Cyprus University of Technology

Faculty of Engineering and Technology

Doctoral Dissertation

A Unified Approach to Assessing the Structural Resilience of Blockchain Overlay Networks

Aristodemos Paphitis

Limassol, September 2023

CYPRUS UNIVERSITY OF TECHNOLOGY

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING, COMPUTER ENGINEERING AND INFORMATICS

Doctoral Dissertation

A Unified Approach to Assessing the Structural Resilience of Blockchain Overlay Networks

Aristodemos Paphitis

Limassol, September 2023

Approval Form

Doctoral Dissertation

A Unified Approach to Assessing the Structural Resilience of Blockchain Overlay Networks

Presented by

Aristodemos Paphitis

Supervisor: Dr. Michael Sirivianos, Associate Professor at Cyprus University of Technology

Signature ____

Member of the committee (chair): Dr. Savvas Chatzichristofis, Professor at Neapolis University Pafos

Signature ____

Member of the committee: Dr. Nicolas Tsapatsoulis, Professor at Cyprus University of Technology

Signature ____

Cyprus University of Technology

Limassol, September 2023

Copyrights

Acknowledgments

First, I express my deep gratitude to my supervisor, Dr. Michael Sirivianos. I am extremely fortunate to have been under his guidance. His compassionate approach and ability to alleviate my stress during challenging moments have been truly invaluable. His unwavering belief in me and his knack for turning difficulties into opportunities have consistently lifted my spirits and given me the strength to go forward.

Second, I would like to thank Nicolas Kourtellis for his expertise and valuable feedback and Herodotos Herodotou for his continuous support.

Sincere appreciation goes to my committee members Savvas Chatzichristofis and Nicolas Tsapatsoulis for their constructive critiques and feedback.

Heartfelt thanks go to my longtime friend and colleague Charalampos Sergiou, whose wisdom and guidance have been instrumental in my journey.

My gratitude extends to Lieutenant Colonel Alexis Hadjiprokopis and Major Angelos Angeli for their unwavering support throughout my academic endeavors.

To my family and friends who have supported and encouraged me directly or indirectly, I thank you from the depths of my heart.

ABSTRACT

Blockchains have gained significant attention for their unique properties, such as immutability, public verifiability, and decentralization. These attributes have generated interest in both the software industry and the research academia, sparking enthusiasm for blockchain-based applications. The success of Bitcoin has showcased the potential of blockchain technology, and its distinct features are expected to revolutionize various industries and disrupt sectors that rely on centralized third parties.

Blockchain systems operate through peer-to-peer (P2P) networks, which play a crucial role in ensuring consensus and data propagation. The security and correctness of blockchain applications hinge on the resilience of these P2P networks. In this thesis, we dive into the structural properties of seven distinct blockchain networks, focusing on the implications of their characteristics for network resilience. Our study reveals vulnerabilities to targeted attacks and uncovers hidden interconnections among networks, emphasizing the importance of strengthening network defenses.

To overcome the challenge of accurate topology inference, we introduce a simple yet effective approach. Using the node advertisements shared by the network nodes, we construct connectivity graphs that include any potential connections between peers. Our methodology captures both actual and potential connections, improving our understanding of blockchain P2P networks and their structural characteristics. Furthermore, we avoid classifying nodes and links according to their role or position in the network. Instead, we adopt a unified network model, streamlining the analysis and enabling the identification of shared structural vulnerabilities present in diverse network configurations across various blockchain systems. Addressing these vulnerabilities can lead to improved network resilience.

Our findings shed light on the dynamic nature of blockchain overlay networks and their susceptibility to targeted attacks. We observe variations in the distribution of session lengths over time and a strong correlation between a node's uptime and its degree. We highlight the importance of considering network implementation in blockchain systems and the need for tailored solutions to enhance security and resilience.

Keywords: Blockchain, P2P Networks, Resilience

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF PUBLICATIONS

The work presented in this thesis is already published in peer-reviewed conferences. Specifically:

- Paphitis, A., Kourtellis, N., Sirivianos, M. (2023). Graph Analysis of Blockchain P2P Overlays and Their Security Implications. In: Arief, B., Monreale, A., Sirivianos, M., Li, S. (eds) Security and Privacy in Social Networks and Big Data. SocialSec 2023. Lecture Notes in Computer Science, vol 14097. Springer, Singapore. https://doi.org/10.1007/978-981-99-5177-2_10

- Paphitis, A., Kourtellis, N., Sirivianos, M. (2023). Resilience of Blockchain Overlay Networks. In: Li, S., Manulis, M., Miyaji, A. (eds) Network and System Security. NSS 2023. Lecture Notes in Computer Science, vol 13983. Springer, Cham. https://doi.org/10.1007/978-3-031-39828-5_6

Also, during my studies I have conducted additional research work that has been published in peer-reviewed journals, conference, and workshop papers:

- M. A. Georgiou, M. Panayiotou, L. Odysseos, A. Paphitis, M. Sirivianos, and H. Herodotou. 2021. Attaining Workload Scalability and Strong Consistency for Replicated Databases with Hihooi. In Proceedings of the 2021 International Conference on Management of Data (SIGMOD '21). Association for Computing Machinery, New York, NY, USA, 2721–2725. https://doi.org/10.1145/3448016.3452746

- M. A. Georgiou, A. Paphitis, M. Sirivianos and H. Herodotou, "Hihooi: A Database Replication Middleware for Scaling Transactional Databases Consistently," in IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 2, pp. 691-707, 1 Feb. 2022, doi: 10.1109/TKDE.2020.2987560.

- C. Sergiou, A. Paphitis, L. Kanaris, and S. Stavrou. 2019. An experimental study of IoT transmission power, in outdoor environment, using Crossbow TelosB nodes. In Proceedings of the 23rd Pan-Hellenic Conference on Informatics (PCI '19). Association for Computing Machinery, New York, NY, USA, 128–133. https://doi.org/10.1145/3368640.3368649

- M. A. Georgiou, A. Paphitis, M. Sirivianos and H. Herodotou, ”Towards Auto-Scaling Existing Transactional Databases with Strong Consistency,” 2019 IEEE 35th International Conference on Data Engineering Workshops (ICDEW), Macao, China, 2019, pp. 107-112, doi: 10.1109/ICDEW.2019.00-26.

- C. Sergiou, V. Vassiliou and A. Paphitis, ”Estimating queue formation rate in Wireless Sensor Networks using a fluid dynamic model”, 2015 IEEE Symposium on Computers and Communication (ISCC), Larnaca, Cyprus, 2015, pp. 544-548, doi: 10.1109/ISCC.2015.7405571.

- C. Sergiou, V. Vassiliou, A. Paphitis, Congestion control in Wireless Sensor Networks through dynamic alternative path selection, Computer Networks, Volume 75, Part A, 2014, Pages 226-238, ISSN 1389-1286, `https://doi.org/10.1016/j.comnet.2014.10.007`.

Chapter 1

Introduction

Blockchains have gained significant attention because of their distinct properties: immutability, public verifiability, and decentralization. Immutability ensures data integrity. Public verifiability allows any external entity to validate the correctness of any operation; thus maintaining and increasing trust and transparency. Finally, decentralization increases resilience and reliability as it removes central, single points of compromise. This technology has generated enthusiasm among both the software industry and research academia due to its ability to be maintained without the need to trust any single individual or entity. However, some people are sceptical towards the technology and support that blockchains merely displace rather than eliminate trust.[1] However, the blockchain market continues to expand steadily, with a growing demand for blockchain professionals from various organizations.[2]

Following Bitcoin's success, the unique features of blockchains have increased this technology's visibility and are expected to revolutionize various industries and disrupt business sectors that traditionally rely on trusted centralized third parties. Due to the decentralization of trust and their immutable nature, blockchain systems have introduced novel solutions to long-standing problems such as secure and transparent transactions, efficient supply management, and decentralized finance. Numerous other applications have been identified for a wide range of use cases, including healthcare, advertising, insurance, copyright protection, energy, cybersecurity, and government [7, 111, 18, 17].

[1]Bruce Schneier. *Blockchain and Trust*. 2019. URL: `https://www.schneier.com/blog/archives/2019/02/blockchain_and_.html`.

[2]*Blockchain Technology Market Size, Share & Growth [2030]*. 2023. URL: `https://www.fortunebusinessinsights.com/industry-reports/blockchain-market-100072`.

Public or permission-less blockchains rely on decentralized peer-to-peer (P2P) networks for their operation. Such network overlays are easily constructed and enable fast diffusion of information. Peers in a blockchain network must constantly maintain a local copy of all transactions and blocks. The swift diffusion of this information is important for the correct and fair operation of the system. The P2P overlay is the backbone of a blockchain system, facilitating the propagation of transactions and blocks to all participating peers.

The availability of the P2P overlay is essential for the propagation of blockchain data. Flaws and vulnerabilities in the network layer can compromise the guarantees offered by blockchain applications and their consensus protocols. Network-level attacks can be used to facilitate or commit double spending. Despite the security provided by the consensus protocol, network-level attacks have the potential to weaken consensus in specific parts of the network and enable double spending or other attacks. Therefore, the P2P infrastructure is an important architectural part of a blockchain system; it defines the level of security, resilience, and scalability of the system [22, 43].

This, in turn, raises the question of whether the P2P infrastructure ensures sufficient resilience and performance. It is desirable for the design and implementation of blockchain networks to consider these factors in order to ensure the overall security and reliability of a blockchain system. Studying the structural properties, topological characteristics, and vulnerabilities of these networks is crucial to realize the full potential of blockchain technology and ensure its resilience against emerging threats [22, 43]. *It is therefore important to have a thorough understanding of the structure of these networks to reveal possible limitations and vulnerabilities.* This would allow blockchain architects to mitigate damage from both natural failures and targeted attacks. Blockchains are already being used to process large amounts of money; considering their potential application in other aspects of everyday life, they become an attractive target for ill-intentioned attacks by malicious actors. Therefore, it is critical to investigate whether small-scale attacks against a few nodes could provide attackers with significant advantages.

Previous research has largely focused on the cryptographic and consensus aspects of blockchain systems, with comparatively little attention paid to the underlying P2P networks. This lack of knowledge has been acknowledged by the research community [27, 11, 34, 28, 52], with the design, connectivity properties, and interdependencies of blockchain networks still largely unexplored.

This gap in our knowledge is partly due to an implicit assumption about the reliability and safety of Internet communications. Similarly, the distributed nature of blockchain systems often leads to the false conclusion of safety and resilience. Furthermore, the lack of formal specification and documentation of blockchain P2P protocols hinders our ability to study their network properties.

One other probable reason for this knowledge gap in the literature is the obscure state of the underlying network structure. To study a network in depth and analyze its characteristics, one needs an accurate representation of its layout; the arrangement of links and nodes in the network, i.e., the network topology. By design, these networks rely on topology hiding techniques to discourage adversarial actions against the participating peers. Such actions could be user deanonymization, node eclipsing, transaction censorship, and partitioning attacks that strategically target central nodes to disrupt the system's operation. In the recent past, several modifications to network formation protocols have been implemented, specifically to prevent any leakage of topology-related information from participating peers. Therefore, obtaining precise topological details for such networks is unattainable, further impeding their comprehensive analysis.

This thesis aims to bridge this gap by examining the structural properties of seven distinct blockchain networks, with a specific focus on the resilience implications of their characteristics. *The objective is to gain a comprehensive understanding of the network properties of blockchain P2P overlays and establish whether their structure is resilient enough to be trusted.* The results collected in this work indicate that this is not the case.

The methodology selected for this study centers on the unified modeling of blockchain networks, where all network components receive equal treatment. Nodes are not categorized based on their roles, whether they are miners, full nodes, or wallets, nor by their connectivity status, whether they are reachable or unreachable. Additionally, the connections between nodes are handled in a uniform way. This methodology enables us to reach conclusions with broad applicability across a diverse spectrum of blockchain systems, strengthening the generality of our findings.

To overcome the challenge of accurate topology inference, we introduce a novel approach that addresses the limitations of existing methods. Previously proposed approaches are no longer viable due to changes in the reference clients or impractical due to excessive transaction fees. (see Section 3.4 and Table 3.1). In addition, these methods require a significant amount of network resources when applied simultaneously to multiple networks because they involve the establishment of numerous peer connections.

Our solution centers on unveiling potential connections between pairs of nodes (peers) in order to circumvent the topology inference problem. By considering a connection possible if one node lists another as a known address, we generate comprehensive connectivity graphs that cover the vast majority of potential links, including actual connections. This trade-off between accuracy and completeness is guided by our objective of identifying structural deficiencies in overlays. We posit that the realized topology of an overlay is unlikely to be resilient if our inferred topology of possible connections is not (see Section 4.2).

Node advertisements serve as an easily accessible aspect of blockchain P2P networks. These messages facilitate network bootstrapping and enable nodes to discover and connect with peers. By monitoring these advertisements, we construct connectivity graphs that reveal the relationships between nodes. Repeatedly querying reachable nodes for their known peers allows us to create a node's network view. Crucially, this methodology captures both actual and potential connections, accounting for future links that emerge as nodes replace disconnected peers. This innovative approach improves our understanding of blockchain P2P networks and their structural characteristics.

1.1 Research Questions

Having outlined our objectives and our methodology, we now turn into the specific research questions that this work seeks to address.

RQ1:

Do all blockchain overlays share similar properties? What are their structural and network characteristics, and how do they compare to other well-known networks such as the Internet AS topology, the Web, and Online Social Networks?

By understanding these differences, researchers can gain insight into the unique properties and behaviors of blockchain networks. Additionally, the characteristics collected from these networks can provide valuable information regarding their structural resilience. The analysis will help determine whether these networks exhibit resilient or vulnerable characteristics, contributing to a deeper understanding of their overall robustness.

RQ2:

To what extent are blockchain overlays prone to random failures and targeted attacks? How does the concentration of nodes within the same Autonomous Systems impact the resilience of the system?

Random failures can occur for various reasons, such as hardware malfunctions or network outages. These types of failure may be more likely to occur in blockchain overlays due to their decentralized nature, as they rely on a network of nodes rather than a single central server.

Targeted attacks, on the other hand, are a deliberate attempt to disrupt or compromise the system. These attacks can come in many forms, such as distributed denial of service (DDoS) attacks or attempts to manipulate the data within the blockchain. The decentralized nature of blockchain overlays may make them more vulnerable to these types of attacks, as there are multiple points of entry for attackers to exploit.

The concentration of nodes within the same Autonomous Systems (AS) also plays a crucial role in the resilience of blockchain overlays. When a significant number of nodes are located within the same AS, the risk of a single point of failure increases. For example, if the AS experiences an outage or becomes the target of an attack, a large number of nodes could be affected simultaneously, leading to potential disruptions in the stability and reliability of the network. Therefore, understanding and mitigating these risks is essential to ensure the robustness and security of blockchain overlays.

RQ3:

What is the interconnection between distinct blockchains when peers and links participate simultaneously in multiple networks? How does this relationship affect the resilience of the blockchain ecosystem?

The interconnection between different blockchains arises when nodes and links simultaneously participate in multiple networks. This concurrent participation enables nodes to establish connections with nodes from other blockchain networks, forming relationships that have substantial implications for the resilience of the entire blockchain ecosystem. If a node involved in multiple networks encounters a failure or becomes inactive, it could disrupt the stability of all interconnected networks. Additionally, an attack or failure in one network has the potential to spread to other networks through these interconnected links, affecting the overall resilience of the ecosystem. Understanding and analyzing these relationships are essential to assess the robustness and security of the blockchain ecosystem.

RQ4:

How do blockchain networks evolve over time? Are the longitudinal characteristics of these networks related to the other network properties of their peers?

Blockchain networks are highly dynamic systems that evolve as nodes join and leave the network and network conditions change. Analyzing the longitudinal characteristics of these networks and their relationships with other network properties of their peers is crucial to gain valuable insights into their underlying structures and behaviors. This understanding can enable researchers to understand the dynamics of the network, identify potential risks, and discover opportunities to improve the resilience and performance of these networks.

1.2 Contributions

This thesis makes several notable contributions that shed light on the structural resilience of blockchain overlay networks. To our knowledge, this work is the first to explore and discuss the network properties of these overlays. By analyzing their structural behavior, we gain valuable insights into their limitations, vulnerabilities, and potential avenues for improvement. This analysis is crucial for developing strategies that improve the resilience, scalability, and security of blockchain systems. In more detail, we make the following contributions:

1. **Insights into Resilience:** One of the key contributions of this thesis lies in the unexpected insights we have uncovered regarding the resilience of blockchain overlay networks. Surprisingly, our study has revealed that many of the networks we analyzed are susceptible to various attack strategies, exposing significant vulnerabilities that challenge the widely perceived decentralization and resilience of blockchain networks.

2. **Centralization and Targeted Attacks:** Contrary to the common notion of decentralization, our findings showcase a concerning trend of high centralization of nodes within blockchain networks. This centralization renders networks vulnerable to targeted attacks, where the removal of just a few nodes can severely disrupt the stability of major

blockchains. This observation calls for a closer examination of the network structures and strategies to mitigate such vulnerabilities. The fact that our study relies on graphs of potential connectivity, which overestimate the number of actual links, adds further to our concerns.

3. **Overlapping Nodes and Ecosystem Vulnerability:** Our research is the first to identify and highlight the existence of nodes participating in multiple blockchains simultaneously, referred to as "overlapping nodes". These nodes are consistently present across distinct blockchain networks and constitute a significant proportion of each network. The presence of these overlapping nodes exposes vulnerabilities that could potentially allow malicious actors to exploit them, leading to disruptions across multiple networks.

4. **Longitudinal Analysis and Network Dynamics:** Through longitudinal analysis, we have established a correlation between node up-time and degree, which presents attackers with the opportunity to identify crucial nodes within networks that rely on topology hiding techniques. This in-depth examination of network dynamics provides valuable insights into how blockchain networks evolve over time and offers avenues to enhance their resilience against various challenges.

5. **Spatial Concentration and Network Resilience:** Our examination extends to the spatial concentration of nodes within Autonomous Systems and its impact on network resilience. By analyzing the distribution of nodes within these systems, we shed light on latent risks and challenges that could undermine the stability and security of the broader blockchain ecosystem.

6. **Innovative Methodology for Analysis:** In addition to the substantial findings presented, this thesis introduces an innovative and efficient methodology for analyzing the topological characteristics of blockchain Peer-to-Peer (P2P) networks.

   Firstly, we adopt an approach that eliminates the distinction between different node types or links between peers, treating all network elements with equal importance. In particular, we avoid categorizing nodes based on whether they are miners or relay peers. Furthermore, we do not differentiate unreachable peers in our analysis, as these nodes play a significant role in the system. This strategy broadens the applicability of our methodology to various networks that might have different characteristics and features. In addition, it simplifies the analysis process, allowing a focus on structural attributes that are relevant for a wide array of blockchain overlay networks.

Secondly, our methodology involves the construction of connectivity graphs that include all potential connections available within the network. This is achieved by leveraging the Node Discovery mechanisms embedded in the network protocols. The methodology is rooted in fundamental principles of graph theory, establishing a robust basis for identifying structural vulnerabilities within these networks. Rigorous experimental validation further affirms the effectiveness of this approach in uncovering structural weaknesses within blockchain networks.

In summary, this thesis makes significant contributions by uncovering vulnerabilities within overlay networks and the broader blockchain ecosystem. By highlighting these deficiencies, we provide researchers with a deeper understanding of blockchain network overlays, paving the way for more resilient and robust network structures.

1.3 Thesis Outline

The remaining chapters of the thesis are organized as follows:

- In Chapter 2, we introduce preliminary concepts on blockchains, with a focus on their network aspects.
- Chapter 3 presents a review of the related literature.
- In Chapter 4, we outline the main idea of this work and explain how the collected connectivity graphs are used to study these networks. We provide a detailed explanation of the methodology used to obtain these graphs and the methods used to analyze them.
- Chapter 5 presents and discusses the collected results. First, we examine the structural properties of the networks studied and discuss the signs of vulnerability. We then analyze the behavior of the networks under random and targeted attacks.
- Finally, Chapter 6 concludes the thesis and offers guidelines for future research perspectives.

Chapter 2

Background

Blockchain technology has emerged as a revolutionary means of processing transactions. At its core, the blockchain is a decentralized and distributed digital ledger that enables secure and transparent record-keeping of transactions.

The fundamental element of a blockchain is the “block”. Each block contains a collection of transactions, data, and a unique cryptographic hash. Blocks are linked in a sequential manner, forming a chain, hence the name. Chaining of blocks ensures that the data within them is tamper-proof and immutable, as any change to a single block would require altering all subsequent blocks. This is because each block references the unique hash of the previous block in its data; i.e., the current block hash depends on the previous block hash. The use of appropriate hashing functions makes it computationally infeasible to alter a block’s contents, while verification of a valid block is efficient and fast. This provides a high level of security and enables public verifiability.

Central to the functioning of a blockchain is its consensus algorithm, which determines how new blocks are added to the ledger and how consensus is reached among participants. One of the most widely known consensus mechanisms is the Proof of Work (PoW), utilized by the pioneer blockchain, Bitcoin. In PoW, participants, known as miners, compete to solve a computational puzzle to add new blocks to the blockchain. This process requires significant computational power, time, and energy, ensuring that transactions are verified and adding a layer of security against malicious attacks. Any miner who solves the puzzle publishes his block on the network. Acceptance of the miner’s block is confirmed when a new block is added on top of it, making it part of the chain.

However, the probabilistic nature of consensus mechanisms such as PoW can lead to the occurrence of forks. Forks occur when different miners successfully mine and add a new block to the blockchain simultaneously. This can result in the creation of multiple competing chains, each with its own version of the truth. Forks can be categorized as hard forks, which result in a permanent split of the blockchain, or soft forks, which introduce backward compatible changes. Forks are resolved through consensus mechanisms and community agreement, with participants choosing which chain to follow based on factors such as network security, community support, and consensus rules. Bitcoin's consensus rules dictate to accept the fork branch with the most proof of work invested in it.

Miners are incentivized to participate in blockchain networks by rewarding them with cryptocurrency tokens that serve as both a form of payment and an incentive mechanism. In the case of the Proof-of-Work (PoW) consensus algorithm, miners invest computational resources and energy to solve puzzles in order to validate transactions and add new blocks to the blockchain. The first miner to successfully solve the puzzle is granted the right to add the next block and is rewarded with a predetermined amount of cryptocurrency, often referred to as a "block reward". This reward not only compensates miners for their efforts but also creates new tokens, effectively expanding the cryptocurrency supply.

In addition, the incentive structure aligns the interests of miners with the overall health and security of the blockchain network. As miners contribute their resources to maintain the integrity of the ledger, they have a vested interest in ensuring that the network remains secure and trustworthy. This alignment of interests helps to prevent malicious behavior and promotes the long-term stability of the a blockchain system.

We have provided a brief summary of the architectural elements of blockchains. For a deeper exploration of the topic, interested readers are recommended to look into the foundational works of Bonneau [11], Tschorsch[98], and Narayanan [75].

2.1 Bitcoin's P2P Network Protocol

Bitcoin's consensus mechanism depends on the ability to broadcast messages to all nodes in the network. When a new transaction is created or a new block is mined, this information must reach all nodes as quickly as possible. Bitcoin nodes are interconnected through a P2P network. Each node maintains a list of known peers' addresses to connect to. All message exchanges are point-to-point; no multi-hop propagation occurs. A gossip protocol is used to quickly spread information.

