Structured superpeers: Leveraging heterogeneity to provide constant-time lookup, in: Internet Applications. WIAPP 2003. Proceedings. The Third IEEE Workshop on, IEEE, 2003, pp. 104–111.

[38] B. B. Yang, H. Garcia-Molina, Designing a super-peer network, in: Data Engineering, 2003. Proceedings. 19th International Conference on, IEEE, 2003, pp. 49–60.

[39] C. R. Lin, M. Gerla, Adaptive clustering for mobile wireless networks, IEEE Journal on Selected areas in Communications 15 (7) (1997) 1265–1275.

[40] M. Sallal, G. Owenson, D. Salman, M. Adda, Security and performance evaluation of master node protocol based reputation blockchain in the bitcoin network, Blockchain: Research and Applications (2021) 100048.

[41] E. Duffield, H. Schinzel, F. Gutierrez, Transaction locking and masternode consensus: A mechanism for mitigating double spending attacks (2014).

[42] M. Babaioff, S. Dobzinski, S. Oren, A. Zohar, On bitcoin and red balloons, in: Proceedings of the 13th ACM conference on electronic commerce, ACM, 2012, pp. 56–73.

[43] D. Stutzbach, R. Rejaie, S. Sen, Characterizing unstructured overlay topologies in modern p2p file-sharing systems, IEEE/ACM Transactions on Networking 16 (2) (2008) 267–280.

[44] T. Neudecker, P. Andelfinger, H. Hartenstein, A simulation model for analysis of attacks on the bitcoin peer-to-peer network, in: Integrated Network Management, 2015 IFIP/IEEE

International Symposium on, IEEE, 2015, pp. 1327–1332.
[45] M. Apostolaki, A. Zohar, L. Vanbever, Hijacking bitcoin: Routing attacks on cryptocurrencies, in: Security and Privacy (SP), 2017 IEEE Symposium on, IEEE, 2017, pp. 375–392.
[46] O. Ugurlu, M. E. Berberler, G. Kızılates, M. Kurt, New algorithm for finding minimum vertex cut set, in: Problems of Cybernetics and Informatics (PCI), 2012 IV International Conf., IEEE, 2012, pp. 1–4.
[47] G. Karypis, V. Kumar, Metis – unstructured graph partitioning and sparse matrix ordering system, version 2.0, Tech. rep., University of Minnesota (1995).
[48] P. Miettinen, M. Honkala, J. Roos, Using METIS and hMETIS algorithms in circuit partitioning, Helsinki Uni. of Tech., 2006.