Specification No official documentation has ever been released that describes the network protocol. Sources of our knowledge include the official bitcoin core codebase[1] and related works [54, 34, 24, 73, 25].[2] Since the protocol implementation is subject to change upon each release of the Bitcoin Core, obtaining up-to-date information on how nodes communicate and react to messages can be challenging. This work assumes the implementation of Bitcoin Core client version 0.20.0.[3]

Protocol Messages

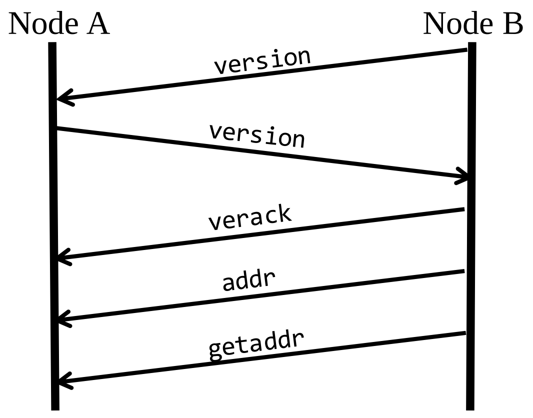
- `version`: exchanged to establish a new connection, contains information about the client, such as protocol version, services, IP addresses, and so on;
- `verack` (version Ack): sent in response to the version;
- `ping/pong`: exchanged to keep the connection alive;
- `getaddr` (get address): request list of known peers;
- `addr` (address): contains a list of IP:port addresses of known peers. It can be sent after a *trickling* event unsolicited or sent in response to `getaddr`;
- `inv` (inventory): announces known transactions or blocks, it contains a list of hashes (not the actual data);
- `getdata`: requests the data of a single block or transaction by hash;
- `tx` (transaction): sent in response to getdata, it contains a single transaction data;
- `block`: sent in response to getdata, it contains a single block data;

Initial Bootstrapping The Bitcoin network is open to participation and everyone can join. A new node must connect to existing peers on the network to be able to send and receive messages. The participating nodes are not always online and cannot be known in advance, while newcomers to the network need a way to contact existing peers. To facilitate new

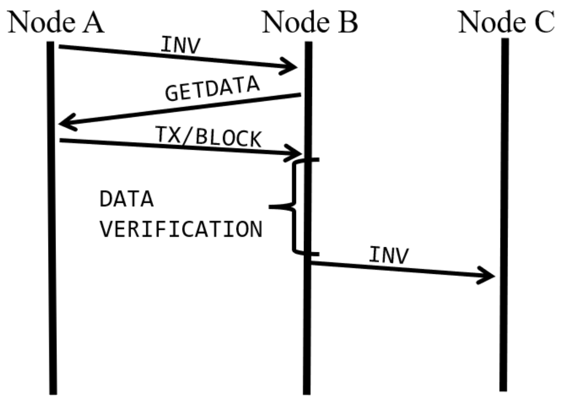
[1]Bitcoin Core Developers. *Bitcoin Core integration/staging tree*. 2021. URL: `https://github.com/bitcoin/bitcoin`.

[2]Bitcoin Core Developers. *Bitcoin P2P Network*. 2021. URL: `https://developer.bitcoin.org/devguide/p2p_network.html`.

[3]`https://github.com/bitcoin/bitcoin/releases/tag/v0.20.0`



(a) Connection establishment



(b) Bitcoin message propagation

Figure 2.1: Message Propagation in Bitcoin

nodes that want to join the network, the client contains a list of some hard-coded DNS seeds that anyone can contact to retrieve an up-to-date list of online nodes. These *seed nodes* are maintained by the Bitcoin developer community. A new node sends `getaddr` messages to the seeds. The `addr` replies may contain up to 1000 IP addresses known to the DNS seeds. The node stores these addresses in a data structure, the `ADDRMAN`, and tries to initiate a connection to them. If the connection is successful, the node asks the connected peer for more addresses via a `getaddr` message and repeats the same procedure. It then tries to maintain 8 outgoing connections at all times. The DNS seeds are also called bootstrap nodes, and the procedure is also known as bootstrapping. [4] Other mechanisms exist to perform an initial discovery of online peers, e.g., by manually adding known IP addresses.

Connecting To Peers Connecting to a peer is done by sending a `version` message, which contains the client's version number, its most recent block number (or block height), and current time to the remote node. This is depicted in Figure 2.1a, where Node *B* initiates an outbound connection to Node *A*. The remote node responds with its own `version` message and a `verack` message. Then, the node that initiated the connection advertises its own address via an `addr` message, and requests a list of known peers with a `getaddr`. In order to maintain a connection with a peer, the nodes exchange `ping/pong` messages periodically. If, after a predefined period, no message is received, the connection is assumed to be closed.

[4]The list of the hardcoded seeds are at `https://github.com/bitcoin/bitcoin/blob/38d06e1561013f4ca845fd5ba6ffcc64de67f9c0/src/chainparams.cpp#L123`

Inbound and Outbound connections A connection initiated by node `B` to node `A` is considered an outgoing connection for node `B` and an inbound connection for node `A`. By default, a peer maintains up to 8 outbound connections and accepts a maximum of 117 inbound connections, totaling 125 connections. Both types of connections are ordinary TCP connections, hence bidirectional. However, the client keeps track of which connections are outbound and inbound, since the network protocol applies different rules to each group. This diversity was introduced to harden topology hiding to mitigate fingerprinting and eclipse attacks[5].

Node Discovery When establishing a new outgoing connection to another peer, a node will advertise its own IP address with an `addr` message and then ask for known addresses via a `getaddr` message. This is also done periodically. Connected peers that receive an `addr` message forward it to 1 or 2 other connected peers, depending on the reachability of the received address. Nodes that receive the forwarded message will forward it further to 1 or 2 of their neighbors, and so forth. The following rules apply:

- `getaddr` messages from outbound peers are not answered. This was introduced as a mitigation to prevent a fingerprinting attack.
- Only one `getaddr` response is sent per connection, in order to reduce bandwidth usage. By disconnecting and reconnecting, more than one `getaddr` messages can be sent.
- After the initial `getaddr` further addresses are propagated via unsolicited, periodic `addr` forwarding.
- `addr` messages are propagated to 1 or 2 of the connected peers. These two peers remain the same for a 24-hour window (unless they disconnect). Unreachable addresses are relayed to only one peer, while reachable addresses are relayed to two. Additionally, `ADDRMAN` keeps track of which peers were forwarded which addresses to avoid repeating an already known address to the same peer.

Address Management The node discovery process runs continuously. All collected addresses are managed by the address manager (`ADDRMAN`). According to the official implemen-

[5]Bitcoin Core Developers. *Ignore GETADDR msg from inbound connections.* 2020. URL: `https://github.com/bitcoin/bitcoin/blob/37e9f07996d3a7504ea54180d188ca91fdf0c884/src/net_processing.cpp#L3567`.

tation[6], the design goals of `ADDRMAN` are the following:

- Keep the address tables in-memory.
- Asynchronously dump the entire table to `peers.dat`
- Make sure that an adversary cannot fill the entire table with malicious nodes.

To this end, addresses are organized into buckets that can store up to 64 entries. There are two types of buckets: *tried* and *new*.

Addresses to which the node has not yet successfully connected go into 1024 *new* buckets. Each address is accompanied by the *source node* that advertised it. To increase diversity in selection and to mitigate routing attacks [6], 64 buckets are chosen according to the IP address of the *source node* or its AS. The actual bucket is chosen from one of these based on the range in which the address itself is located, and the position in the bucket is chosen based on the full address. One single address can occur in up to eight different buckets to increase the selection chances for addresses that are seen frequently. When a new address is insert into an already occupied bucket position, the existing entry will not be replaced unless it already exists in a different bucket or does not satisfy certain quality criteria.

Addresses that are known to be accessible go into 256 *tried* buckets. Each address range selects at random 8 of these buckets. The actual bucket is chosen from one of these, based on the full address. Older addresses already in the tried bucket are favored in the case of bucket collisions. Before evicting an existing address from the tried bucket, the node tests whether it still accepts connections. If not, it is moved in the new bucket.

Peer Selection As long as a node has fewer than 8 outgoing connections, it will try to connect to known peers. An IP address is selected at random with equal probability from the *new* and *tried* buckets. When selecting addresses, the node tries to diversify connections so that not all outgoing connections are from the same IP domain. This check is employed as a means to mitigate routing and eclipsing attacks [6, 54].

Message propagation Transaction and block propagation follow a three-round protocol. Inventory (`INV`) messages are used to signal the presence of one or more transactions or

[6]Bitcoin Core Developers. *Bitcoin Stochastic address manager*. 2020. URL: `https://github.com/bitcoin/bitcoin/blob/be3af4f31089726267ce2dbdd6c9c153bb5aeae1/src/addrman.h#L108`.

blocks. When a node receives or generates a new transaction or block, it informs its connected peers by creating an `INV` message that contains the transaction hash (see Figure 2.1b). Peers unaware of the announced item will request it using a `GETDATA` message. Furthermore, if a node receives an `INV` message requesting a specific item and sends a `GETDATA` message to retrieve it, the requester will wait up to 2 minutes for the offering node to respond with the item. If the initial node does not respond, any subsequent requests for the same item will be queued and served on a first-in, first-out basis.

Node characterization Bitcoin's initial vision was that all participants would be equal in the system. But with economic growth, specialized mining equipment (ASICs[7]) came to the fore, and mining became unfeasible for general purpose personal computers. Additionally, the sheer size of the blockchain in terms of space required a dedicated machine to be able to store all blocks and transactions. The size problem was partially addressed by allowing nodes to participate without having to download the entire blockchain; nodes could choose to only store block headers or prune the blockchain by keeping only the latest information. This, of course, changes their capability to verify past transactions or blocks. This diversity in capabilities has led to a network where nodes offer different services depending on their configuration.

Nodes can be characterized as miners if they participate in the proof of work, or non-mining if they do not. Full nodes are those that store the blockchain in its entirety, including all past transactions and blocks. Nodes can indicate to other peers what services they provide by including their capabilities during the initial handshake.

Currently, the vast majority of mining nodes are part of mining pools. Usually, the mining nodes are not directly connected to the network. They only receive transactions and block data to perform hashing computations, as required by the proof-of-work algorithm. Transactions are disseminated from a pool server, usually by means of specialized protocols, i.e., Stratum[8].

An other special case of nodes are those that do not accept connections from other peers, either due to limited connectivity or because they are located behind firewalls or Network Address Translation services (NAT). Nodes that can initiate connections to the reachable part of the network but do not accept incoming connections are known as *unreachable* nodes. These nodes have been largely overlooked, but recent work acknowledges their importance.

The different roles that peers can have in the network are shown in Figure 2.2. Unreachable

[7]Application Specific Integrated Circuits

[8]SlushPool. *STRATUM V1 Docs*. 2022. URL: `https://braiins.com/stratum-v1/docs`.

nodes (blue) in the unreachable region of the network can initiate connections toward reachable peers (black) in the reachable area but cannot connect to other unreachable peers. Reachable nodes can connect to any other reachable node and also accept incoming connections from any unreachable node. Mining nodes are mostly located in mining pools, since mining alone is no longer profitable. A mining pool server creates a block template that includes transactions and splits it between the pool members who work until one of them finds a valid hash value. Some mining farms have all their mining equipment in one location. Other mining pools are distributed; each miner operates his equipment from his own location. Mining equipment is usually isolated from the network, but miners may choose to connect their equipment to the P2P network, not only to the Pool Server. It is not uncommon for miners to switch pools [96], in order to increase their revenue.

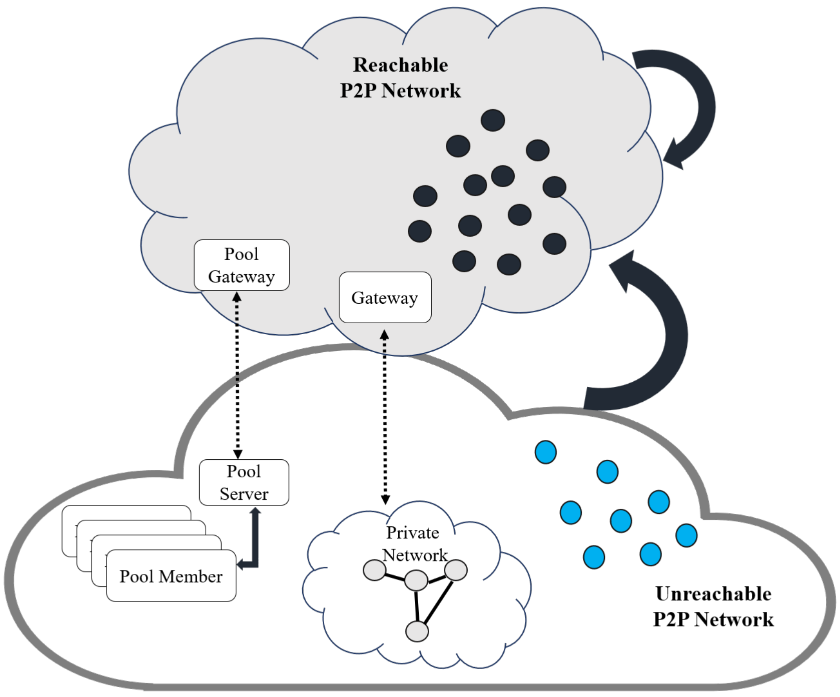


Figure 2.2: Network state

2.2 Ethereum's P2P Network

Ethereum was designed as a next-generation decentralized application platform. Its most prominent feature is its built-in Turing-complete programming language. Ethereum can be viewed as a transaction-based state machine. Starting with an initial state, valid transactions can be applied to it, morph it, and return a new state. Similarly to Bitcoin, transactions are collected in blocks and the blocks are linked in a hash chain.

Ethereum's founders identified the applicability of decentralized computation beyond trading, so Ethereum was designed to support the execution of programs, which are called smart contracts.

Smart contracts are expressed as specific opcodes according to the Ethereum Virtual Machine (EVM) [117]. Ethereum also operates as a cryptocurrency that supports the transfer of tokens called *ether* between Ethereum accounts. Ethereum nodes are incentivized to participate by collecting fees per smart contract computation.

Smart contracts have been shown to be applicable in many domains, including financial industry, public sector, and cross-industry.

Similar to Bitcoin and other cryptocurrencies, Ethereum's blockchain is also supported by a P2P network. Ethereum's network stack is somewhat more complicated, consisting of two separate protocols running concurrently: a) the discovery protocol and b) the DEVP2P protocol. Discovery finds new peers that are then used by the DEVP2P protocol to form connections with them. All blockchain-related data are transmitted via the DEVP2P protocol. Ethereum's reference network implementation is outlined in[9] and both network protocols are collectively referenced as DEVP2P. The protocols and sample message exchange are depicted in Figure 2.3.

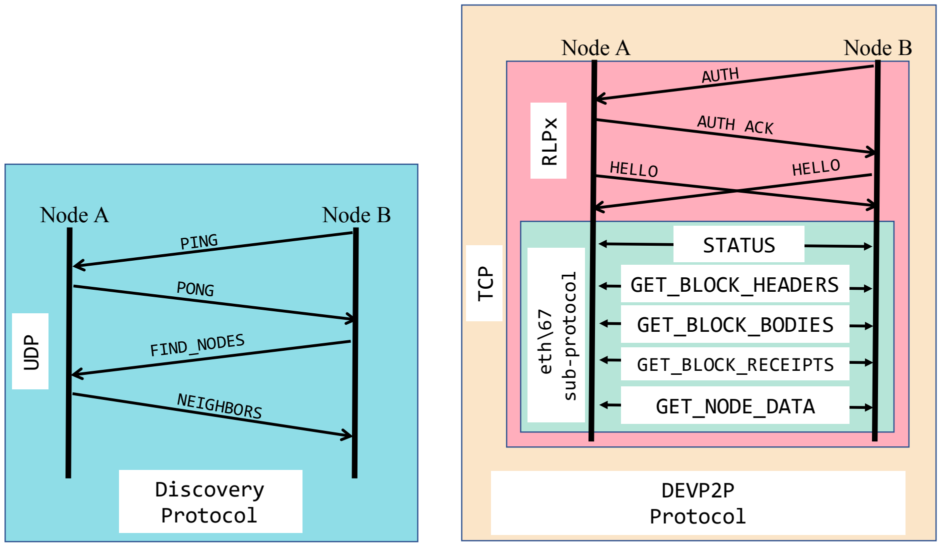


Figure 2.3: Ethereum's Protocols

[9]Ethereum. *Ethereum peer-to-peer networking specifications*. 2014. URL: `https://github.com/ethereum/devp2p`.

2.2.1 The Discovery protocol

As the name suggests, it is used to discover new peers in the network. It is a UDP protocol based on the Kademlia distributed hash table (DHT) [72]. Kademlia provides an efficient means for storing and retrieving content from a distributed P2P network. Its key property is that each content item stored can be discovered by querying a small number of network nodes; $O(\log n)$ in a network of size n. In contrast, the DHT in Ethereum is only used to find new peers. The upshot here is that Ethereum inherits most of the complicated logic of Kademlia protocol, even though it rarely uses the key property for which Kademlia was designed, i.e., content retrieval [71].

The node discovery process is based on a routing algorithm, with the goal of forming an efficient network topology with a low diameter. In Ethereum, every node has a cryptographic identity, a key on the secp256k1 elliptic curve. The public key of the node serves as its identifier or 'node ID' (`ENODEID`).

Initially, the node starts querying the already known peers for other nodes. If no peer is already known, a set of bootstrap nodes, hardcoded in the client, is used. The asking node sends a `FINDNODE` message with a *target* parameter. In most Ethereum clients, the target parameter has the `ENODEID` value of the asking node but a custom implementation can specify any value at will. When a `FINDNODE` is received, the recipient replies with a `NEIGHBORS` packet that contains the 16 nodes closest to the requested target, found in its local table. The distance to a node is calculated using the `logdist` function, which returns the logarithm of a bitwise XOR on the hash values of two `ENODEIDs`: $log_2[Hash(target) \oplus Hash(enode)]$ The integer value of this bitwise XOR is the distance between two `ENODEIDs`. To protect against traffic amplification attacks, replies to `FINDNODE` requests are only sent to verified peers, i.e., peers that have sent a sufficiently recent `PONG` message. [10] Additionally, Ethereum's client implementations impose a 4 second delay between successive `FINDNODE` messages. Nodes create 16 UDP connections at most, to support the discovery process.

Peer Management Known nodes are stored in a routing table consisting of k-buckets. The table is kept in memory and is emptied at each reboot. There exist 256 buckets, each of which holds up to $k = 16$ entries. The entries in each bucket are sorted by time last seen. Whenever a new node is encountered, it can be inserted into the corresponding bucket. If the bucket is full,

[10] `https://github.com/ethereum/devp2p/blob/master/discv4.md#known-issues-in-the-current-version`

the eviction policy favors older nodes: the least recently seen node in the bucket is re-validated by sending a `PING` packet. If no reply is received, it is considered dead, and it is replaced.

All nodes are periodically inserted into a persistent database. There is no limit to the size of the database. Nodes that do not respond to a `PING` message for more than 24 hours are deleted from the database. The entries in the database are used to connect a node during restarts.

2.2.2 RLPx

The RLPx transport protocol is a TCP-based protocol and takes its name after the Recursive-Length Prefix (RLP) serialization format, which is extensively used in Ethereum's execution environment. It uses encryption and it can server as communication protocol for various applications; it's use is not limited to Ethereum. RLPx carries encrypted messages belonging to one or more *capabilities* which are negotiated during connection establishment. Node IDs also function as public keys and are used in RLPx to establish an authenticated and secure TCP connection. The initial RLPx handshake uses an asymmetric encryption technique called Elliptic Curve Integrated Encryption Scheme (ECIES). An RLPx connection is established by creating a TCP connection and exchanging keys for further encrypted and authenticated communication, through the `AUTH` and `AUTHACK` messages. After the initial handshake, both peers must send either a `HELLO` or `Disconnect` message. Upon receiving the `HELLO` message, a session is active and any other message may be sent. The `HELLO` message also includes a list of supported capabilities and their respective version. Capabilities refer to P2P sub-protocols available to Ethereum nodes. The Light Ethereum Subprotocol (les/4) for example, is the protocol used by light-clients, which only download block headers as they appear and fetch other parts of the blockchain on-demand. The Ethereum Wire Protocol (eth) facilitates exchange of Ethereum blockchain information between peers. The RLPx frames packets in a way that allows nodes to exchange messages on various sub-protocols.

The total number of TCP connections at any time is controlled by the `maxpeers` variable that can be defined during the client's startup. The default values of `maxpeers` for Geth and Parity, the most popoular ethereum client implementations, are set to 25 and 50 respectively[11][12]. There is no distinction between *outbound* or *inbound* connections in the protocol logic,

[11]Go Ethereum. *Getting Started with Geth.* 2022. URL: `https://geth.ethereum.org/docs/getting-started`.

[12]Parity Ethereum. *Parity Ethereum.* 2022. URL: `https://github.com/openethereum/parity-ethereum`.

contrary to Bitcoin.

2.2.3 Ethereum's DEVP2P

The DEVP2P protocol uses RLPx transport for communication. After RLPx session establishment, peers communicate via application-specific sub-protocols. For example, Ethereum's sub-protocol, used to exchange blocks and transactions, is referenced as `eth/67`. After peer nodes have been discovered and a secure TCP connection is established using RLPx, DEVP2P negotiates an application session between the two connected peers. The initiating node sends a `HELLO` message, which includes its *ENODEID*, DEVP2P version, client version string, supported application protocols/versions, and the TCP port number (30303 by default) on which the node is listening. Based on the `HELLO` message information, nodes may begin to transmit application data packets over DEVP2P, or end communication if they have incompatible protocol versions. DEVP2P nodes will periodically send DEVP2P PING messages (different from Discovery PINGs) at an interval set by the client to ensure that their connected peers are still online. At any time after protocol negotiation, a `Disconnect` message may be sent, which may include an error code explaining the reason. If a corresponding DEVP2P PONG message is not received within the maximum allowed idle time, then the node will send a DISCONNECT message.

2.2.4 The Ethereum application-level protocol

The Ethereum subprotocol runs on top of DEVP2P and is denoted as 'eth' during DEVP2P `HELLO` exchange. At a high level, the Ethereum sub-protocol is used to retrieve and store information on the Ethereum blockchain.

The first message that must be sent by both peer nodes after the DEVP2P `HELLO` handshake is a `STATUS` message, which conveys the current state of a node's blockchain. It contains the version of the node protocol (Ethereum), the network ID (there are several distinct Ethereum networks), and the hash of the first block (genesis hash) on the blockchain, since there may also be multiple different blockchains for a single network ID. Indicatively, a 2018 measurement study found that almost half of the nodes participating in the network do not support the official Ethereum blockchain but are operating on different chains or networks [62]. In fact, the study revealed more than 4000 different network ids, with 19000 different genesis hashes. The mainstream Ethereum blockchain exists on network ID 1, with genesis hash `d4e56740...b1cb8fa3` and its called the *Mainnet*. There are several testnets that developers can use to test their application functionality before going live. `STATUS` information is used by nodes to determine which peers they should connect to. If a node encounters an Ethereum peer that is on a different Ethereum network or genesis hash, it will disconnect from that peer. Peers who remain connected after `STATUS` message exchange utilize two `STATUS` message fields to coordinate blockchain synchronization: the hash of the most recent block known to a node (the best hash) and the total difficulty of its blockchain.

2.3 Known Attacks to Blockchain Systems

In this section, we review the most prominent attacks that have been identified by the research community against cryptocurrencies and blockchain systems. The usual aim of these attacks is to gain monetary value or attack user anonymity. The attacks described below can be targeted against a particular node/user in the network or against the whole network depending on the adversaries capabilities and intentions. An adversary may choose to combine several types of attack to amplify or facilitate his goal. Some techniques can achieve multiple goals or can be used as an intermediate step to complete an attack.

Understanding these attack vectors is crucial, as they underscore the significance of a robust and resilient underlying P2P network. Deficiencies in the P2P network can lead to vulnerabilities that can be exploited by malicious actors to execute these attacks. By examining these attack vectors, we aim to emphasize the essential role of a secure network structure in protecting the overall integrity and functionality of blockchain ecosystems.

2.3.1 Double Spending

The aim of this type of attack is to spend a coin twice. A common scenario is to trick a merchant to accept a transaction in exchange of services or goods while having a hidden double-spending transaction (typically sending the coins back to the adversary) accepted by the rest of the network and mined into the blockchain rendering the payment invalid.

Double spending is trivial if an attacker has sufficient mining power to produce blocks at will (also known as a 51% attack), but this is usually not the case. A more practical approach is to manipulate the merchant's view of transactions and blocks until he releases the goods. The attacker can then reveal his hidden transaction, which is already mined. In order to manipulate a node's view, other attacks may be employed, like network partitioning or node eclipsing.

2.3.2 Unfair Revenue

In the PoW consensus model, the revenue of the miners is directly related to the number of blocks they can produce. This, in turn, is proportional to their computing capability (hashing power). In other words, the more computing power is dedicated to create blocks, the more coins are earned. However, it has been shown that miners or mining pools can gain an advantage and produce blocks disproportional to their hashing power [29]. For that purpose, a dishonest miner can follow different strategies, commonly known as *Selfish Mining*. Unfair revenue attacks can be amplified or facilitated by attacking the network to induce delays or by employing a DoS attack against a competing miner or mining pool. It has also been demonstrated how network connectivity can enhance a miner's ability to perform such attacks.

2.3.3 Selfish Mining

While the protocol requires nodes to quickly distribute newly created blocks, nodes with enough hashing capabilities can, in fact, gain higher payoffs by withholding blocks they create and selectively postponing their publication. The key insight behind this selfish mining strategy is to force honest miners to perform wasted computations on the stale public branch. Specifically, selfish mining forces honest miners to spend their cycles on blocks that are destined to not be part of the blockchain. Depending on the miner's connectivity, he can enjoy disproportional gains by deviating from the protocol [29, 92]. Delay attacks and other network manipulation attacks can amplify the effect of selfish mining.

2.3.4 Partitioning Attacks

The goal of a partition attack is to completely disconnect a set of nodes from the network. This requires the attacker to divert and cut all the connections between the set of nodes and the rest of the network.

A malicious party can use a network partition to attack the consensus layer and perform a 51% attack, with less than 50% hash power, by stalling the transmission of blocks and waste the mining efforts of other miners. In addition, an attacker can perform a selfish mining attack or a double-spending attack against partitioned parties. Furthermore, prolonged delays in block propagation on the network cam have detrimental effects by increasing the fork rate, as explained in[22]. A powerful adversary, such as a nation-state attacker or a Tier-1 ISP, may even aim to disrupt a large portion of the underlying P2P network of a cryptocurrency. Moreover, irrational attacks with no obvious benefits are possible, as an adversary can find ways to ”bet against” the security of a cryptocurrency in other financially connected systems [33]. In a less destructive case, the adversary can arbitrarily censor selected transactions from being processed by the network.

2.3.5 Delay attacks

The purpose of a delay attack is to slow down the transmission of blocks to or from a specific group of nodes. Unlike partition attacks, which require a perfect cut, delay attacks can be effective even when only a subset of connections is intercepted. This means that attackers can perform delay attacks on connections they are naturally intercepting, making them more difficult to detect. In a delay attack, the attacker aims to prevent a target from receiving a transaction or block for a certain amount of time, without disrupting its connection.

2.3.6 Eclipse Attacks

Against Consensus. An eclipse attack is described as the situation in which all peer connections of a node are monopolized by an adversary. In such a case, the node’s access to information is under the control of an attacker who can therefore deceive the node, or even take control of the node’s mining power to his own benefit. Research works demonstrate how eclipse attacks can be used in favor of an adversary to perform double spending, selfish mining, or attack the consensus algorithm[43, 44, 54, 77].

Against layer-2 protocols Layer-two protocols avoid disseminating every transaction to the entire network by exchanging authenticated transactions off-chain. Peers transact between them, via *payment channels*, using fast intermediaries without recording transactions on the slow-advancing blockchain. The blockchain is only used to settle the final balances after a predefined period or in the event of disputes. The promise of layer-two protocols is to complete off-chain transactions in sub-seconds rather than minutes or hours while retaining asset security, reducing fees, and allowing blockchains to scale [50]. Examples are Bitcoin’s

Lightning Network and Ethereum's Raiden.[13][14] In such a setting, off-chain transactions are not allowed after the channel has closed. An attacker can trick an eclipsed victim into thinking the payment channel is still open, while the rest of the network sees the channel as closed. If the victim, a merchant, releases goods in exchange for off-chain payment, then the attacker can obtain the goods without paying.

2.4 Importance of the Network overlay

Network's ultimate goal is data availability and message propagation. It exists to ensure fast delivery of blocks and guarantee that transactions will reach all peers in a timely manner. Contrary to established content sharing P2P networks it is not the network's purpose to ensure that historical data is available. This is the responsibility of the peers, who maintain a copy of all transactions and blocks locally.

Consequently, availability of the P2P overlay is pertinent in order to facilitate the blockchain system's data propagation. Moreover, network vulnerabilities can be leveraged by adversaries in order to carry out various attacks against blockchains' consensus and fairness. In previous paragraphs, we briefly described implications of such network-level deficiencies.

[13]Thaddeus Dryja Joseph Poon. *The Bitcoin Lightning Network:Scalable Off-Chain Instant Payments*. 2016. URL: `https://lightning.network/lightning-network-paper.pdf`.

[14]Raiden Network. *What is the Raiden Network?* 2017. URL: `https://raiden.network/101.html`.

Chapter 3

Literature Review

Following the Bitcoin paper [144] in 2009 and the subsequent widespread implementation and adoption of the technology, a substantial body of research has emerged around the blockchain phenomenon. In this chapter, we conduct a comprehensive review of the literature of relevant previous research. Our selection criteria prioritize works of notable significance and recognition, contributing to a refined overview of the field. Specifically, our focus is directed towards investigating the network layer within the context of blockchain systems. Throughout this review, it becomes evident that the majority of related research has primarily focused on the examination of two of the most prominent blockchain networks. Although Bitcoin and Ethereum networks have received substantial attention, only a limited number of research papers have ventured into the exploration of other cryptocurrencies. Arguably, aspects of the network layer of blockchain systems have received much less attention from the research community [34, 27].

Before we give more details on each work, we present a summary of our knowledge today.

- Network measurements have revealed that propagation delay is a critical parameter positively correlated with the appearance of blockchain forks [22]. However, Bitcoin's network infrastructure shows signs of improvement [30].

- Major cryptocurrencies face centralization issues. A large fraction of reachable nodes are located in a handful of Autonomous Systems (AS). This opens the door for adversaries to mount network attacks at the Internet level by hijacking the BGP protocol [6]. Such attacks can isolate a large group of nodes from the rest of the network and introduce delays in message propagation. In fact, such attacks are becoming more sophisticated and are not easily detected [97]. Furthermore, less than five mining pools control the majority of the hashing power. By combining knowledge of network topology and message broadcasting, researchers were able to identify highly influential nodes that have an advantage in block production and dissemination, strengthening centralization indications.

- A large percentage of nodes participating in the Bitcoin network are unreachable, making it difficult to accurately analyze their behavior and characteristics. However, research work has shown that these unreachable nodes still play a significant role in the network, as they propagate a large number of transactions and initiate a small number of connections to the reachable part of the network [101]. The number of unreachable nodes is estimated to be between 10x and 100x the number of reachable nodes. These nodes have been found to have less secure wallets and initiate fewer connections to the reachable part of the network than the default bitcoin client. In general, understanding the behavior and characteristics of unreachable nodes in the Bitcoin network is important to improve our understanding of the network as a whole.

- Accurate topology inference of blockchain peer-to-peer (P2P) network overlays is a complex challenge that remains unsolved. While some research has successfully devised methods to uncover the network topology, these techniques are often outdated due to changes in official client protocols. Additionally, only a limited number of studies have extended their analysis to compute network metrics, which offer valuable information on network conditions. In essence, accurately inferring P2P network overlay topology remains a persistent challenge, necessitating further research for accurate mapping and comprehensive characterization.

- Numerous attack vectors, or methods that can compromise blockchain systems, have been proposed and analyzed in the literature. Review articles have analyzed these attack vectors, highlighting how network attacks can be related to other types of attack and how the state of the network can facilitate the success of an attack. These reviews provide important information on the various ways networks can be targeted and the factors that can increase the probability of a successful attack [34]. Understanding these attack vectors and their relationships to network conditions is crucial for developing effective defenses and countermeasures.

We begin by highlighting significant surveys that have contributed to the understanding of blockchain research. These surveys lay the foundation for our exploration of related works in the following sections. For each specific topic, we present relevant research papers in their respective sections.

Key Surveys in Blockchain Research For researchers seeking a comprehensive understanding of blockchain systems, several significant sources offer invaluable insights. Delgado provides an extensive classification of bitcoin nodes, shedding light on the distinctions of cryptocurrency P2P networks compared to existing implementations [23]. In [34], Franzoni and Daza explore various attacks, emphasizing their correlation with network dynamics. Dotan *et al.* [27] recognize that blockchain overlay networks have different requirements than traditional communication networks and observe that the fundamental design aspects of these networks are not well understood. Their work identifies differences and commonalities between blockchains and traditional networks and highlights open research challenges in network design for distributed decentralized systems. Bonneau *et al.* [11] deliver an in-depth introduction to the essential components of blockchain systems, unveiling their intricacies, along with discovered attacks, potential alternative approaches, and challenges. Additional influential surveys are offered by Natoli [76] and Tschorsch [98], providing further critical insight into the realm of blockchain research.

Garay *et al.* provide on of the first formal analysis of Bitcoin's consensus protocol [39]. In their analysis, they identified three properties required for the correct function of Bitcoin and showed that these are satisfied in practice. However, they also proved that the protocol does not satisfy the Byzantine agreement problem. In [38], Garay and Kiayias provide a detailed review of the consensus problem and bridge classical approaches with modern implementations in the blockchain era. Importantly, they highlight the difference that in the blockchain setting the communication resource is a P2P network, a much weaker communication medium than point-to-point channels, which is often the case in the classical approach.

Croman *et al.* [20] enumerate a number of ideas to extend the scalability of blockchain frameworks. Among others, they observed that the network layer is a bottleneck to Bitcoin's throughput, and the available bandwidth is underutilized. They also highlight the need for robust network protocols that can maintain strong connectivity between honest nodes in a highly dynamic environment.

3.1 Decentralization

Arguably, one of the most prominent features of blockchain frameworks is decentralization. Motivated by this property, Gencer*et al.* [41] tried to quantify decentralization by defining appropriate metrics. Their measurements focused on the Bitcoin and Ethereum networks. Among others, they calculated the provisioned bandwidth and the geographical distance between nodes. They reported that Bitcoin nodes were more clustered in terms of geographical distance, while Ethereum nodes were more dispersed. In addition, they also confirmed the existence of centralization trends in both networks. Gencer's results showed that both platforms rely on very few mining entities (less than 20) to maintain the blockchain, meaning that a centralization of computing power exists. Two years later, in 2019, Mariem *et al.* [10] made similar measurements and observed even greater centralization of the mining power. Within two years, four mining pools were responsible for half of the blocks produced. This phenomenon was also examined by Gervais *et al.* in [42] and later by Sarada in [45], showing that both networks tend to become more centralized as time passes. A taxonomy of centralization metrics and categories can be found in the works of Sai [90] and Karakostas [59].

3.2 Network Measurements

3.2.1 Measuring the Bitcoin network

In 2013, Decker and Wattenhofer [22] conducted one of the first measurement studies on the Bitcoin network to determine how the propagation of information affects consensus and stability. They analyzed the block propagation time for 10,000 blocks and discovered that it followed an exponential curve. Although the median and average values were low (less than 13 seconds), the distribution showed a long tail, with 5% nodes requiring more than 40 seconds to receive the blocks. Importantly, they found that the main reason that affected the fork rate (i.e., competing blocks) was the propagation delays experienced between peers in the network.

In a similar measurement study, Feld *et al.* [31] traversed Bitcoin's P2P network and collected information on the size of the network and the distribution of peers among autonomous systems (AS). Their results indicated a high centralization of the nodes. Most peers were found to reside in the same ASes, which leaves Bitcoin's resilience and security flawed, since forks and double spending attacks can be facilitated by taking AS information into account. Specifically, 30% of the discovered peers were located in only 10 ASes. Furthermore, they discovered a large fraction (98.5%) of unreachable IP addresses. A similar measurement was carried out by Donet *et al.* [26], corroborating previous evidence.

A more recent work carried out in 2020 by Fechner*et al.* [30], repeated the same experiment with Decker [22], taking into account changes in the protocol. Fechner expanded their measurements to other networks as well, namely BitcoinCash, Litecoin, and Dogecoin. Their results show that propagation delays and fork rates have been significantly reduced. The main reason for this reduction is attributed to the better infrastructure that is now available and the introduction of compact blocks. Furthermore, they observe that transaction propagation delays are significantly longer than block propagation. A new finding by Fechner is that there exists a high centralization in the locality of blockchain nodes, particularly by state-owned network providers. This result, combined with sudden changes in laws, such as the Chinese government banning cryptocurrency mining in 2021, can significantly affect network stability. Similar trends were observed in all networks in their study.

Another factor related to propagation delays is churn. Imtiaz *et al.* [56] report that the significant majority (97%) of Bitcoin nodes are intermittently connected. This results in an increased number of message exchanges, roughly twice the figure for continuously connected nodes. In particular, they experimentally demonstrate that this rate of churn leads to a 135% average increase in block propagation time and up to 800% in the worst case.

Neudecker studied various characteristics of the Bitcoin network, such as the number of peers, propagation delays, client versions, and distribution of nodes in ASes in a long-term study [78]. His work makes the case for an always-on network monitor capable of capturing abnormal events or sudden changes in the state of the network. For example, Neudecker observed short periods with a rapid increase in the number of connections originating from the same set of IP addresses (Sybil nodes). Neudecker argues that constant monitoring of the network is essential for the health of the system.

Estimating degree distribution A spam wave of fake IP addresses appeared on the Bitcoin network during the summer of 2021. Grundmann *et al.* [47] used this event to estimate the degrees of reachable nodes in the Bitcoin network, using an idea previously suggested by Wang and Pustogarov in [101]. The intuition behind this method is based on the official implementation of the bitcoin client. When receiving an `addr` message, the receiving node (denoted as A) will propagate the IP addresses in it, to a fraction of its connected peers. A monitoring node (M) connected to A, will receive a number of IP addresses that can be approximated by $\frac{2}{k-1}$ when the number of IP addresses to be relayed is large enough. The parameter k is equal to the degree of node A, i.e., the number of peers connected to A. Using this knowledge, node M can estimate the number k, i.e., the degree of node A. In fact, M can use the same methodology to discover all of his peers' degrees. In interpreting the data, Grundmann observed that half of reachable nodes have the maximum number of connections (125), according to the client's defaults. This could be worrying for the openness of the network, since this means that most clients will not be able to accept connections from "new-coming" nodes. The work also estimates that there are 18 "supernodes" (nodes connected to all reachable peers) in the network. In addition, based on the data collected, Grundmann estimated the number of unreachable peers that could exist on the network and found that it is close to 32000.

Unreachable peers The role of unreachable peers in the Bitcoin P2P network is studied to a limited extent. To improve our understanding of this part of the Bitcoin system, Wang and Pustogarov conducted a large-scale network experiment in which they deployed multiple probing nodes for a week [101]. These nodes were used to study unreachable peers in the Bitcoin P2P network, which play a role in the dissemination of blocks and transactions, but are difficult to identify and quantify due to their unreachable nature. For each peer connected to one of their probes, they tested whether the peer was reachable. They observed an average of 10,000 unreachable addresses in a six-hour window and estimated that there are at least 155,000 unreachable nodes in each interval. Grundmann calculations are more conservative, suggesting the existence of 35,000 unreachable nodes [49].

3.2.2 Measuring the Ethereum network

Kim *et al.* [62] made one of the first measurements in the Ethereum network. In their work, they deploy a number of highly connected nodes on the Ethereum network to enumerate all participating peers. They find that Ethereum's network is cluttered by various different blockchains; almost half of the network peers are not synchronized on the main chain. In their work, they do not identify any relationships among the discovered nodes and, therefore, do not analyze the network topology.

Wang *et al.* developed Ethna [102], a tool that measures the degree distribution of Ethereum nodes. Ethna measures the number of transaction messages sent by each node. Taking into account the randomness of Ethereum's message forwarding protocol and assuming that this randomness is closely related to the actual connections of the nodes, Ethna calculates the degrees of the nodes that are accurate enough to reflect the characteristics of the network topology. This assumption was verified experimentally. Their results indicate that the degree distribution in the Ethereum network follows a power-law, although they report high p-values in their validation (cf. [102] Section VA2). Also, the network appears to be highly dynamic since, for the two captured snapshots, they calculate different power-law exponents.

Kiffer *et al.* also study the Etherum network [61]. The main novelty of this work is the combination of peer connectivity observations with the block information provided by the peer. In their 2019 measurements, Kiffer *et al.* observed a significant churn in the network, with more than 45% peers staying connected only for up to 10 seconds per connection. Furthermore, they found that not all of these peers actually contribute to block propagation, and only about 27% of the discovered peers are useful.

3.2.3 Measuring other blockchain networks

Similarly to the works mentioned above, Daniel *et al.* map the ZCash network [21]. They report on the size of the network and the geographical distribution of nodes. They also derived insights into centralization and presented an inference method based on a block arrival time analysis, similar to the work of Neudecker in [80] (see Section 3.4). Their results suggest a high centralization of the mining power, which is consistent with the observations made on the Ethereum and Bitcoin networks [41, 45]. Although the suggested topology inference method had adequate precision, the authors did not provide any information on the characteristics of the network graph. Finally, they observe a high number of stable long-lived peers, in contrast to the high churn observed in other blockchain networks [61, 56].

Javarone *et al.*, used the Bitcoin discovery mechanism, and performed a measurement that compared Bitcoin with the BitcoinCash network [58]. They suggested that the node degrees of both networks *may* follow a power-law distribution, but did not provide a goodness-of-fit test.

3.3 Attacks

3.3.1 Eclipse attacks

Heilman *et al.* presented an eclipse attack on the Bitcoin P2P network [54]. They demonstrate how an adversary controlling a large number of IP addresses can monopolize all connections to and from a victim node and exploit it to mount attacks on Bitcoin's mining and consensus. In their devised attack, the victim's node `ADDRMAN` is filled with: a) IP addresses controlled by the adversary and b) "trash" or unused IP addresses. These trash addresses are IANA unallocated IPv4 addresses, so any connections to them will fail.

Addresses controlled by the adversary initiate connections to the victim node. According to the protocol (before version 0.10.0), incoming connections, if reachable, will be inserted into the `tried` table of `ADDRMAN`, i.e., newer addresses are preferred to older ones (cf. Section 2.1). Then, the *new* table is filled by sending `addr` messages with trash addresses. This leaves the victim node with a large number of adversarial addresses in its `tried` table and a bunch of unused addresses in its `new` table.

The next step of the attack is to wait (or force) the node to restart. Upon restart, the node will choose some peer addresses from `ADDRMAN`, which is already poisoned with adversarial and trash addresses. Thus, the victim node will connect to adversarial nodes and will effectively be eclipsed from the rest of the network. Heilman calculated that a successful attack required fewer than 6000 addresses, which could easily be achieved by using a botnet. Furthermore, real-world experiments showed that most attacks had a very high success rate.

Yang *et al.* [107] has also proposed another eclipse attack, based on BGP-hijacking. The attacker does not disconnect the victim from legitimate peers but instead changes the content of exchanged messages. This tactic is feasible due to the plaintext transmission of bitcoin messages. The attacker will change all `addr` messages directed at the victim node and replace the entries with malicious IP addresses under his control. During the attack, the victim node can communicate normally, making the attack difficult to detect. In their work, an attacker was shown to fill the `ADDRMAN` of the victim in less than forty minutes.

Mitigation of eclipse attacks in Bitcoin Following the publication of the eclipse attack by Heilman, bitcoin core developers implemented his suggested mitigation mechanisms in the official bitcoin client[1]. Older addresses are not evicted in favor of fresher addresses unless they do not respond. Feeler connections were introduced to eliminate unused addresses in the `new` table and increase valid entries in the `tried` set. Furthermore, to avoid connecting to malicious peers after restarting, current outgoing connections are stored in an *anchor* table. Since it is more difficult for an attacker to disrupt these (already existing) connections, *anchor* nodes are preferred when a node restarts. Finally, the number of buckets was increased to further enhance the complexity of the attack.

Ethereum Eclipse attacks against the Ethereum network have been demonstrated by Marcus *et al.* in [71]. In contrast to Bitcoin, Ethereum uses a more secure transport protocol with cryptographically authenticated messages. This makes Ethereum more robust against man-in-the-middle attacks and attacks that manipulate the BGP routing protocol [6]. Marcus *et al.* observed that Ethereum's protocol distinguishes peers by their ENODEID (see Section 2.2.1), an ECDSA public key, and does not consider the peer's IP address. This allows an attacker to craft various identities from the same machine and, by abusing the protocol, fill a victim's node DHT table with his own set of addresses. Furthermore, Ethereum's peer selection is biased (making some ENODEIDs more likely to be chosen) and can be easily predicted by an attacker. Therefore, by carefully selecting node IDs, an attacker will be in an advantageous position against other legitimate peers.

Henningsen *et al.* [55] introduced another type of eclipse attack against Ethereum. Similarly to Marcus [71], Henningsen takes advantage of the properties of the peer selection mechanism to achieve the same result with only one Sybil node per neighbor table bucket. In addition, they observe that there is no need to forcefully restart the victim node, since connections on the Ethereum network are rather short-lived. Thus, it is possible to mount an eclipse attack against a non-restarting victim node, in a matter of days.

Following the suggestions of [71, 55], Ethereum developers incorporated appropriate changes in their client software, which mitigate these attacks.

3.3.2 Partitioning

The feasibility of a partitioning attack was studied by Neudecker in [79], through a simulated analysis. Neudecker's method was to simulate a Denial-of-Service attack to the minimum

[1]Pieter Wuille. *Countermeasures against eclipse attacks.* 2015. URL: `https://github.com/bitcoin/bitcoin/pull/5941`.

vertex cut of the network. This, of course, implies knowledge of the underlying topology, and currently this is mitigated by topology-hiding techniques implemented in Bitcoin's protocol. Neudecker highlights that topology hiding is required to achieve network security and robustness.

Apostolaki presented an AS-level partitioning attack in [6]. The attack is based on vulnerabilities of the BGP routing protocol and the fact that a majority of Bitcoin clients reside in a small number of Autonomous Systems. More precisely, Apostolaki presented an eclipse attack at the BGP level that can be easily scaled to partition the network due to the centralization of nodes at the AS level. Apostolaki showed that an AS-controlling adversary would be able to isolate almost half of the mining power in Bitcoin by hijacking less than 100 prefixes in less than 2 minutes. Currently, the attack is mitigated by diversifying node connections based on their BGP route, when this information is available. More recently, Tran *et al.* [97] proposed a stealthier version of a partitioning attack. His work also utilized the BGP protocol.

Saad in [88] explores various partitioning attacks against the Bitcoin network. In a related work, [89] Saad introduces a temporal attack, taking advantage of the long tail of block propagation times observed in the network. By carefully measuring the information flow in the network, Saad located vulnerable points in time when up to 90% of the network is 1-4 blocks behind. Furthermore, the work showed how these vulnerable spots appear systematically. By identifying nodes that were lagging simultaneously, an attacker could isolate them and mislead them with false blocks.

3.4 Blockchain Topology Inference

Miller *et al.* [73] were the first to devise a method that could discover peer-to-peer links in the Bitcoin network. Their measurement method was based on the timestamps included in the `addr` messages. The role of the timestamp was to ensure that terminated nodes were not propagated on the network. Before this work, the Bitcoin client updated the timestamp corresponding to each peer in its `ADDRMAN`, in the following way:

- For outbound connections, the timestamp was updated each time the node received a message from the peer.
- For inbound connections, the timestamp was set to when the connection was established, but was not updated on each message exchange.

- For all other peers, learned from `addr` messages, the timestamp was the time of reception minus two hours.

With this in mind, issuing `getaddr` messages to all nodes in the network and analyzing the timestamps could reveal the network connection map. Having the topological information of the network, Miller analyzed its characteristics. Most of the nodes had a degree between 8 and 12, which is close to the client's defaults (8 outbound connections plus any inbound). The maximum number of connections allowed by the client is 125. However, Miller observed a number of nodes with significantly higher node degree, close to 10,000. The authors believed that these nodes were part of a coordinated effort to measure the network. Furthermore, using the Louvain algorithm for community detection, Miller came to the conclusion that the Bitcoin network is not random; this was an unexpected result, as the network was supposed to be similar to a random graph [22]. Miller also observed a rather stable network with few variations in its structure during the measurement period. Finally, Miller identified a small set of nodes (approximately 2%) that were more influential than the rest of the network, accounting for more than 75% of the total hash power. Interestingly, these nodes cannot be identified by their network characteristics or connection uptime.

After Miller's publication, the official bitcoin client was updated to discourage similar attempts at topology inference [124].[2] Specifically, the procedure for updating a peer's timestamp in `ADDRMAN` has changed in a way that cannot be used to map the network topology. Timestamps are now updated only during disconnections.

Neudecker *et al.* [80] performed a timing analysis of the propagation of transactions to infer the network topology. A highly connected monitor logs the reception time of each transaction from all connected nodes. The intuition behind Neudecker's method is that reception times of the same message (transaction) correlate with the network topology. The proposed timing analysis infers the number of hops a message has traveled through the network by comparing the observed delay to the delay expected by several path lengths. This approach requires that the monitoring node is connected to the message's originator (i.e., the peer that first created a transaction). If this is not possible, the monitoring node must actively create transactions and send them to the network peers. Although Neudecker's model had sufficient precision, their published work did not include any information about the actual network and its characteristics. Additionally, updates to the official bitcoin client's propagation mechanism

[2]Jonas Nick. *Guessing Bitcoin's P2P Connections*. 2015. URL: `https://jonasnick.github.io/blog/2015/03/06/guessing-bitcoins-p2p-connections/`.

have made Neudecker's approach impractical due to the introduction of random delays in the transaction relay process. These changes have effectively rendered Neudecker's methods ineffective [155].

Taking into account the limitations introduced by protocol changes, Delgado *et al.* introduced *TxProbe*, suggesting a new method to explore the Bitcoin network [24]. Their method is based on the way bitcoin clients propagate transactions specifically out-of-order, or *orphaned*, transactions. As described in Section 2.1, transaction propagation occurs in a three-round protocol. A Bitcoin node validates each transaction it receives before relaying it to its peers.

A valid transaction must have correct signatures and must only spend existing and currently unspent coins. Otherwise, the transaction is deemed invalid and the transaction is discarded. To aid in validation, each Bitcoin full-node maintains a view of the current set of available coins, called *Unspent Transaction Outputs* or the UTXO set. Nodes also maintain a collection of pending transactions in the `MEMPOOL` data structure. These are validated transactions (without double spends) that have not yet been included in a block. Sometimes transactions are received out-of-order; they spend coins that have not yet been created or are not in the `MEMPOOL` (i.e., the transactions spent an output that the node has not seen yet). These are called *orphaned* transactions and are stored in a separate data structure, the `MapOrphanTransactions` pool. When the parent transaction of an orphan arrives, it can be validated without re-requesting it from the network. Importantly, orphaned transactions are not requested using texttttGETDATA messages, and TxProbe exploits exactly this behavior. Briefly, TxProbe operates in the following steps:

1. A pair of transactions that spend the same coin (double spending) is created; tx_P is referred to as the parent, and tx_F as the flooding transaction.
2. tx_P is sent to node A and tx_F to node B. It is assumed that both transactions reach their destination at the same time, so A will reject tx_F if B sends it to him and vice versa.
3. A marker transaction tx_M that spends an output of tx_P is sent to A.
4. There are two possible cases: either A and B are connected, or they are not. If connected, according to protocol rules, tx_M will be relayed to B, and since tx_M's parent is tx_P (which B does not know), tx_M will go into the `MapOrphanTransactions` pool. If A and B are not connected, tx_M will never reach node B.
5. Finally, node B receives an `INV` message containing tx_M's hash. If A and B are connected, tx_M will be in B's orphan transactions pool, thus node B will not send a

GETDATA message. If B responds with a GETDATA message, we can infer that the two nodes are not connected.

This sketches the intuition behind TxProbe's logic. However, the solution suggested is more complicated since the presence of a third node would make things even more complicated. To overcome this limitation, TxProbe blinds all other nodes using the *invblock* technique, suggested by MIller [73]. Recall that a node that requests a transaction with a GETDATA message in response to an INV message will wait up to two minutes before requesting the same transaction from any other peer (cf. Section.2.1).

Since this method can interfere with ordinary transactions, Segura has only performed measurements in Bitcoin's testnet and not in the main net. The collected results indicate that Bitcoin's testnet, similar to the main net measured by Miller[73], does not resemble a random graph. Segura also provided insights into the network structure by calculating graph metrics such as the network's diameter, clustering coefficient, assortativity, and more. TxProbe is an effective method, but it comes with some limitations. Transactions that are created and propagated through the network incur a cost, namely, the transaction fee. Using this method with current Bitcoin prices and estimating 10,000 reachable bitcoin nodes would require more than 113,700(USD) in transaction fees[3] and more than 12 hours to cover the entire network. Finally, this method is no longer accurate enough in practice due to protocol changes, as reported in [34].

Grundmann *et al.* [48] explored two mechanisms for inferring the topology of Bitcoin's network. Their first approach exploits the accumulation of multiple transactions before their announcement to other peers. After Neudecker's timing analysis method [80], INV messages are not sent immediately after receiving and validating a transaction, but are randomly delayed. Bitcoin's client maintains one outgoing queue for each connected peer, and each queue contains all transactions that are to be announced to the respective peer. After some time, all messages in a queue are announced to the connected peer with a single INV message. To infer connections between peers, a monitoring node creates one transaction for each reachable peer. For example, tx_A corresponds to node A, tx_B to node B, and so forth. All transactions are sent simultaneously, and each peer receives his respective transaction. The monitoring node then waits for the *INV* announcements from each peer. The topology is inferred using the following rules:

1. If the INV message that peer A sends to the monitor contains only tx_B and no other from

[3]Bitcoin Fees. *bitcoinfees.earn.com.* 2021. URL: https://bitcoinfees.earn.com.

the transactions created, then A and B are connected.

2. If the `INV` message that peer A sends to the monitor contains more than one transaction, at least one of the peers associated with the announced transactions is connected to A.

Unfortunately, this work introduces a large number of false positives and the authors admit that it is impractical to carry out in the real world.

The second approach suggested by Grundmann exploits the fact that clients drop double spending transactions. This method aims to reveal the connections of a target node, but with enough resources, it can scale to cover the entire network. A monitoring node M can infer the connections of node T as follows. Assume that M is connected to all reachable nodes. A transaction is created for each peer, except for the target node T. Furthermore, all transactions are double spends of the same parent transaction, but unique (e.g., each spends a different output of the parent transaction). Similarly to the previous method, each node is assigned a transaction, and all are propagated at the same time. The monitoring node then waits for the transaction that the target node will announce and can conclude that the peer associated with the forwarded transaction is directly connected to the target peer. This procedure can reveal only one of the target node's connections, so it has to be repeated. Grundmann also suggested a variation to avoid repeated inferences of the same connection. The approach was validated and found to have very good accuracy. Since the method suggested by Grandmann involves transactions, it also comes at a cost. Due to ethical reasons and high transaction fees, their validation was performed against Bitcoin's testnet. Additionally, no data on the characteristics of the network were reported. As a final note, Grundmann required 99 rounds to reveal all connections of a node with significant accuracy, but does not mention how many connections were found. As such, we cannot calculate the exact cost of the proposed method. For context, assuming a network of 10,000 nodes and 99 rounds required per node, with current fee estimates, the total cost to infer the network topology would be close to 100,000(USD).

Ethereum Li*et al.* introduced TopoShot [67], a method that exploits transaction manipulation in Ethereum clients to measure the network topology. TopoShot's main idea is similar to Grundmann's [48], repurposed to Ethereum. The intuition behind it is as follows. Unconfirmed Ethereum transactions (that have not been included in a block yet) can be replaced or evicted from a node's `MEMPOOL` by a subsequent transaction with a higher transaction fee. TopoShot runs a measurement node M to detect a connection between two remote peers, A and B. Node M first propagates a medium-priced transaction tx_C to all reachable nodes in the network,

Table 3.1: Topology inference proposed solutions

Network	Reference	Cost(USD)	Duration(h)	Mitigation
Bitcoin	Coinscope [73]	/	4	Address timestamps are updated less frequently
	TxProbe [24]	113,000	8.3	Invblock technique mitigated. Orphan pool eviction changed.
	Double-spending transactions [48]	96,930	20	
	Timing-Analysis [80]	/	/	Random delays introduced into tx propagation
Ethereum	TopoShot [67]	60,000,000	approx. 12	

including nodes A and B. Then it floods the `MEMPOOL` of node B, effectively removing tx_C. Subsequently, it sends a low-priced transaction tx_B to the target node B and a high-priced transaction tx_A to the target node A. TopoShot then observes whether tx_A is present in node B and if so, draws the conclusion that the two nodes are actively connected. The purpose of tx_C is to isolate the rest of the network from the two nodes. Since TopoShot relies on fabricating and propagating transactions in the network, it incurs costs due to transaction fees. For this reason, TopoShot was not deployed on Ethereum's main net. In fact, TopoShot's authors acknowledge that measuring the entire main net would cost 60 million USD at Ethereum prices as of May 2021. However, Li *et al.* calculated the network properties based on the topology discovered in the Ethereum test net.

3.4.1 Threats imposed by topology inference

Topology inference, or the process of determining the structure of a network, is not necessarily considered an attack in and of itself. However, having knowledge of the topology of a network can give a potential attacker a significant advantage in launching a variety of attacks, such as causing partitioning, eclipsing, and attacks against user anonymity. These attacks can target specific users or a significant part of the network. As a result, information that could be used to discover the topology of a blockchain network is carefully protected and deliberately hidden through various techniques. Previous research, not related to blockchains [105], has shown that hiding even a small fraction of nodes can be an effective and cost-effective strategy to improve the robustness of complex networks against topology inference attacks. The most important methods to infer the topology of the Bitcoin and Ethereum networks, as well as their cost and mitigations, are summarized in Table 3.1.

3.5 Protocol improvements

Franzoni *et al.*, argue that topology hiding techniques hinder the detection of structural problems that affect security and efficiency [35]. They show that topology obfuscation is not a sufficient measure against network-layer attacks. In their work, they propose an alternative network protocol, highlighting that the benefits of an open topology outweigh its risks. The conclusions derived from this thesis agree with this statement.

After conducting a measurement study by collecting nodes' ip addresses, Park*et al.* conclude that nodes are typically located in various geographical locations and hence experience delays when exchanging information [84]. Park suggests a protocol change on how Bitcoin clients should choose peers. The proposed protocol selects peers according to geolocation distance to improve propagation delays.

3.6 Discussion

There is a lack of knowledge about the structure and topological properties of blockchain overlay networks, as past research has mainly focused on techniques to deduce the hidden topology of Bitcoin or Ethereum overlays, without providing any network metrics or characteristics. These studies were validated against the testnet [24, 67] or against selected nodes [80, 48], with the exception of Coinscope [73]. However, many of the suggested methods for topology inference are no longer applicable due to changes in reference clients, infeasible due to transaction fees, or impractical to use on multiple blockchain networks because they require maintaining connections to a high number of peers. Furthermore, Bitcoin core developers are constantly updating the protocol to prevent any leakage of information that could facilitate topology inference.

The main objective of this research is to examine the underlying network of blockchain systems in order to identify possible limitations and vulnerabilities. As more and more money and services are built on top of the P2P infrastructures and protocols that support blockchain systems, these become critical assets. However, they are largely undocumented and may be susceptible to severe attacks. Despite the distributed nature of P2P networks, their reliability cannot be assumed due to the possibility of biases (such as a concentration of nodes in a particular geographical location) and attacks (such as eclipse attacks, or partition attacks) that can disrupt the network.

In order to achieve our primary goal of analyzing the underlying network of blockchain systems, we must first obtain a topology mapping of the participating peers. To this end, our approach revolves around two axes.

First, we address the challenges posed by topology-hiding techniques by considering all potential connections that could exist within these networks at any given time. Second, we adopt a unified network modeling strategy. In particular, we do not attempt to categorize nodes based on their roles within the system, a common practice in blockchain network protocol simulators [5]. Furthermore, we treat both reachable and unreachable peers equivalently and we refrain from discriminating against links originating from unreachable peers. The inclusiveness of our unified approach not only streamlines the processing of collected data but also enhances the versatility of our method, making it applicable to diverse network architectures, or to networks with varying numbers of unreachable peers.

The following chapter outlines our proposed method for studying these networks and demonstrates its accuracy. We present a simple network monitoring tool capable of probing seven different blockchain networks simultaneously. This tool is practical and operates at low cost. We then proceed to assess the resilience of these networks against random and targeted attacks. To our knowledge, this is the first study to examine the structural properties of various blockchain networks.

Chapter 4

Research Methodology

To explore the resilience of blockchain P2P overlays, it is essential to understand the structure of the network. This chapter presents our main idea, which eliminates the requirement for a precise topology map of the network. We establish the validity of this approach and subsequently detail the methods employed for data collection and validation.

4.1 Measured overlay networks

This section introduces the seven blockchain networks that form the focus of our study. All of these networks have consistently ranked within the top 50 cryptocurrencies by market capitalization for several years, as reported by coinmarketcap.com.[1] Presented below in alphabetical order are the blockchain networks under consideration:

1. **Bitcoin** [144] was the first cryptographic currency to gain widespread adoption.

2. **Bitcoin Cash** is a hard fork of Bitcoin with an increased block size, with the aim of increasing transaction throughput and reducing clearance delays.[2,3]

3. **Dash** is another fork of Bitcoin. It employs a two-tiered network, consisting of mining nodes and master nodes, enabling very fast transaction confirmation times.[4]

[1]CoinMarketCap. *CoinMarketCap*. 2021. URL: `https://coinmarketcap.com`.

[2]BitcoinCash. *Bitcoin Cash*. 2021. URL: `https://www.bitcoincash.org`.

[3]Jimmy Song. *Bitcoin Cash: What You Need to Know*. 2017. URL: `https://jimmysong.medium.com/bitcoin-cash-what-you-need-to-know-c25df28995cf`.

[4]Daniel Diaz Evan Duffield. *Dash: A Payments-Focused Cryptocurrency*. 2018. URL: `https://github.`

4. **Dogecoin** [143] derived from Litecoin, features rapid block generation times and has gained a notable market capitalization.[5,6]

5. **Ethereum**[117], renowned for the execution of smart contracts in decentralized applications, has the second highest market capitalization.

6. **Litecoin**, the first Bitcoin fork, employs distinct features such as shorter block generation time and the Scrypt hashing algorithm [85].[7]

7. **Zcash** focusing on user privacy, utilizes zero-knowledge proofs for transaction anonymity.[8]

All blockchains mentioned above, except Ethereum, trace their origins back to Bitcoin, sharing similar overlay implementations [23]. The details of Bitcoin and Ethereum overlay network protocols are discussed in Chapter 2.

4.2 Main Idea

Topology inference in blockchain overlays is a challenging problem that has not yet been solved. Our approach is to solve a simpler problem while still being able to reason about the resilience of these networks. Instead of trying to accurately capture the existing connections between online nodes, which is almost impossible due to the design of blockchain networks (see Section 3.6 and Table 3.1), we focus on collecting all *possible* connections that may exist over a period of time. A connection between two nodes is considered possible if one node includes the other in its list of known addresses. Using this strategy, we trade accuracy for completeness and are able to synthesize connectivity graphs that include the vast majority of potential links between nodes. This method also captures actual connections, i.e., all active links between nodes. Our main goal is to identify structural deficiencies in the overlays, and we believe that if the synthesized graph of all possible connections is not resilient, then the actual realized topology of the overlay is unlikely to be resilient either.

com/dashpay/dash/wiki/Whitepaper.

[5]Ian Young. "Dogecoin: A Brief Overview & Survey". In: *SSRN* (2018). URL: http://dx.doi.org/10.2139/ssrn.3306060.

[6]Kevin Roose. *Is There a Cryptocurrency Bubble? Just Ask Doge.* 2017. URL: https://www.nytimes.com/2017/09/15/business/cryptocurrency-bubble-doge.html.

[7]Litecoin. *Litecoin.* 2021. URL: https://litecoin.org.

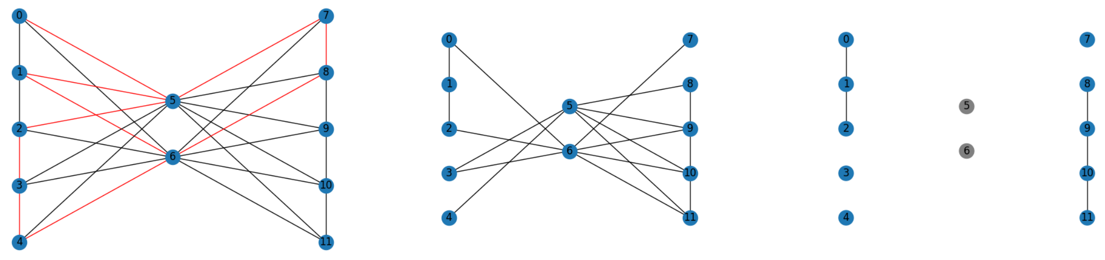
[8]Daira Hopwood et al. *Zcash Protocol Specification.* 2020. URL: https://coinpare.io/whitepaper/zcash.pdf.

In our data collection, we do not differentiate between mining nodes, light nodes, or full nodes. Additionally, our approach includes any unreachable peers discovered, which sets it apart from prior studies that have excluded these network elements [73, 24, 48]. We view all nodes as important contributors to the health of the ecosystem and vital in the dissemination of transactions and blocks. As mentioned previously (see Chapter 3), consensus is heavily based on correct and timely propagation of blocks [11]. The fact that miners may be reluctant to propagate blocks further emphasizes the importance of all other nodes in the system [8]. Transaction propagation is also crucial, as it is the primary purpose of the system and ensures its continued operation.

Proof To explain the intuition behind our reasoning, Figure 4.1 is used as an example. A synthesized graph would contain *more* or the same number of edges as the actual graph. These edges are highlighted in red in Fig. 4.1a. In this case, the actual topology would look like in Fig. 4.1b. In both graphs, removing the two middle nodes, colored gray in Fig. 4.1c, would partition the network. In other words, if a denser graph (synthesized graph of all possible links), including more edges, can be partitioned, a derived subgraph (actual graph) will always be partitioned as well. This holds as long as the address crawler does not miss any edges between the left and right groups, according to Fig. 4.1. In Section 4.4 we validate the proposed data collection method and show that the vast majority of potential links are captured.

To illustrate the logic behind our approach, we use Figure 4.1 as an example. A synthesized graph would contain the same or more edges than the actual graph, as shown in red in Figure 4.1a. In this case, the actual topology would look like Figure 4.1b. In both graphs, removing the two middle nodes, shown in gray in Figure 4.1c, would partition the network. In other words, if a denser graph (the synthesized graph with all possible links) can be partitioned, a derived subgraph (the actual graph) will also be partitioned as long as the address crawler does not miss any edges between the left and right groups, as shown in Figure 4.1. In Section 4.4, we validate the proposed data collection process and show that all potential links are captured.

A simple proof that the actual connection graph is not likely to be resilient when the synthesized one already is not, is provided here to further support our argument. A synthesized graph, G, consists of all possible connections that could exist in the network. In this case, the actual graph R, which contains only the real links (active links between nodes), would be a spanning subgraph of G. A spanning subgraph is a subgraph that contains all the vertices (nodes) of the original graph but not all the edges (links).



(a) Synthesized Graph **(b)** Actual subgraph **(c)** Partitioned praph

Figure 4.1: Resilience intuition: synthesized graphs and actual topology.

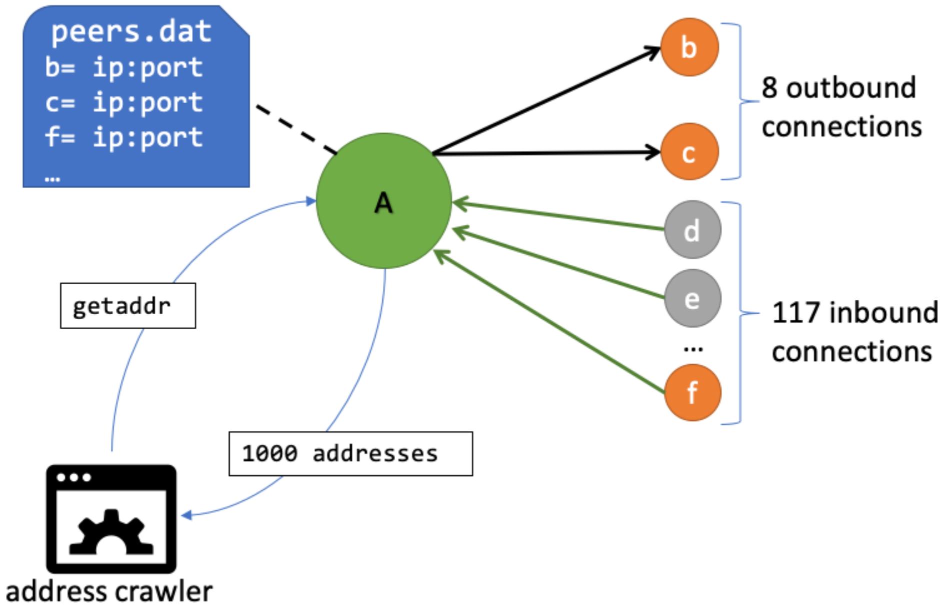


Figure 4.2: Node's connections.

Our proposition can be readily demonstrated based on Lemma 1 by Harary [53] and Theorem 6 by Whitney [104]: If R is a spanning subgraph of G, the connectivity of R cannot exceed that of G: $k(R) \leq k(G)$. In other words, if G is disconnected (with $k(G) = 0$), then R will also be disconnected. Put differently, if removal of node v results in disconnection of G, it will also lead to disconnection of R. In the simplest case, at least one node v', adjacent to v will become isolated.

4.3 Methodology

The goal of our data collection process is to capture the contents of the `peer.dat` of every reachable peer in a blockchain network. This consists of the peer's *view of the network*, which contains all available peers to which it can connect. This is easily achieved by repeatedly asking peers for addresses they know of. A graphical example is represented in Fig. 4.2. Node A is interrogated by a crawler for known addresses. Black and green arrows represent outbound and inbound connections, respectively. The orange nodes represent peers that are reachable by Node A, while the gray nodes are unreachable.

According to the reference client (see also Section 2.1), the following hold:

1. All outbound connections are contained in the `peers.dat` file. These addresses are known to the node and, after the establishment of a connection, are moved to the `tried` bucket by `ADDRMAN`.

2. Nodes try to retain 8 outbound connections. To initiate a connection, a node selects a candidate IP address from the *tried* and *new* buckets with equal probability.

3. Nodes will accept inbound connections from legitimate peers that are not *terrible*, as described in the official source code.[9]

4. IP addresses of inbound connections **are not** stored in `peers.dat`, **unless** the initiating node includes his address in the initial handshake `addr` message. The reason behind this decision was that the addresses of the inbound peers may not be reachable.[10]. Occasionally, an unreachable inbound address is present in the `peers.dat`, e.g., node *f* in Fig. 4.2. This can occur when the address of node *f* was included in an `addr` message sent from another peer (e.g., *b*) and at a later time node *f* initiated a connection to node *A*.

The next step is to combine the views collected from all reachable nodes into a connectivity graph. This synthesized graph would contain all inbound and outbound links and a collection of *potential* links between all network peers.

[9]Bitcoin Core. *Bitcoin Core, addrman implementation*. 2021. URL: `https://github.com/bitcoin/bitcoin/blob/d0d256536cdfb1443067fb7cc0a19d647f636a5c/src/addrman.cpp#L41`.

[10]Bitcoin Core Developers. *Do not add random inbound peers to addrman.PR8594*. 2020. URL: `https://github.com/bitcoin/bitcoin/pull/8594#issue-173338388`.

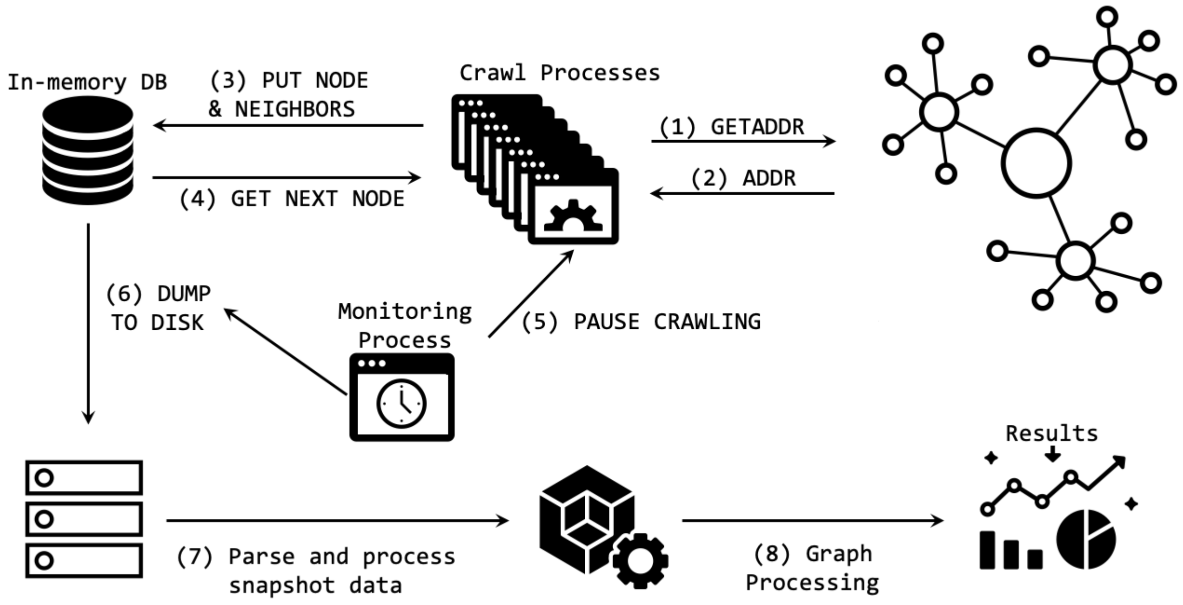


Figure 4.3: Data Collection and Processing Pipeline.

Note: When node *B* disconnects from node *A*, the latter will try to establish a fresh connection using an address from its `peers.dat` file. As we capture the entire `peers.dat` content, the newly formed connection is very likely to be part of the set of potential links that we deduce.

We describe our data collection methodology in Figure 4.3. Steps 1-4 correspond to the parallel crawling process. Steps 5 and 6 represent the synchronized data dump of the snapshots collected from the in-memory database to disk. In steps 7 and 8 we preprocess and analyze the collected graphs.

4.3.1 Crawling Process

To discover the nodes (peers) of the overlay networks, we modified the crawler maintained by the popular site *bitnodes.io* to meet our needs.[11,12]. We added features that enable: a) crawling multiple chains using distinct processes; b) storing the mapping of each node to its known-peers; c) and synchronizing the processes to dump the collected data for each blockchain at the same timestamp.

Implementing an Ethereum crawler is substantially different since it uses a different protocol.

[11]bitnodes.io. *Global Bitcoin Nodes Distribution*. 2020. URL: `https://bitnodes.io` (visited on 09/18/2020).

[12]Addy Yeow. *Bitnodes Network Crawler*. 2021. URL: `https://github.com/ayeowch/bitnodes`.

The Ethereum crawler was built based on the open source Trinity client and disabled all blockchain-related processing.[13] We implement only those parts of the protocols necessary to instantiate connections to Ethereum peers and participate in the discovery process. Since a discovery message on Ethereum provides us with up to 16 addresses, we repeat the discovery process for each discovered peer after a 5-second interval, each time generating a new random target (see Section 2.2 for more details). We note that it is more time-consuming to scrape the DHT content of each Ethereum peer compared to other blockchains.

As already mentioned, our aim is to scrape the contents of any reachable node's `peers.dat` for its known connections and to build a connectivity graph including *any* possible connection that could be realized in the overlay network.

Each blockchain network is assigned to a process that creates hundreds of *greenlets*. Greenlets are lightweight coroutines, for in-process sequential concurrent programming, and are commonly used to provide asynchronous I/O. The intermediate data collected during crawling are maintained in an in-memory key-value store, each process having its own instance. Following the protocols of each blockchain, each process connects to its assigned network and recursively asks each discovered node for its known peers (*Steps 1-2*). Each new discovered node is stored in a `pending` set of the in-memory instance (*Step 3*). Threads constantly poll their `pending` set for a new node (*Step 4*), initiate a connection, and retrieve a list of the node's known peers.

Upon a successful connection with a peer, its entry is moved from `pending` to the `tried` set. Additional data for the peer is collected through the protocol's `VERSION` message, including provided services, the peer's latest block height, the client's software version, and the peer's P2P protocol version. If a node fails to reply or the connection times out, it is moved into a `failed` set.

On each response received to a `getaddr` message, the process makes an entry, mapping the originating node (N_{origin}) to the peer list it knows of: $N_{or} \rightarrow \{P_0, P_1, ..., P_n\}$, where P_{0-n} are the peers included in N_{origin}'s reply. This entry is stored in the `edges` set. When the `pending` set becomes empty, the crawler moves all entries from `tried` to `pending`, and starts over. The `edges` set remains intact and is updated in subsequent rounds. Replies from nodes that are already mapped in the `edge` set are appended to the respective entry. After a period of approximately two hours, all processes synchronize and dump their `edge` set to storage (*Steps*

[13]trinity.ethereum.org. *The Trinity Ethereum Client*. 2021. URL: `https://trinity.ethereum.org`.

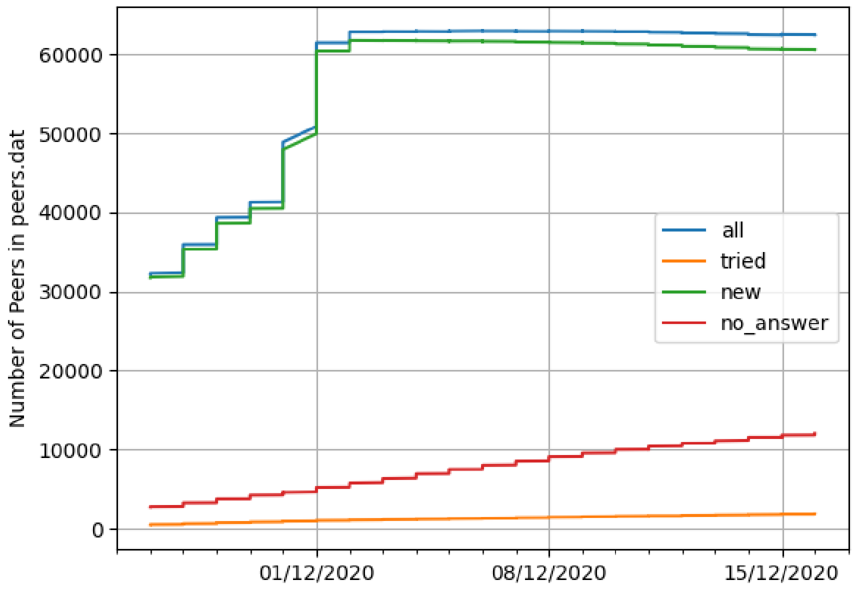


Figure 4.4: Collected IP addresses from the monitoring node. Retrieved by periodically dumping the contents of the peers.dat file.

5-6).[14] After the dump, all sets are emptied and each process restarts and repeats the same procedure. The implemented crawler does not accept incoming connections and only probes IPv4 peers.

In Steps 7-8, we construct the connectivity graph using the collected `edge` set. For each entry $N_{origin} \rightarrow \{P_0, P_1, \dots, P_n\}$ in the set, we create a directional graph with nodes $\{N_{origin}, P_0, P_1, \dots, P_n\}$ and add n outgoing edges from N_{origin} to nodes $P_0, P_1, \dots, P_n$. All edge graphs are then merged into a single synthesized graph. The collected graphs are then analyzed using the SNAP [66] and NetworkX [51] packages.

4.4 Validation

In order to assess the viability of our goal, we setup an unmodified Bitcoin reference node using the official implementation.[15] We allowed the reference node to perform its initial bootstrap

[14]Two-hour periods were chosen, to allow future analysis of longitudinal evolution of the networks. We believe that a larger window would not capture enough of the evolution dynamics.

[15]Bitcoin Core. *0.20.1 Release Notes*. 2021.

of the blockchain for one week. Subsequently, every ten minutes, we retrieve the following information from the reference node: a) all inbound and outbound connections, b) a snapshot of the `peers.dat` file, and c) the `addr` reply to a `getaddr` probing message. We observe that by issuing enough `getaddr` messages, we are able to reconstruct the `peers.dat` file to its entirety.

In Figure 4.4 we plot the contents of the `peers.dat` file. Up until 1st of December, the node was performing its initial bootstrap, to sync the full contents of the blockchain. Soon after the complete download of the blockchain, the monitoring node collects more than 60,000 addresses. Most of the peer addresses are sorted into the `new` buckets (green line). Addresses that are tried and respond to connect requests are moved to the `tried` bucket, while peers that do not respond are plotted with the red line. Evidently, most known peers are either unreachable or cannot support more connections, an observation made by Grundmann in [47] as well. Since the size of the tried set is significantly smaller than that of the new set and a node corresponds to a `getaddr` message with a list of nodes from both sets with equal probability, all addresses in the new set are captured. In fact, the crawler was unable to retrieve only 2 IP addresses from the `tried` set and 3617 addresses from the `new` set, resulting in a total loss of 4.5% of all addresses from `peers.dat`.

During our validation period, the monitoring node created a total of 12,241 connections with other peers, 466 were outbound and 11,775 inbound. We observed 994 unique IP addresses, 368 corresponding to outbound connections, and 634 to inbound connections. Of these, 4 IP addresses were in both sets. The crawler did not capture 444 of the 944 connected IP addresses. Looking into this weakness, we found that the missing IP addresses were not included in the `peers.dat` file. As expected, these were inbound connections from unreachable peers in the network.

Further inspection of the monitoring node logs revealed that most of these peers created short-lived connections that were dropped after the initial handshake. Based on their duration, these could be feeler connections, as described in [54], but this cannot be confirmed.

We also noted client strings that were previously identified as non-contributing nodes by the community.[16] These are summarized in Table 4.1.

Interestingly, only 10 of the missed peers actively contributed to the network. The other missing peers mainly opened connections and periodically pinged. Some of them corresponded to

[16]Bitcoin Forum. *UASF nodes wrongly reporting IP*. 2017. URL: `https://bitcointalk.org/index.php?topic=1954151.0`.

well-known blockchain explorers, like `bitnodes.io`. After excluding the non-contributing peers, the monitoring node connected to a total of 570 unique IP addresses, with the ten missing nodes resulting in a loss of 1.75% of useful connections.

Furthermore, an analysis of messages from the 10 missing nodes using Bitcoin's local RPC API, revealed that all of them were significantly behind on their blockchain, with their most recent block being several days behind the latest block observed by the monitor.

In practical scenarios, if a node becomes disconnected, it will scan its `peers.dat` to identify new peers to establish new connections. Our validation reveals that the uncovered inbound connections (those missed) do not transmit any useful information to the node, including transactions or peer addresses. Consequently, the node remains connected to non-contributing peers, effectively becoming detached from the network.

4.4.1 Validation against external data sources

To further validate the coverage of our crawler against external data sources, we compared our results against the *DSN Bitcoin Monitoring* infrastructure in `https://www.dsn.kastel.kit.edu/bitcoin`, originally presented in [80]. Since the IP addresses of [80] are anonymized (IP addresses are not public), we compare the number of reachable nodes we capture with the number of nodes scanned by the DSN Monitor. Counting only the reachable peers, we find that our crawler is able to capture a few hundred more nodes on a daily basis. Similarly, we compare node counts with the historical data collected by the `bitnodes.io` crawler and a known Bitcoin core developer with similar results.[17]

4.5 Datasets & Experiments

Using the methodology aforementioned, we crawled the selected blockchain networks from CUT's data center. The monitoring server has an 8-core/3.2GHz CPU, 64GB RAM, and 2.1TB of HDD storage.

The crawling operations were performed for a period of about one month (26/06-22/07/2020). Past work [21, 22, 62, 73] used similar duration for their analysis. At the end of the crawling period, we had collected 335 network snapshots for each BC network; 2345 graphs in total.

[17] Luke Dash Jr. *Bitcoin Historical Node Count*. 2022. URL: `https://luke.dashjr.org/programs/bitcoin/files/charts/historical.html`.

Table 4.1: List of non-contributing client strings discovered by the monitoring node.

Client version string	**Description**
/btc-seeder:0.0001/	BTC DNS seeder
/bitcoinj:0.15.x/	Wallet Java implementation [114]
/bitnodes.io:0.1/	Network Monitor
/nodes.multiven.io:0.1/	Network Monitor
/dsn.tm.kit.edu/bitcoin:0.9.99/	Network Monitor
/bitcore:1.1.2/	Network Monitor
/btcwire:0.5.0/bitcoinstats.com:0.2/	Network Monitor
/bitnodes.earn.com:0.1/	Network Monitor
/bitcore:8.19.0/	Node.js wallet [119]
/btcwire:0.5.0/btcd:0.12.0/	Network Monitor
/Block.io:0.1.0/	Network Monitor
/BUCash:1.1.2(EB16; AD12)/	Suspicious Node [136]
/btcwire:0.5.0/bitmarkd-payment-lightclient:0.1.0/	wallet
/bitcoinj:0.15.5/Bitcoin Wallet:7.22/	wallet
/libbitcoin:3.2.0/	wallet
/bitcoinj:0.15.1/BitRafael:2.0.0/	Transaction Monitor [118]
/bitcoinj:0.15.5/Bitcoin Wallet:7.27/	wallet
/pytxhorn:0.03/	Unknown

Our results (see Section 5.5) indicate that 4 weeks of data exposes a sufficient amount of information and dynamicity to be studied. The data set collected is available online at [147].

We denote by C the set of the 7 blockchain networks crawled.

$$C = \{Bitcoin, BitcoinCash, Dash, Dogecoin, \\ Ethereum, Litecoin, Zcash\}$$

At the end of every two-hour interval, we have seven different `edge` sets, one per blockchain $c \in C$. At the end of each day, all edge sets belonging to the same network are merged in a 24-hour set. All sets are annotated with the date t of their crawl. Each set of edges corresponds to a graph, denoted S_c^t, representing a snapshot of the blockchain network c, on date t.

4.6 Discussion

The validation results show that the proposed method can effectively create a network snapshot that accurately captures all current connections in the overlay, as well as potential connections that may be established. Previous research [12] has found that a 5% error rate in missing links is acceptable, even in denser graphs (cf. Section 5.1.1), when analyzing the impact of targeting strategies on network structure. We believe that the error rate observed in our study is small enough to allow us to draw meaningful conclusions.

In the upcoming chapter, we will delve into the network properties of the synthesized graphs. Using the collected data, we will analyze the networks' resilience against targeted attacks, explore the temporal characteristics of the networks, investigate the simultaneous participation of blockchain nodes in multiple networks, and examine the properties of these overlapping peers.

4.7 Graph Analysis of Blockchain Networks

Graph analysis is a powerful tool for understanding network resilience. It has been widely used to characterize complex networks and investigate resilience in various fields and applications in a variety of network types, such as technological, social, infrastructure, transportation, and biological. A recent survey highlights the prevalence of graph analysis with respect to network resilience research [37]. Graph analysis has also been used extensively to study the transaction graphs of major blockchains, namely Bitcoin and Ethereum [112, 15, 60, 68, 83, 108, 110]. Using similar methods, Lee *et al.* analyze Bitcoin's Lightning Network [65]. In their work, they found that it exhibits strong scale-free network characteristics, implying that the Lightning Network can be vulnerable to DDoS attacks targeting some central nodes in the network.

Although graph analysis is an indispensable tool for assessing network robustness, it has not been applied to blockchain networks so far. We believe that a contributing factor to this omission in the literature is mainly the lack of topological information on the underlying networks.

In this study, we employ graph analysis techniques to explore the structural and resilience aspects of these networks. Through their graph properties, we compare their structures and examine how these characteristics influence their security attributes. The following section presents the evaluation metrics used for our analysis.

4.7.1 Evaluation metrics for resilience

The robustness of complex networks cannot be fully represented by a single metric. Different graph metrics have been proposed to assess network resilience against errors and attacks. This section introduces both local (node-level) and global metrics used to evaluate the resilience of blockchain networks. These metrics are derived from previous studies [3, 37, 86, 57, 2].

Diameter The diameter of a connected graph is defined as the longest shortest path between all pairs of nodes. A smaller diameter usually indicates better robustness, as adding edges would shorten the longest shortest path between distant nodes, making the network more tightly coupled.

Average Shortest Path This metric is closely related to network connectivity. Smaller average shortest paths imply increased robustness, since the distance between any pair of nodes is reduced.

Average Node Betweenness The average node betweenness is the sum of node betweenness centrality for all nodes in a graph. Betweenness centrality of a node v is the sum of the fraction of the shortest paths of all pairs that pass through v [36]. A smaller average betweenness indicates that the shortest paths are more evenly distributed among the nodes; thus, it implies greater robustness. Nodes with high-betweenness centrality tend to play a prominent role in networks, as they act as a bridge between groups of other nodes. Nodes with fewer connections than others may still have high betweenness, allowing them to fulfill a broker role and facilitate communication and information flow throughout the network. In effect, high average betweenness implies that network connectivity relies on a few central nodes and that such networks are more susceptible to targeted attacks.

Assortativity The assortativity coefficient lies within the range $\rho \in [-1, 1]$ and defines two types of networks. Disassortative networks with $\rho < 0$ have a surplus of links that connect nodes of different degrees. The opposite applies to assortative networks, with $\rho > 0$, that have an excess of links connecting nodes of similar degrees. According to Iyer *et al.* [57], disassortative networks tend to exhibit greater vulnerability to targeted attacks (see also [70] and Newman [82], Fig.3).

Clustering Coefficient The global clustering coefficient C is based on the number of triplets of nodes in the graph and provides an indication of how well the nodes tend to cluster together. A triplet is defined as three nodes connected by two edges. A triangle is a closed triple, i.e., three nodes connected by three edges. The global clustering coefficient is the number of closed triplets (or 3 x triangles) over the total number of triplets (both open and closed). A higher clustering coefficient indicates the presence of redundant pathways between nodes (due to the higher number of triangles), increasing the overall robustness of the network.

4.7.2 Limitations

Table 4.2: Measurement error simulation results.
Each data point is an average over 20 realizations with $N = 1000$. *Betweenness not normalized.

Metric$\rightarrow$	**Avg. Shortest Path**	**Average Degree**	**Clustering**	**Assortativity**	**Avg Betweenness***
G_{real}	1.89	114.6	0.21	-0.02	447,893
G_{obs}	1.56	437.7	0.63	0.07	280,648
Chebyshev Distance	0.34	333.9	0.43	0.12	172,904

Arguably, the observed connectivity graphs contain a number of false-positive edges in the graph, i.e., they contain edges that do not exist in the real network. To understand how much the network properties are affected by these errors, we turn to an area of research that deals with measurement errors in network data. Wang *et al.* [100] studied the effect of measurement errors on node-level network measures and found that networks are relatively robust to false-positive edges. Similarly, Booker described the effects of measurement errors on the attack vulnerability of networks [12]. Booker also finds that false-positive edges have the least impact on the effectiveness of random and targeted attacks.

To investigate the accuracy of the observed graphs compared to real networks, we adapt the methods used by Booker and Wang [12, 100]. In particular, we construct a random graph G_{real} consisting of $N = 1000$ vertices, assigning to each vertex k outgoing links, so that k is drawn from the real Bitcoin degree distribution, as calculated by Grundmann *et al.* in [47] (cf. Section 3.2.1).

Then, starting with G_{real}, we add random edges with the constraint that the resulting observable graph, G_{obs}, has a degree sequence drawn from the observed degree distribution we obtain using the methodology described previously; i.e., by probing peers for their known addresses (see Section 4.3.1). Since Grundmann's calculated degree distribution applies only to reachable peers, we also use the degree sequence of reachable peers, ignoring any unreachable nodes. In this way, the resulting *observable* graph, G_{obs}, contains a number of real links plus an additional number of edges that correspond to the known peers of each node (false positive edges in [12]). To inspect the effects of false edges on the observed network characteristics, we calculated a set of graph metrics for both graphs G_{real} and G_{obs} and compared them.

The average values calculated from 20 simulations are presented in Table 4.2. G_{obs} exhibits more robust characteristics, evident by a higher clustering and a lower average betweenness. This is expected as it contains much more edges than G_{real}. On the other hand, the average shortest-path values are very close in both sets of graphs. The results of this simulation show that the differences in the calculated metrics are consistent and almost constant. Thus, the calculated properties of the observed graphs can serve as a bound to the properties of the real graphs. The Chebyshev distance in the last row indicates the maximum absolute distance between the corresponding values.

4.8 Towards a Unified Model for Blockchain Networks

Blockchain systems exhibit a diverse array of network states, each governed by its unique mechanisms and possessing specific intricacies and features. These networks are dynamic, with peer nodes assuming varying roles and importance in information propagation over time. For example, a highly connected relay node that is the first to have received a valid block from a miner may be crucial for timely propagation of the block to a significant proportion of the network.

This heterogeneity presents a challenge when trying to analyze and compare different blockchain systems. The question arises: How can we comprehensively assess these networks, considering their distinct characteristics and the ever-evolving nature of their peers?

To address this challenge, we propose a unified network model. Rather than differentiating network elements based on their roles or significance within the system, our approach treats all elements uniformly. This model allows us to simultaneously study various blockchain networks. while maintaining a consistent framework for analysis. In particular, our uniform treatment extends to include previously overlooked unreachable peers identified through our measurements, enriching our insight into network dynamics. This approach improves our ability to study the structural attributes of diverse networks, regardless of their unique characteristics and complexities.

The advantages of this uniform treatment extend beyond inclusivity and relevance to diverse networks. Although the importance of nodes can vary over time in such dynamic networks, trying to assign varying degrees of importance to nodes at each timestamp could complicate the analysis. A uniform network model simplifies our data processing pipeline. By focusing on structural characteristics while avoiding distinctions based on the role of network elements, our methodology provides a comprehensive framework for studying and comparing blockchain overlay networks across different systems and network conditions. This unified approach forms the foundation of our research, allowing us to gain meaningful insight into the structural resilience of blockchain networks and uncover vulnerabilities that transcend the boundaries of specific blockchain systems.

Having outlined our approach and its advantages, the next step involves processing the collected data to retrieve the characteristics of each network and analyze their resilience properties in the following chapter.

Chapter 5

Results

In this chapter, we present the findings and insights garnered from our measurements and analysis. These results shed light on various aspects of blockchain overlays, including their structural properties, vulnerabilities, temporal dynamics, and interdependencies. By thoroughly investigating these aspects, our goal is to gain an understanding of how blockchain overlay networks behave and their key features.

5.1 Analysis of P2P Overlays

We aim to answer the following questions about blockchain overlay networks: a) What are their structural properties and network characteristics? b) Are they all structured similarly? c) Do they share common properties? d) Do their properties relate to other networks such as the Internet topology, Web or social networks, or are they random? e) How do their characteristics affect their resilience?

This section presents metrics, adapted from previous research [103, 3, 37, 86, 57, 2], to assess the resilience of a blockchain network. These metrics are considered standard for analyzing networks and understanding non-obvious properties, and can be used to evaluate network resilience to errors and attacks. In this section, we use the following notation for clarity and conciseness: each set of edges corresponds to a graph, denoted S_c^t, representing a snapshot of the blockchain network c, on date t.

Other online networks Online social networks, the Web and the Internet / AS topology are the most studied online networks [74, 69, 93, 14]. This section shares much of the methodology used in such studies. It is reasonable to compare the structure of blockchain networks with the structure of other known technological and information networks. Nevertheless, we are aware that: a) the studied graphs do not represent the actual network topology, and b) the P2P structure of blockchain networks is fundamentally different from the aforementioned networks. The comparisons made throughout this section serve as a reference point for the results collected. However, we note that useful conclusions can be drawn about blockchain overlays, especially when comparing the different networks between them, since they implement similar protocols [23] and we follow the same measurement methodology.

5.1.1 Fundamental graph properties

The most important properties of the derived graphs are summarized in Table 5.1. The metrics were individually calculated on each graph S_c^t and then averaged. The values extracted from the collected data sets match the values reported in related measurement works [62, 25, 21]. Specifically, each day, the monitoring node was able to discover 120081 nodes in Bitcoin, 19543 in Ethereum, and 4132 in Zcash (reporting median values). On average, the monitoring node made more than $1.3M$ requests per day, covering all networks.

The diameter of a connected graph is defined as the *longest* shortest-path between all pairs of nodes. A smaller diameter usually indicates better robustness, as adding edges would shorten the longest shortest path between distant nodes, making the network more tightly coupled. The Average Shortest Path (ASP) is closely related to network connectivity. Smaller average shortest paths imply increased robustness, since the distance between any pair of nodes is reduced. All networks appear to be well connected, given the size of their largest connected component, their low diameters, and short ASP. Moreover, we observe that Dash is markedly the most dense network and is almost fully connected. It has a strongly connected component (SCC), i.e., a subgraph in which all nodes are reachable from all other nodes. The SCC comprises 75% of the total network nodes. Larger blockchain networks have a smaller SCC compared to smaller ones. Networks differ mainly in size, but this is independent of their protocols; in a free market, user perception of value determines a network's popularity. We believe that variations in participation also affect the presence of unreachable peers, which is further discussed in Section 5.1.10.

Table 5.1: Basic network graph metrics per network (average values across all collected snapshots).
For each metric (row) we highlight the value that indicates less resilience.
*Average Betweenness Centrality of Normalized Vertex Betweenness using the min-max method.

Network:	**Bitcoin**	**Bitcoin Cash**	**Dash**	**Dogecoin**	**Ethereum**	**Litecoin**	**Zcash**
Nodes	120k	33k	9k	2.1k	17.5k	11.7k	4.1k
Edges	37M	748k	29M	330k	556k	3.7M	231k
Connected Component	1	1	1	1	0.99	1	1
Strognly Connected Component	0.06	**0.03**	0.75	0.2	0.13	0.14	0.06
Diameter	4	4	3	3	**5**	3	4
Density	0.004	**0.001**	0.5	0.11	0.004	0.047	0.024
Avg. Degree	254.16	**20.22**	2370.88	126.45	31.14	278.85	48.84
Assortativity	-0.2	**-0.64**	-0.06	-0.13	-0.02	-0.01	-0.22
Reciprocity	0.32	0.21	0.49	0.34	0.02	0.27	0.25
Global Clustering Coefficient	0.049	0.011	0.166	0.28685	**0.0022**	0.0735	0.3094
Avg. Shortest Path	2.55	2.82	1.93	1.77	**3.78**	1.96	1.72
Average Betweenness	**2.40e+07**	1.95e+06	2.74e+06	1.62e+04	1.12e+06	5.35e+05	1.43e+04
Normalized Betweenness*	**49727**	23018	8666	1257	8871	8160	1462

5.1.2 Degree Distribution

A node's degree is the number of its links with other nodes. Degree distribution affects many network phenomena, such as network robustness and efficiency in information dissemination (see Barabasi [9], Section 2.3 and 8.3). In addition, random networks have binomial degree distributions, whereas in real systems we usually encounter highly connected nodes that the random network model cannot account for. In Figure 5.1, we plot the complementary cumulative distribution (CCDF) of the out-degree of all snapshots collected for all networks in our study.

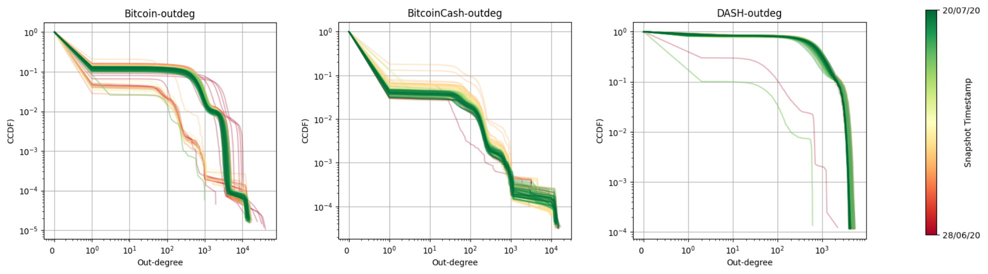
We color the snapshots according to their timestamp. Our first observation is that networks such as Bitcoin and Ethereum manifest considerable variability in the degree distribution between snapshots. On the contrary, the degree distributions in Dash and Dogecoin have less variability (seen by the distance between snapshots). Another interesting observation is that in most networks we have a high fraction of unreachable nodes, either because they are offline or behind NATs. This observation confirms the findings of Wang and Pustogarov [101] who studied the prevalence and deanonymization of unreachable peers. The presence of unreachable peers is discussed later in this chapter.

Our results also suggest that these blockchain networks have heavy-tailed degree distributions. We further discuss their best distribution fit and their scale-free property in a following paragraph. Finally, we observe significant deviations from the network protocols. In Bitcoin, for instance, one would expect that reachable nodes would have at least 1K out-degree, since Bitcoin clients with the default parameters are set to respond with 1K known peers. In contrast, we observe a number of nodes with an out-degree less than 100, i.e., nodes reply with fewer addresses than the default parameter. We note that this behavior along with network churn could be leveraged to amplify eclipsing or network attacks similar to the SyncAttack [87].

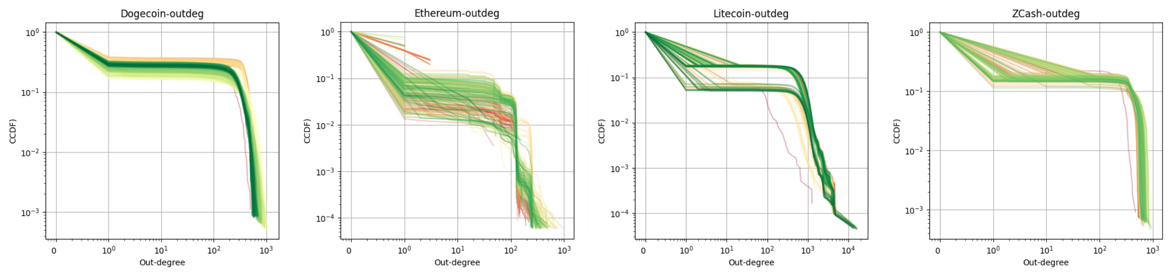
Comparing the network densities, we observe that DASH has a very tight network, while Bitcoin, BitcoinCash, and Etherum are much less dense. This result indicates that DASH and Dogecoin have a more resilient structure than other networks.

5.1.3 Degree Assortativity

In general, a network shows degree correlations if the number of links between the high- and low-degree nodes is systematically different from what is expected by chance. In some



(a) Bitcoin **(b)** Bitcoin Cash **(c)** Dash **(d)** Colorbar



(e) Dogecoin **(f)** Ethereum **(g)** Litecoin **(h)** ZCash

Figure 5.1: Out-degree complementary cumulative distribution function of collected graphs. Snapshots are colored according to the colorbar.

types of networks, the high-degree nodes (or hubs) tend to link to other such hubs, while in other types, the hubs tend to link to low-degree nodes, i.e., what is known as a hub-and-spoke pattern. Assortativity, or assortative mixing, is a preference for nodes in a network to attach to others that are similar in some property; usually a node's degree.

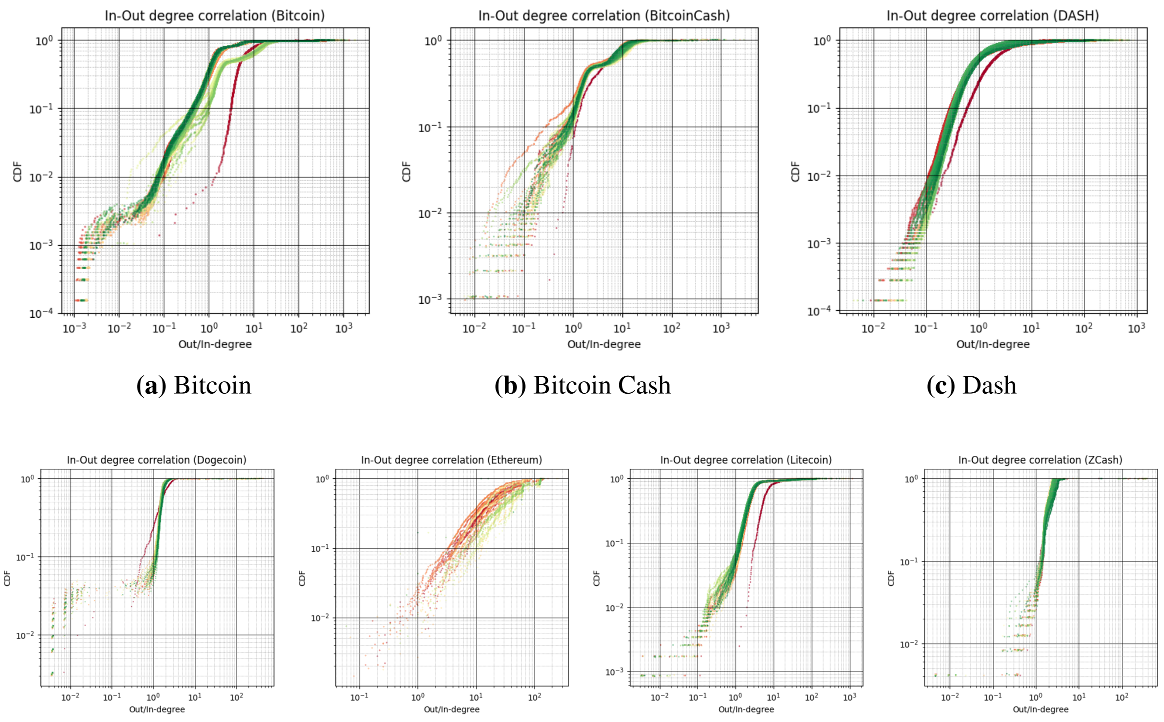
The assortativity coefficient, ρ, is the Pearson's correlation coefficient of degree between pairs of linked nodes and lies in the range $-1 \leq \rho \leq 1$. A network is said to be assortative (ρ tends to 1) when the high-degree nodes tend to link to each other and avoid linking to the low-degree nodes, while the low-degree nodes tend to connect to other low-degree nodes. A network is said to be disassortative (ρ tends to -1) when the opposite happens. A random network has ρ close to zero and can be characterized as neutral. Incorporating this feature into network models improves the accuracy of the model in simulating the behavior of real-world networks. Disassortative networks tend to exhibit greater vulnerability to targeted attacks [57, 82, 70].

Correlations between nodes of similar degree are common in various observable networks. Social networks tend to exhibit assortative mixing, whereas technological and biological networks often show disassortative mixing, with high-degree nodes connecting to low-degree nodes. In disassortative networks, low-degree nodes, particularly those that have recently joined the network, can be discovered more quickly when connected to hubs, i.e., nodes with very high degree (above the network average). Removing these hubs can impact node discovery, graph connectivity, and potentially facilitate attacks such as eclipsing. Adversaries with high connectivity can exploit this knowledge to advertise malicious peer addresses, compromising the `ADDRMAN` of benign peers.

We compute the assortativity coefficient for each snapshot, reporting the average values in Table 5.1. The networks analyzed exhibit negative assortativity, with DASH, Dogecoin, and Litecoin being closer to neutral (assortativity close to 0). On the contrary, Bitcoin Cash, Zcash, and Bitcoin display a more pronounced disassortativity. Negative assortativity indicates a hub-and-spoke structure in these networks, suggesting the presence of central peers that are crucial to the network and susceptible to targeted DDoS attacks.

5.1.4 Out/In Degree Ratio

Link analysis using the in-degree and out-degree distributions has proven very powerful in identifying authoritative and central nodes in the Web and social networks [63]. We computed the out-degree over in-degree ratio of individual nodes for all snapshots and used it to compare the structure of blockchain networks to the Web and social networks. In the Web, most nodes have considerably higher out-degrees than in-degrees ($\frac{Out}{In} > 1$), while a small fraction of nodes have significantly higher in-degrees than out-degrees ($\frac{Out}{In} < 1$). Social networks have a substantial correlation between in-degree and out-degree and most nodes have an in-degree within 20% of their out-degree [74]. In Figure 5.2, we show the CDFs of the outdegree-to-indegree ratio for nodes in all networks. Dogecoin marginally resembles a social network with 45% of its nodes having a good correlation, within 20%. This can be explained by the high number of symmetric links, since Dogecoin has high reciprocity (see Table 5.1). In Bitcoin and Bitcoin Cash, most nodes have considerably higher out-degree than in-degree, while a small fraction of nodes have significantly higher in-degree than out-degree, a characteristic similar to the Web. Their similarity to the Web can be further strengthened by their dissassortative nature (see Table 5.1). In Dash, we cannot observe any correlations, with half the nodes having higher out-degree and the other half having higher in-degree. This indicates that half of peers in Dash are frequently online in contrast with the other half that participate periodically. Our longitudinal analysis in Sec. 5.5, confirms this indication. In Litecoin and Zcash, we observe a concentration of nodes having higher out-degrees, and compared with their assortativity, this indicates that nodes tend to connect to lower degree



(a) Bitcoin **(b)** Bitcoin Cash **(c)** Dash

(d) Dogecoin **(e)** Ethereum **(f)** Litecoin **(g)** Zcash

Figure 5.2: Cumulative Distribution Function (CDF) of Out/In ratio per network. Snapshots are colored according to Fig. 5.1.

nodes. In Ethereum, due to its distinguished discovery mechanism, the vast majority of nodes have a very high out- over in-degree ratio.

5.1.5 Reciprocity

Moreover, we measure the reciprocity property, which is a measure of the likelihood of vertices in a directed graph to be mutually linked. It has been shown to be critical in modeling and classification of directed networks [40]. Table 5.1 lists the average reciprocity across all snapshots for each network. Zcash, Dogecoin, and Dash have significantly higher reciprocity values. We can attribute this finding to a similar explanation for size and clustering. Closer examination also reveals that these networks have a considerable number of nodes that are frequently online. Nodes that are frequently connected are more likely to connect to nodes that are also frequently connected, which drives reciprocity higher.

While the likelihood of reciprocity emerging by chance is minimal, it's noteworthy that numerous technological networks exhibit this characteristic [81]. Regardless of its origins, the presence of reciprocal links has a notable impact on a network's structure. Increased reciprocity contributes to enhanced overall connectivity and reduced network diameter. In the context of our measurements, reciprocity is considered a desirable attribute as it signifies that connected peers are more readily discovered by other nodes in the network.

5.1.6 Clustering Coefficient

The global clustering coefficient C is based on the number of triplets of nodes in the graph and provides an indication of how well the nodes tend to cluster together. A triplet is defined as three nodes connected by two edges. A triangle is a closed triple, i.e., three nodes connected by three edges. The global clustering coefficient is the number of closed triplets (or 3 x triangles) over the total number of triplets (both open and closed). A higher clustering coefficient indicates the presence of redundant pathways between nodes (due to the higher number of triangles), increasing the overall robustness of the network. The global clustering values are presented in Table 5.1.

We observe that larger networks tend to have lower clustering than smaller networks, with Ethereum having the lowest value. This indicates that larger networks exhibit less robust characteristics. We suspect that this is closely related to the presence of unreachable peers.

Unlike global clustering, the local clustering coefficient CC_i measures the density of links in the immediate neighborhood of node i: $CC_i = 0$ means that there are no links between i's neighbors, while $CC_i = 1$ implies that each of i neighbors also links to each other. In a random network, the local CC is independent of the node's degree, and average CC, i.e., $< CC >$, depends on the size of the system with respect to the number of nodes, N. On the contrary, measurements indicate that for real networks, e.g., the Internet, the Web, science collaboration networks, CC decreases with the degree of the node and is largely independent of the size of the system [9]. The local CC in a random network (CC_{rand}) is calculated as the average degree $< k >$ over N, i.e., $CC_{rand} = \frac{<k>}{N}$. The average degree of a network is $\frac{2L}{N}$, where L is the number of links. The average CC of a real network is expected to be much higher than that of a random graph (see Barabasi [9], Section 3.9).

In Figure 5.3(a), we compare the average CC of the collected graphs with the expected CC for random networks of similar size.

The y-axis corresponds to $\frac{<CC>}{<k>}$, where $< k >$ is the average degree. On the x-axis we plot the network size N. Size and CC are averaged across snapshots S_c^t for all timestamps: $S_c^t \forall t \in T$. Markers correspond to the networks of Table 5.1. Lines correspond to the prediction for random networks, $CC = \frac{<k>}{N}$, with constant $< k >$ and varying size N. Similar to other known networks, the average CC appears to be independent of the network size N, indicating that the synthesized graphs deviate significantly from random networks.

In Figure 5.3(b), we plot the dependence of CC on the degree of the node. $CC(k)$ is measured by averaging the local CC of all nodes with the same degree k (showing results of aggregating all snapshots of a given network). Horizontal lines correspond to the average CC of the network. We plot only two of the networks studied, where we make some remarkable observations. Although the empirical rule of Barabasi [9] states that higher-degree nodes have lower CC, in Bitcoin we observe a significant fraction of high-degree nodes with high CC. The same finding was observed in the Ethereum and Zcash graphs. Another deviation from the same empirical rule is observed in Dash, where all nodes appear to have an almost constant CC, independent of the node degree. We attribute this behavior to its temporal characteristics, previously discussed in the results related to Fig. 5.1. Further inspection reveals that Dash has very low churn and that most nodes are always online. Temporal characteristics are presented in more detail in Section 5.5. The observed CC distributions indicate that the collected graphs are governed by rules that are rarely encountered in other known network systems. Note that the actual networks represented by these synthesized graphs are likely to have lower CCs, since we would expect fewer edges (see also Table 4.2).

As explained in Chapter 4, synthesized graphs are constructed by node advertisements. From Figure 5.3 we can say that almost all nodes in the Dash network know and advertise almost all other peers. This is not surprising given the size of the network and the strongly connected component being very high. In contrast, the Bitcoin network exhibits variations in the clustering coefficient, indicating that not all nodes know and advertise all other peers. This is partly explained by the size of the network and the high presence of unreachable peers (see also Sect. 5.1.10). The temporal dynamics of the network could also affect peer announcements.

5.1.7 Average Betweenness Centrality

Average betweenness centrality measures how many short paths between vertices in the network pass through a given vertex. The betweenness centrality of a node v is given by the expression: $g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$ where σ_{st} is the total number of shortest paths from node s to node t and $\sigma_{st}(v)$ is the number of those paths that pass through v. Nodes with high

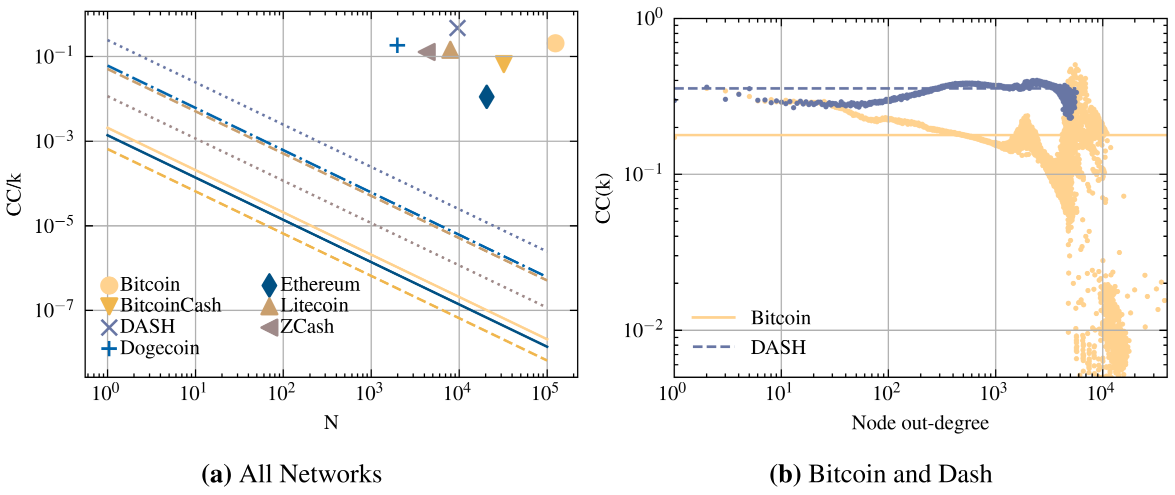


Figure 5.3: Analysis of Clustering Coefficient (CC) results.
(a) $\frac{<CC>}{<k>}$ vs. network size; Markers correspond to the networks of Table 5.1. Lines correspond to the prediction for random networks, $CC = \frac{<k>}{N}$, with constant $< k >$ and varying size N. (b) Dependence of the local CC on the node's degree for each network.

betweenness centrality act as bridges between parts of the network and therefore have a great control in the connectivity and information propagation of the network. It has been demonstrated that attacking or removing highly central nodes is one of the most effective strategies to partition a network or diminish its largest connected component [69, 57].

The average node betweenness is the sum of node betweenness centrality for all nodes in a graph. Betweenness centrality of a node v is the sum of the fraction of the shortest paths of all pairs that pass through v [36]. A smaller average betweenness indicates that the shortest paths are more evenly distributed among the nodes; thus, it implies greater robustness. Nodes with high betweenness centrality tend to play a prominent role in networks, as they act as a bridge between groups of other nodes. Nodes with fewer connections than others may still have high betweenness, allowing them to fulfill a broker role and facilitate communication and information flow throughout the network. In effect, high average betweenness implies that network connectivity relies on a few central nodes and that such networks are more susceptible to targeted attacks. High variance in the betweenness centrality distribution is also an indication of lower robustness, as observed in [106, 2]. Bitcoin and BitcoinCash have very high values of average betweenness, which further suggests that these networks are less resilient.

5.1.8 Scale-free property

One network property, closely related to the degree distribution of a network, is the scale-free property. A scale-free network is defined as a network whose degree (k) distribution follows a power law, i.e., having a probability distribution $p(k) \propto k^{-\alpha}$. The exponent α is known as the scaling parameter and typically lies in the range $2 < \alpha < 3$. The scale-free property strongly correlates with the network's robustness to random failures and has received tremendous attention in the scientific literature (e.g., see [9]). Many real-world networks have been reported to be scale-free, although their prevalence is questioned [19].

In their seminal work [1] titled "Statistical mechanics of complex networks", Albert and Barabasi study the importance of network topology in the error tolerance of complex networked systems. Their results indicate a strong correlation between robustness and network topology. In particular, scale-free networks are more robust than random networks against random node failures but are more vulnerable against targeted attacks. On the other hand, random networks exhibit greater resilience against targeted node removal. Hence, in this subsection, our focus lies in identifying whether the structural attributes of blockchain overlays align with the features of random or scale-free networks.

To test how well the degree distribution of each network snapshot can be modeled by a *power-law* (*PL*), *log-normal* (*LN*), *power-law with exponential cutoff* (*PLEC*) or *stretched exponential* (*SE*), we calculate the best fit using the *powerlaw* package available by Alstott *et al.* [4].

In Table 5.2, we report the number of times each type of distribution was the best fit, for all snapshots of the same network. Evidently, most snapshots do not satisfy the power-law distribution. However, the degree distributions of the collected graphs belong to the exponential family of distributions.

According to sources [24, 22, 27] Bitcoin's network formation procedure is intended to induce a random graph. Previous research [73, 24] showed that the Bitcoin network does not resemble a random graph. Our results indicate that the synthesized graphs are also substantially different from random networks, thus less resilient against targeted attacks.

Furthermore, the calculated results indicate the dynamic nature of blockchain networks. Such networks that change over time may fit different distributions depending on the snapshot collected, something that is also visible in Figure 5.1. These results also suggest that blockchain overlays are not structured in the same way.

Table 5.2: Degree distributions of graphs best-fit for different types of exponential distributions. *PL*:power-law; *LN*: log-normal; *PLEC*: power-law with exponential cutoff; *SE*: stretched exponential.

Disrtibution	Bitcoin	Bitcoin Cash	Dash	Dogecoin	Ethereum	Litecoin	Zcash
LN	6.29%	76.90%	-	49.40%	21.90%	40.10%	0.60%
PL	0.60%	16.20%	1.80%	4.80%	24.60%	12.60%	18.90%
PLEC	93.11%	6.90%	57.20%	-	18.30%	46.40%	-
SE	-	-	41%	45.80%	35.30%	0.90%	80.50%

5.1.9 Small-world property

The small-world phenomenon states that if you choose any two individual nodes in a small-world graph, the distance between them will be relatively short and definitely orders of magnitude smaller than the size of the network. The small-world effect has obvious implications for the diffusion processes that take place in networks. For example, if one considers the spread of information across a network, the small-world effect implies that the spread will be fast.

We examined all collected snapshots to see if they satisfy the small-world property, by calculating the ω metric proposed by Telesford *et al.* in [95]. The metric is defined as $\omega = \frac{L_r}{L} - \frac{C}{C_l}$ where L and C are the average shortest path and the average clustering coefficient of the snapshot, respectively. L_r is the average shortest path for an equivalent random network, and C_l is the average clustering coefficient of an equivalent lattice network. The value of ω ranges between -1, when the network has lattice characteristics, to $+1$ when the network has random graph characteristics, with values near 0 interpreted as evidence of small worldliness. The average shortest path of a random network, L_r, is given by $\frac{ln(n)}{ln(k)}$ [9]. The Clustering Coefficient of the lattice, C_l is calculated as $\frac{3}{4}\frac{k-2}{k-1}$ [91]. (See also[1]) The parameter k is the average degree.

We did not find evidence that the networks under study satisfy this property. Although we observe low average distances in all graphs, they do not have high enough clustering coefficients to be considered as small-world. Indicatively, the ω values we calculated are

[1]Yaron Singer. *CS 134 / Econ 1034: Networks. Class Notes.* 2017. URL: `https://web.archive.org/web/20210415052757/http://www.hcs.harvard.edu/~cs134-spring2017/wp-content/uploads/2017/01/section2.pdf`.

greater than 0.5 for Dash and Zcash. The rest of the networks have values greater than 0.8. According to Table 4.2 we would expect the real networks to exhibit lower clustering coefficients but similar average shortest path length, therefore driving ω even higher. Thus, we expect that the real blockchain networks would not satisfy the small-world property. Furthermore, this suggests that blockchain overlays could be further improved to increase efficiency and reduce propagation delays.

5.1.10 Presence of Unreachable Nodes

It is well known that the vast majority of nodes on the Bitcoin overlay network are unreachable [101, 46]. Our collected data verify this and also suggest that unreachable peers are present in all blockchain overlays. In Table 5.3 we list our findings. The in-degree indicates how many reachable peers advertise an unreachable address. Notably, a high percentage of unreachable nodes appears in all networks, leading to the observation that blockchain networks have a strongly connected core and a high number of unreachable nodes that lie on the fringe of the network. DASH stands out for having much less unreachable peers.

Unreachable nodes were previously known to exist in the Bitcoin and Ethereum networks, but this class of peers has received little attention from the research community. It has been demonstrated to play an important role in blockchain systems [101]. Our results indicate that they are also present in all blockchain networks, although at different percentages.

The presence of unreachable peers, which can affect the properties of a network, is not related to the network protocols used. Their presence is more likely to be influenced by socioeconomic factors such as the popularity of a cryptocurrency, its value, and the availability of compatible wallet software. Many blockchain clients, such as cryptocurrency wallets, appear as unreachable peers in a network, and the number of these peers depends on the factors mentioned above. However, we observe that networks with a high percentage of unreachable nodes exhibit rather less robust properties (see Table 5.1) such as high average betweenness, lower density, and lower clustering.

5.2 Discussion

In this section, we analyze the structure of seven blockchain networks and evaluate their resilience based on the computed graph properties. Our results are summarized below:

Table 5.3: Presence and median in-degree of unreachable peers in each overlay.

Network	% of unreachable nodes	Median in-degree
Ethereum	98%	4
BitcoinCash	96%	3
Bitcoin	88%	3
Litecoin	86%	**75**
ZCash	84%	4
Dogecoin	73%	**68**
DASH	18%	**984**

- Major blockchain networks have characteristics that indicate a less resilient structure. In particular, Bitcoin, BitcoinCash, and Ethereum display lower density and higher average average betweenness than other networks, suggesting increased vulnerability to targeted attacks.
- Among the networks studied, BitcoinCash appears to be the most vulnerable, demonstrating lower density, a dissassortative nature, and high average betweenness.
- Despite using similar protocols (excluding Ethereum), the networks exhibit distinct structural properties and resilience traits. Possible explanations for these differences include variations in network size, temporal characteristics, and the presence of unreachable peers.
- The networks' degree distribution per snapshot demonstrates significant variation. While some snapshots align with power-law distributions, others exhibit better fits with log-normal, power-law with exponential cutoff, or stretched exponential distributions. Generally, the degree distributions observed belong to the exponential family. However, the observed variations indicate very dynamic networks that change often.
- Their clustering coefficient distributions are similar to other real networks and differ from random networks with similar size and average degree. They have low diameters and short average shortest path lengths, but we did not observe evidence of satisfying the small-world property.

It is important to note that our results are derived from connectivity graphs constructed using P2P address propagation, rather than representing the real topology of the networks. As a result, the networks studied may not accurately reflect precise network properties. Table 4.2 illustrates how these results can establish limits for the properties of real networks. Our simulations in Section 4 indicate that real networks are likely to exhibit lower clustering and higher betweenness, rendering them less resilient than our observations suggest.

5.3 Network Resilience to Attacks

This section answers the following research question: To what extend are blockchain overlays prone to random failures and targeted attacks? We start this investigation by first describing the attack model. Then, we define four strategies that an attacker could employ to partition a blockchain network and evaluate the efficacy of each strategy.

Furthermore, we study the effect of having nodes that are located within the same Autonomous Systems on the resilience of the system against attacks.

Attack Model. An adversary may have various incentives to attack a blockchain system. In this work, we specifically study attacks on the underlying topology of blockchain networks with the goal of impairing the main functions of the network. Specifically, we define the following two goals of an attacker:

1. Network partitioning: to force the overlay into two or more network partitions. A network partition is the decomposition of a network into independent subnets, so that no information flow between the partitions is possible due to the absence of links between nodes.

2. Disturb the information propagation mechanisms by introducing intolerable delays. Such delays can typically increase the time to reach consensus among all participants and create a split in the application layer of a blockchain system. In fact, propagation delays are known to be key contributors to blockchain forks [22]. Garay *et al.* [39] perform a theoretical analysis of the relation between network synchronicity and both stability and adversarial power.

Such attacks would impair the main functions of a blockchain network, potentially causing a decrease in users' trust in the system. Attackers with external incentives would be highly motivated to carry out such attacks. To measure the effectiveness of each goal, we use the following three metrics: a) the size of the largest weakly connected component, b) the number of connected components, and c) the diameter of the network.

Note: Decker and Wattenhofer in [22], demonstrated that reducing the network diameter to 2 hops results in a significant 53% reduction in fork rate (as discussed in Sections V, paragraphs C and D). Therefore, we adopt the network diameter as an intermediate metric to gauge the propagation delay.

To this end, we consider the following attack strategies:

1. Attack minimum-cut edges, in order to partition the network by removing edges that are positioned in key places in the graph.

2. Targeted attacks on unique nodes, based on a selected network metric. We test out-degree, betweenness centrality, and page-rank.

3. Random attacks using random node removal emulate failures that can occur in the network in a random fashion and are used as a baseline.

5.3.1 Minimum Edge Cuts.

To compute the minimum edge cuts, we used the algebraic connectivity of the derived graphs. The algebraic connectivity of a graph is defined as the second smallest eigenvalue of its Laplacian matrix L, $\lambda_2(L)$ and is a lower bound on node/edge connectivity [32]. Since calculating the algebraic connectivity of a graph is computationally very expensive (i.e., $>$ 3 compute hours per snapshot), we analyzed one snapshot per network. Using the computed eigenvector, we count how many edges are required to be removed to split the network in two parts, and compute their sizes and ratio of the two subnets (cut-ratio, computed as largest subnet over the total). The results are presented in Table 5.4. Most cuts are highly unbalanced. Bitcoin Cash has an almost perfect cut, although removing a large fraction of edges (6.5% of edges or 10k edges out of total). Bitcoin and Zcash are somewhat affected by removing less than 0.5% of their network edges. Overall, targeting minimum cut edges does not have a significant effect on the networks' state and would require the removal (or disruption) of a considerable number of edges.

Table 5.4: Resilience of synthesized graphs in edge and node removal when attacking minimum-cut edges.

	Bitcoin	Bitcoin Cash	Dash	Dogecoin	Ethereum	Litecoin	Zcash
Edges Removed	5545 (0.1%)	10603 (6.5%)	1451 (0.02%)	581 (0.44%)	2220 (2.71%)	544 (0.08%)	363 (0.33%)
Network Split	9964/ 43949	11367/ 11895	46/ 8556	11/ 1069	436/ 15345	37/ 6576	258/ 1231
Cut Ratio	0.815	0.511	0.995	0.990	0.972	0.994	0.827

5.3.2 Targeted Node Attacks

The removal of a node simultaneously cuts all its adjacent links, therefore, it is more efficient for an attacker compared to the removal of targeted links. We focus on how to remove nodes in the most efficient way to minimize the number of node removals necessary to cause a disruption. A node can be removed from the network through various means, including DoS attacks. We follow a static procedure in the sense that each node is given a static priority of removal, based on a chosen metric. For example, when using the out-degree metric, the higher the degree, the greater the importance of the node to be attacked. After removing a node, the priorities are not recalculated. We remove only **reachable** nodes from the network one by one, following the given priority. After each removal, we calculate the size of the largest connected component and the approximate diameter of the resulting graph. We report the effectiveness of the three node ranking metrics (betweenness centrality, out-degree, and page-rank), and compare with the baseline random removal strategy. We perform the procedure described on all 24-hour-long snapshots per blockchain. Due to the high number of graphs collected, we stopped execution after removing 12% of nodes per snapshot.

The results are plotted in Figure 5.4, where the x-axis reports the percentage of nodes removed. In the upper part we plot the diameter of the network. Lower part is the size of the largest weakly connected component (WCC). The lines correspond to the median value across all snapshots and the shaded area indicates values between 1st-3rd quartile. The different colours correspond to the network metric used: Orange x: Out-degree of unique nodes; Beige +: Random unique nodes; Green *: Betweenness of unique nodes;

As can be seen in Figure 5.4, in Bitcoin and Bitcoin Cash, the betweenness and the out-degree

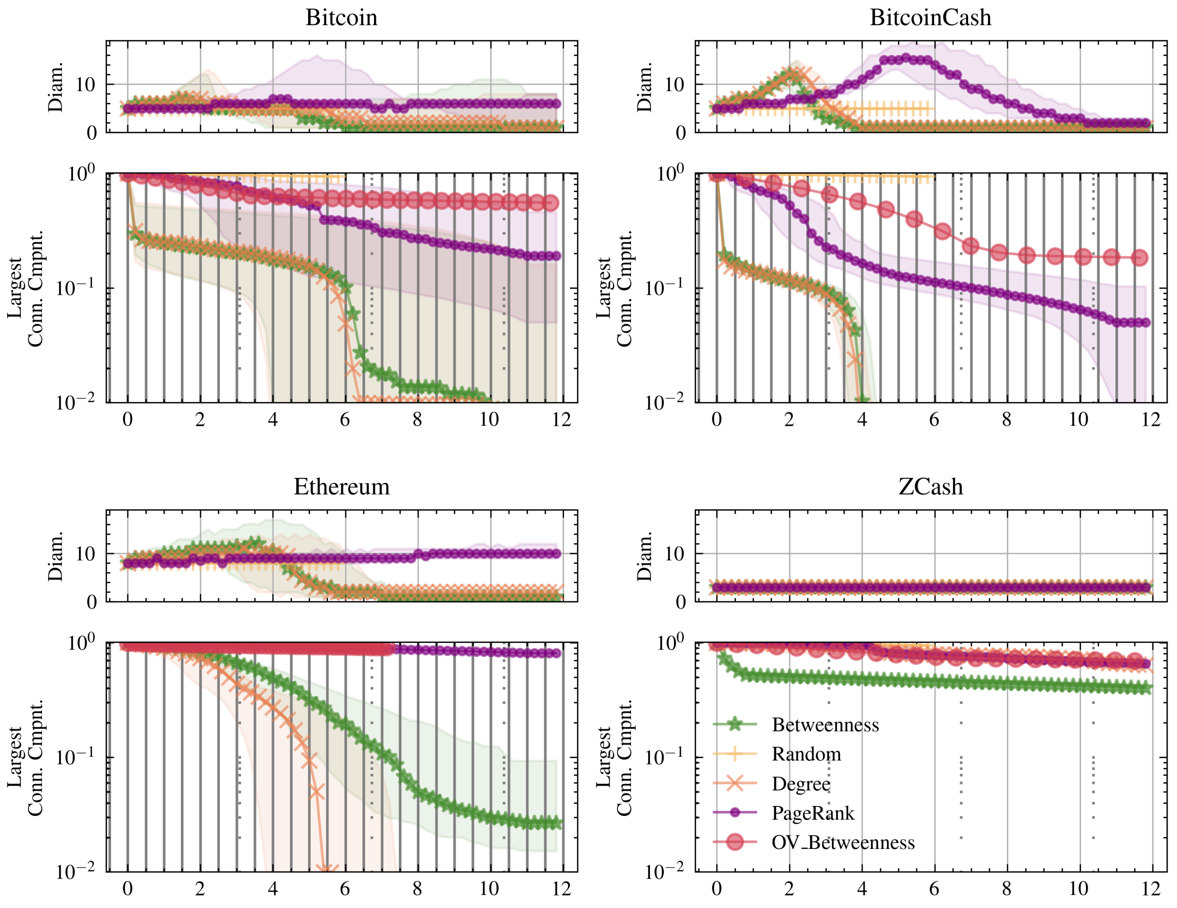


Figure 5.4: Evolution of the approximate diameter (upper part) and size of largest WCC (lower part). X-axis reports percentage of nodes removed.

strategies have roughly the same shape. The size of the largest connected component decreases significantly after the removal of only a few nodes. Further removal of nodes gradually shrinks the size to a threshold where the connected component abruptly falls to 1% of its initial size. This occurs after removal of 6% and 4% of the nodes, respectively. Similar behavior has also been found on the Internet [69]. This finding may not seem very concerning, since the reported percentages correspond to a few thousand nodes. However, closer inspection (shown in Figure 5.7) of these two networks unveils that the removal of the first five nodes reduces the size of the largest connected component by 60%, which is rather alarming. Unlike the size of the largest component, the diameter of the network starts to increase earlier in this process. This is more pronounced in Bitcoin Cash.

A visual representation of the graph before and after the removal of the top-6 nodes in a Bitcoin snapshot is shown in Figure 5.8. High betweeness nodes are highlighted in pink in the

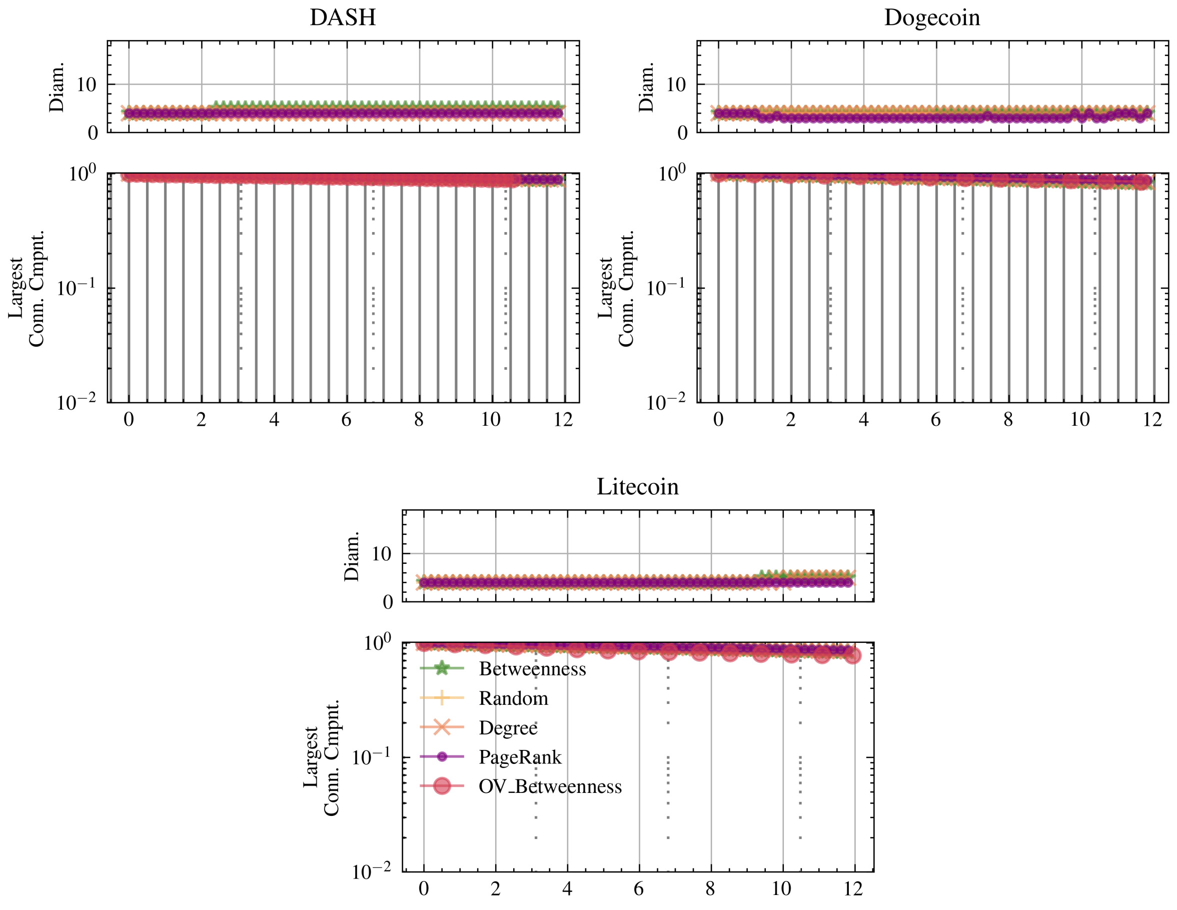


Figure 5.5: Evolution of the approximate diameter (upper part) and size of largest WCC (lower part).

top figure. Their removal results in the disconnection of a large part of the network, indicated by the gray (disconnected) nodes in the bottom figure.

In Ethereum, the out-degree strategy is more potent. Unlike Bitcoin and Bitcoin Cash, the size of the largest component does not drop initially. After removing 2% of the nodes, the size gradually drops to a threshold, close to 5%, where its size abruptly drops to 1%. The diameter of the network starts to increase early, but not as quickly as in Bitcoin Cash.

When targeting high-betweenness nodes in Zcash, the largest component's size initially falls abruptly. Similarly to Bitcoin, the first removal of nodes reduces the largest component by 40%. When 4% of the nodes are removed, the largest component drops to 50% of its initial size, and then shrinks almost linearly. Targeting high out-degree nodes is less damaging in Zcash. More than 5% of the nodes must be removed to observe a 20% reduction of the largest component.

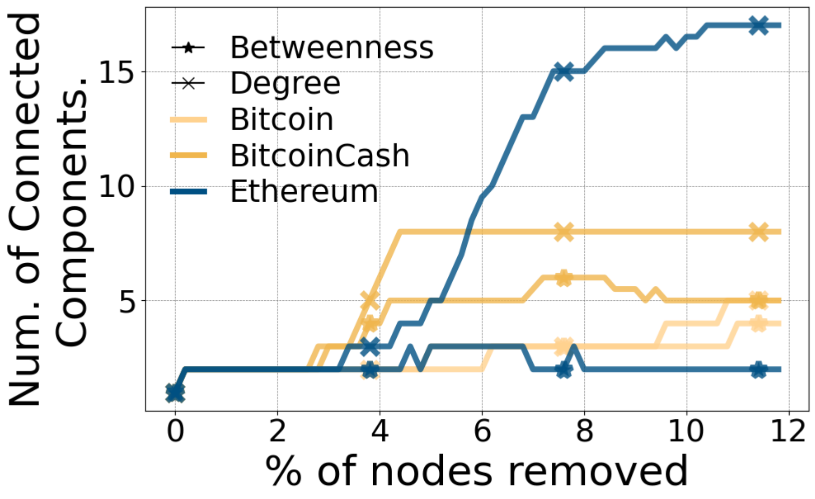


Figure 5.6: Evolution of number of connected components during the same experiment as with Fig.5.4

The number of connected components for Bitcoin, BitcoinCash, and Ethereum during the same experiment is plotted in Figure 5.6. We cannot observe a notable rise in the number of connected components, until the networks are significantly diminished.

Dash, Dogecoin and Litecoin seem equally resilient to random and targeted attacks (see Figure 5.5). The size of the largest component drops linearly with the number of nodes removed, and their diameter is not significantly affected. We can attribute their resilience to their structural characteristics discussed previously (see Section 5.1.1). All three networks have a very large strongly connected component, high clustering, and high average degree (Table 5.1). We summarize our findings in Table 5.5.

Discussion A closer examination of the partitions created by the removal of nodes in a targeted attack reveals that most of the disconnected nodes are actually unreachable, meaning that the nodes that were selected for removal acted as a bridge between reachable and unreachable nodes. The remaining connected core consists mainly of reachable peers. This pattern holds until the largest connected component is nearly 10% of its initial size. At this point, nodes belonging to the reachable part of the network also begin to disconnect. From this point on, the three major networks are greatly disrupted and eventually break apart. We readily admit that it is possible to miss connections from unreachable peers towards reachable peers. This resilience assessment is based on the assumption that these links constitute a small minority of all possible links. Our validation results in Section 4.4 support this assumption.

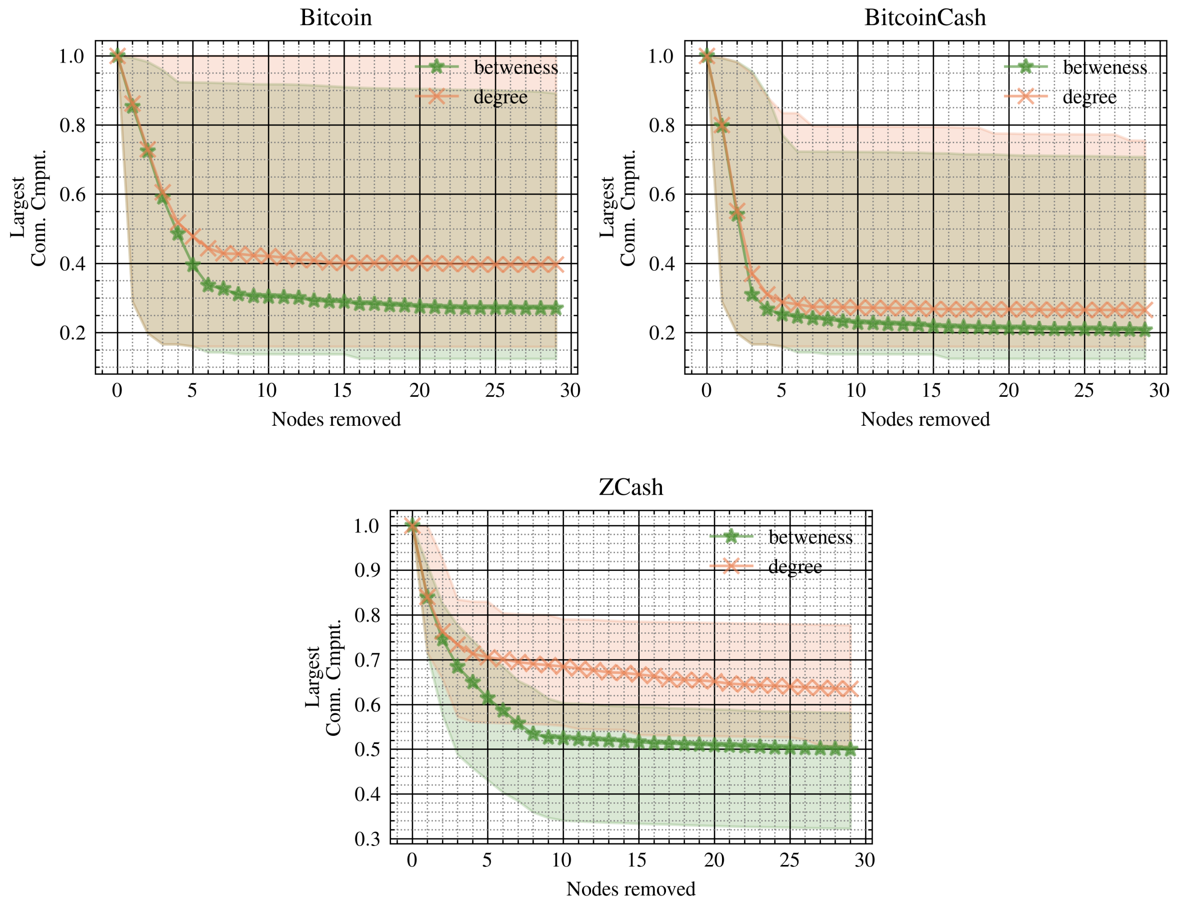
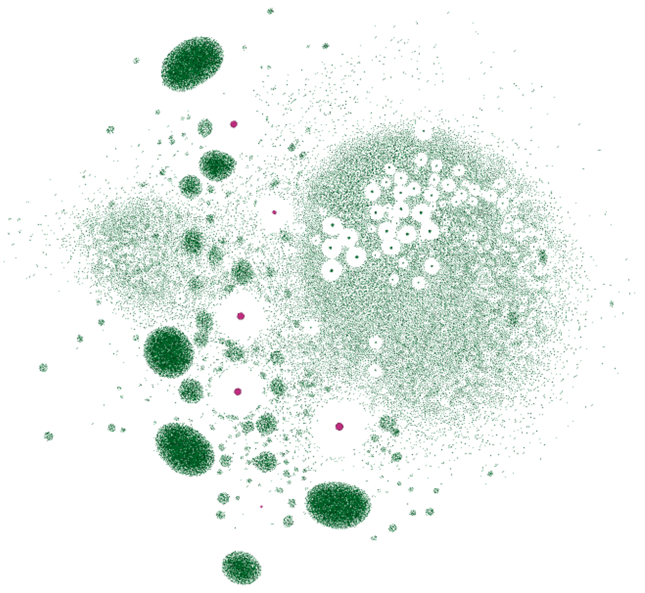


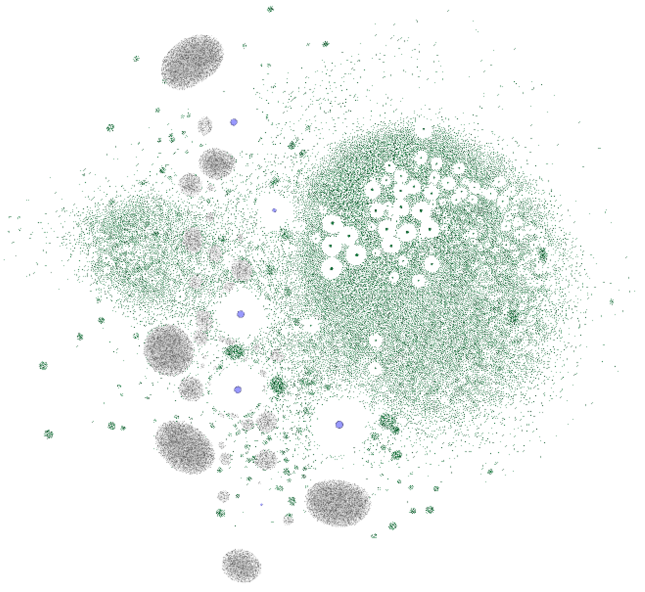
Figure 5.7: Evolution of the largest WCC. X-axis reports number of nodes removed. Orange x: Out-degree of unique nodes; Green *: Betweenness of unique nodes.

This assumption is also supported by [49], which estimates an average degree of 9.8 for unreachable peers on the Bitcoin network. For context, our analysis estimates an average degree of 37, indicating that our methodology does not miss connections between unreachable and reachable peers. Furthermore, in [101] Wang and Pustogarov estimate that unreachable peers establish only 3.5 connections to the network, on average. Interestingly, they also find that such unreachable peers are not merely disposable nodes of the network. Instead, they are involved in the propagation of 43% of Bitcoin transactions. Furthermore, comparing these results with Table 5.3 it is obvious that the in-degree of unreachable is related with the resilience of a network under a targeted attack.

Our resilience study demonstrates that blockchain overlays are resilient against random failures, but targeted attacks can considerably affect their connectivity. Attacking a handful of key peers has the potential to disconnect a large number of unreachable peers and thus can



(a) Initial condition



(b) Network after removal of nodes

Figure 5.8: Bitcoin Snapshot: Top-6 Betweenness Centrality Nodes Removed
Edges omitted for clarity. Nodes with high-betweenness are colored pink. Gray clusters indicate nodes outside the largest connected component. Nodes removed are colored blue.

Table 5.5: Resilience of graphs to targeted node attacks. We report the number and percentage of nodes that, when removed, reduce the largest component to 50% and 1% of its initial size, respectively.

Network	Bitcoin	Bitcoin Cash	Dash	Dogecoin	Ethereum	Litecoin	Zcash
# of Nodes (50% reduction)	10	10	-	-	300	-	6
% of Nodes (99% reduction)	6.5%	4%	>12%	>12%	5.5%	>12%	>12%

severely affect message propagation in the network.

Note: We acknowledge that, at specific instances, certain nodes may assume more significant roles in distributing information throughout the network. This could include relay nodes responsible for broadcasting newly discovered blocks to a substantial portion of the network, or a miner who just discovered a new block. However, in our analysis, we treat all nodes as equally important. As highlighted in Section 4.8, the decision to treat all nodes uniformly is a deliberate choice made to simplify the analysis and focus on structural characteristics that can have broad implications for blockchain overlay networks. Although it is true that some nodes may be more important than others at specific timestamps, the presented assessment provides a comprehensive overview of the network's structural vulnerabilities. Attempting to assign varying degrees of importance to the nodes at each timestamp would introduce complexities that could obscure the broader structural patterns of the network. By treating all potential nodes and connections evenly, our aim is to identify common structural weaknesses.

It is important to note that the observed network partitioning is based on the observed structural characteristics within the limitations of the chosen methodology. While the severity of network partitioning may vary at different timestamps, the focus of this research is not solely on the immediate impact but on understanding the fundamental vulnerabilities that can lead to partitioning. Our claims about network partitioning are based on observed structural weaknesses and aim to highlight potential areas for improvement.

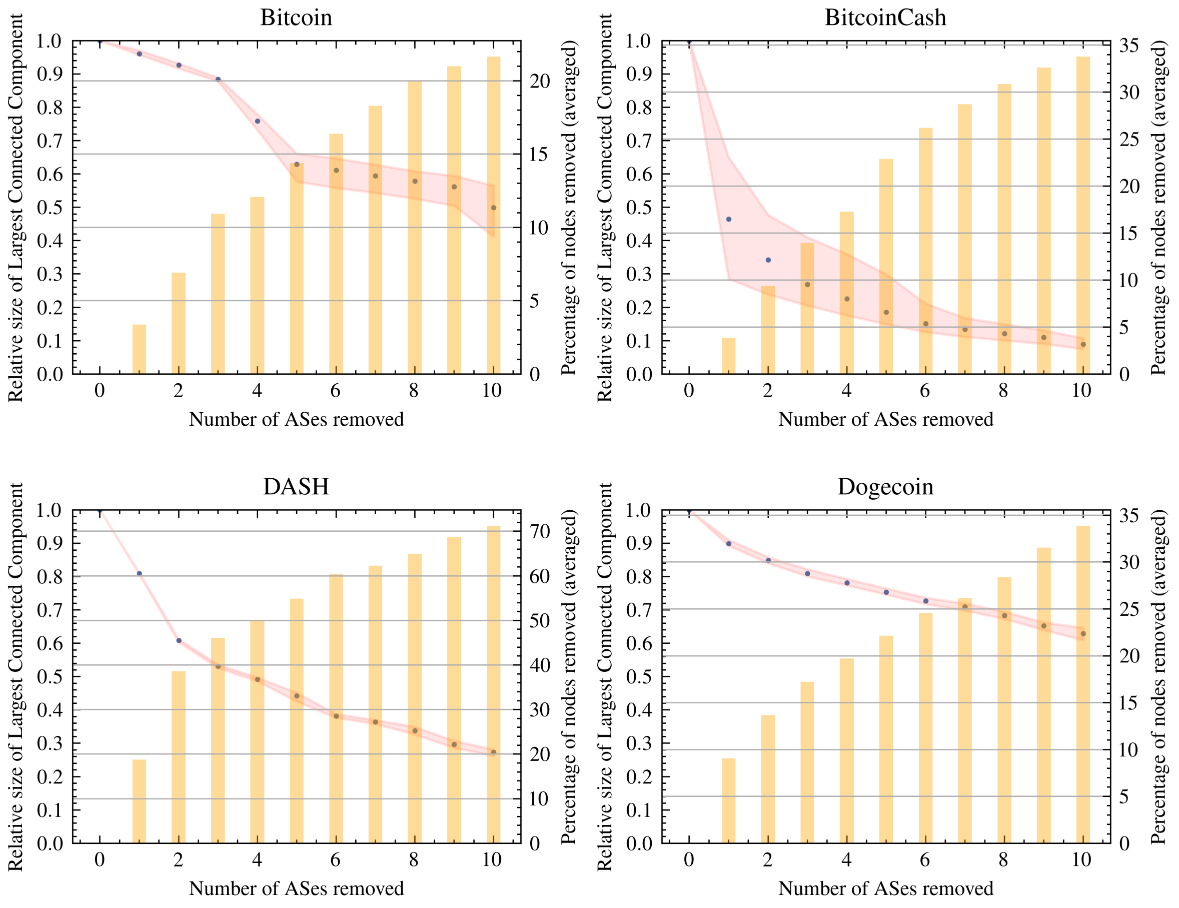


Figure 5.9: Targeting selected ASes. X-axis reports number of ASes removed. The Y-axis on the left reports size of the Largest Connected Component (blue dots). Right Y-axis reports the (average) percentage of nodes removed (yellow bars).

5.3.3 Spatial Study of blockchain overlays

As already pointed out in published research [6, 89], BGP routing attacks can be mounted against Bitcoin by leveraging the fact that a high percentage of Bitcoin nodes reside in only a small number of Autonomous Systems (AS). We also verify this result by mapping the collected IP address to ASes using the `https://ip-api.com` API. Going a step forward, we were able to identify a single AS that hosts 20% of highly connected Bitcoin nodes in all timestamps, making it a strong candidate for such attacks. In more detail, we identify the 100 highest connected nodes in each snapshot. We then look at the distribution of these nodes in ASes. Our results are summarized below.

1. 20% of the top Bitcoin nodes are located in a single AS.

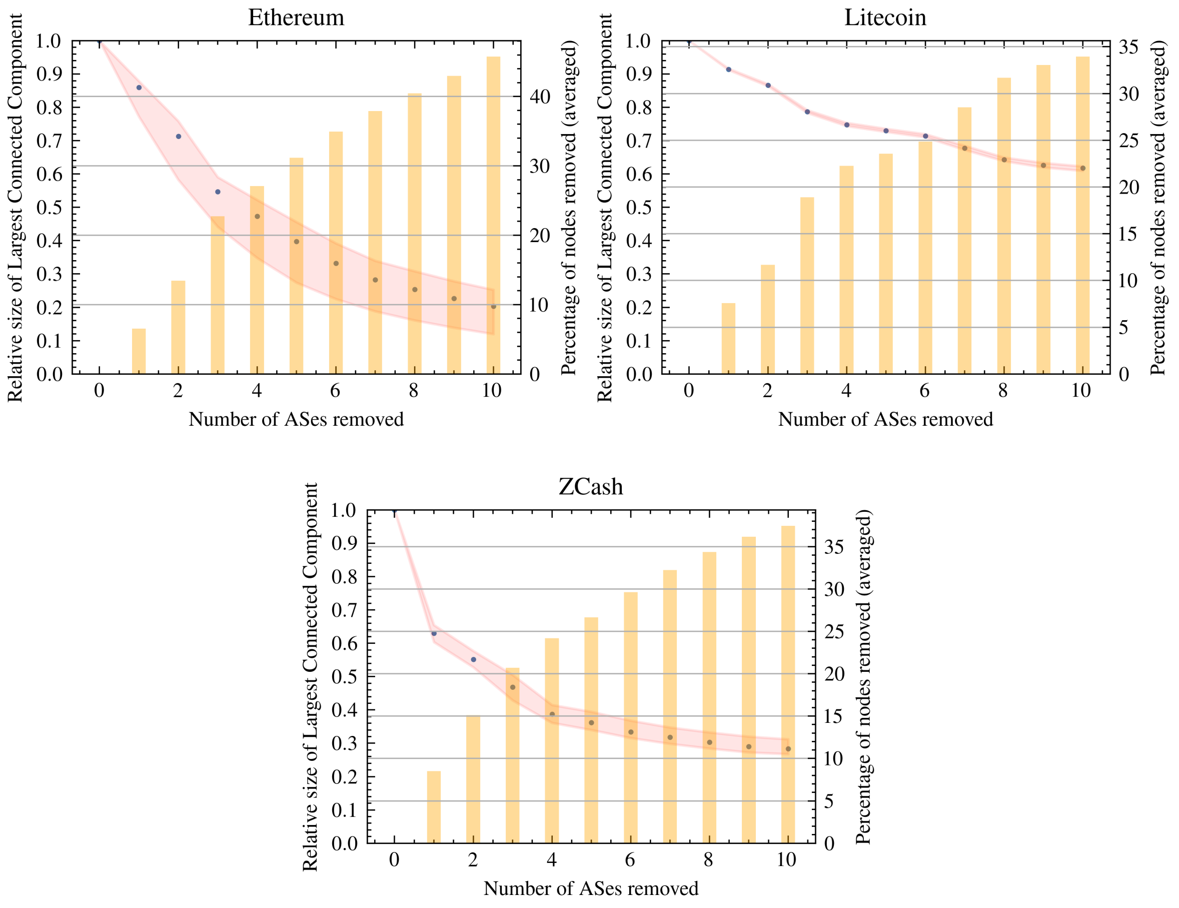


Figure 5.10: Targeting selected ASes. X-axis reports number of ASes removed. The Y-axis on the left reports size of the Largest Connected Component (blue dots). Right Y-axis reports the (average) percentage of nodes removed (yellow bars).

2. A single AS hosts a significant number of highly connected nodes in all blockchains (see Sec. 5.4).

3. Ethereum's top clients are spread in more than 550 ASes and has the most wide distribution. Bitcoin nodes are spread in 200 ASes, BitcoinCash/Dash/Dogecoin in 160 and Zcash/Litecoin in 65.

To measure the effect of targeted attacks against Autonomous Systems, we did the following test. For each snapshot, we identified the top 10 ASes with the highest geometric mean of out-degree. Then we simulated the effect of an attack against these ASes by removing all collocated nodes. The results are plotted in Figures 5.9 and 5.10. Blue dots correspond to the relative size of the largest connected component, on the left y-axis (we report median values across all snapshots). The shaded area indicates values between 1st-3rd quartile. The yellow bars indicate the percentage of nodes removed (averaged) and correspond to the scale on the right y-axis.

Notably, these plots reveal the high centralization of blockchain nodes in the same Autonomous Systems, an observation made by previous works as well. Interestingly, all networks are sensitive to such attacks, mainly due to the centralization of nodes. This is true for DASH, Dogecoin, and Litecoin, where a single AS hosts 20% of each network's nodes. On the contrary, Bitcoin is less affected by this strategy (compared to attacking individual nodes), indicating that high-degree nodes are scattered in different ASes. Note that results may differ using a different selection strategy.

5.4 Dependency in Blockchain Overlays

In this section, we address the second research question *RQ3*: Are there network entities (peers, links) that participate in more than one blockchain network, concurrently? What are their properties in comparison to entities appearing in only one network? Finally, how do these common entities impact the resilience of overlay networks?

Chatzigiannis *et al.* [16] showed that miners can distribute their computational power over multiple pools and PoW cryptocurrencies to reduce risk and increase profits. Despite [16], there are no other indications that peers in blockchain systems participate in more than one cryptocurrencies at the same time. Of course, it would not come as a surprise to discover end-users being present in multiple networks. However, this has not been observed or reported for participating peers so far.

To make it easier for the reader to understand the analysis that follows, we have provided a summary of the notation used in Table 5.6.

We define as *overlapping nodes* those nodes that participate in more than one network at the same timestamp. The intuition of our analysis is as follows. In each snapshot, we compare the set of *overlapping* nodes with all the other nodes, in order to draw insights on *overlapping* nodes' properties. Before describing the details of our study, we outline our mathematical notation to help explain our analysis. As mentioned earlier (cf. Section 4.5), C is the set of blockchains under study. Notation S_c^t represents a snapshot of a blockchain network c, at t timestamp.

Table 5.6: Notation summary

Notation	**Meaning**
C	Set of blockchain networks.
t	Timestamp.
S_c^t	Network Snapshot of network c, at time t.
S^t	Network Snapshots of *all* networks, at time t.
G_c^t	Group of nodes that belong only in S_c^t.
$G_c^{'t}$	Group of nodes that belong in a network snapshot of network c and at least an other network $c^{'}$, at timestamp t. Ie., nodes that participate in more than one networks at the same timestamp.

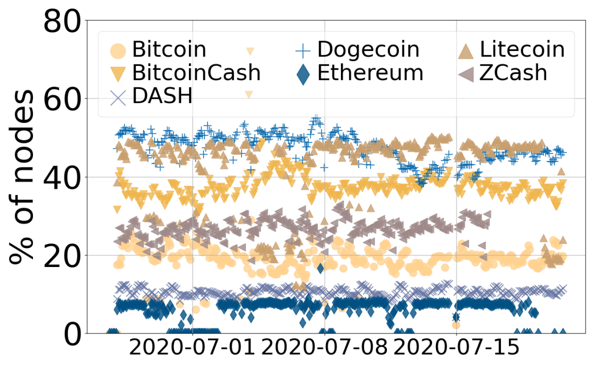
Table 5.7: Edge and Node overlaps (aggregated). *ON*: number of networks where a unique entity (node or edge) was found to be overlapping, regardless of time.
Edges are identified by their end-points.

	ON=2	*ON*=3	*ON*=4	$ON >= 5$
Nodes	34814	3909	1489	779
Edges	143577	11034	1958	222

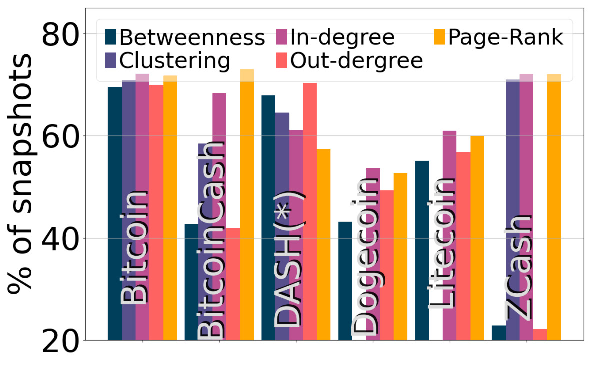
We define the set **S** as our collected data-set, that consists of all snapshots S_c^t. We denote as S^t the subset of **S** that contains all networks at timestamp t. Subsequently, for each snapshot $S_c^t \in S^t$ we define two groups, G_c^t and $G_c^{'t}$, such that $G_c^t = S_c^t - G_c^{'t}$. The first set, G_c^t, is constructed such that $\forall$ nodes $n \in G_c^t, n \notin S_{C \setminus c}^t$. That is, set G_c^t contains the nodes that participate only in blockchain c at timestamp t. Conversely, set $G_c^{'t}$ contains the *overlapping nodes*; those that participate in blockchain c **and at least one other blockchain** $c^{'} \in C \setminus c$, at the same timestamp t.

A first approach in finding overlaps between networks is to look at our aggregated data set, **S**, and count how many nodes and edges (i.e., pairs of endpoints), appear in more than one network, regardless of time. Table 5.7 shows the summary of these results. Evidently, there exists a significant number of network entities (both nodes and edges) that reside in more than one blockchain network.

The second step is to investigate whether overlapping entities occur frequently or sporadically over time. For this, we count all the overlapping peers in each S_c^t. In Figure 5.11a, for each blockchain network c, we plot the ratio of $|G_c'^t|$ over $|S_c^t|$, i.e., the number of overlapping peers in snapshot c to the total number of nodes in the same snapshot. Our observations show that in all networks, there is a consistently high percentage of nodes that overlap and belong to more than one blockchain network. Based on this and the previous results, we can conclude that there is significant overlap between blockchain overlays and that this overlap occurs consistently over time.



(a) Overlaps



(b) KS-Test results

Figure 5.11: a) Percentage of nodes that were found in more than one blockchain network at the same timestamp. X-axis indicates the timestamp. b) Percentage of snapshots where the KS-test indicates a significant distance between the distributions of the overlapping and non-overlapping nodes.

5.4.1 Structural Properties of Overlapping Nodes

In this section, we study the properties of the overlapping nodes, compared to the rest of the network. The main idea is to identify differences between sets G_c^t and $G_c'^t$ that can be supported by statistical significance. For the next analysis, we focus on the graph metrics calculated in Section 5.1.1. Specifically, for each network, we compared distributions of in-degree, out-degree, betweenness centrality, clustering coefficient, and page-rank. As we show in Section 5.1.1, we found that some of these metrics (degree distributions) are highly skewed. For this reason, we performed a normality test on all metrics to decide on the statistical method to be used for the comparison. The normality test confirmed that the distributions of all metrics are in fact not normal. Normality was checked by performing the Shapiro-Wilk test, provided by the SciPy package [99]. Since we wish to compare distributions of non-normal data, a non-parametric test is needed. Using the same package, we performed the 2-sample Kolmogorov-Smirnov test (KS-test) between all pairs of G_c^t and G'_{c^t}, for all

$c \in C$ and all timestamps. The KS-test returns the test statistic D, which is the maximum distance between the cumulative distributions of the two samples. It also returns the *p-value* for the hypothesis test. If the test statistic D is small, or the *p-value* is higher than the selected statistical level α (e.g., 0.05), then we cannot reject the null hypothesis that the distributions of the two samples are the same. Our results indicate that in all networks, there is a significant distance between the metrics' distributions among groups G_c^t and $G_c^{\prime t}$. Interested to see if there exists a metric that describes this difference better than the others, we looked into our results for the metric that gives significant D values consistently. In Figure 5.11b, we plot the percentage of all snapshots of a given network S_c^t where the p-value of the test is small enough, i.e., *p-value*< 0.05, and $D > 0.1$. The plot indicates that the distributions between the two groups are often non-equal. We can also observe that in most networks the in-degree and page-rank metrics are the ones found more frequently, meaning that these metrics capture the differences between G_c^t and $G_c^{\prime t}$, more often than the other metrics. We also found that the CDFs of overlapping nodes are lower than non-overlapping nodes, meaning that the metrics of overlapping nodes are statistically higher. An exception in this finding was the Dash network (indicated with an * in Fig. 5.11b), where the opposite is true: in Dash, overlapping nodes have lower metrics than the rest. A key takeaway from this test is that overlapping nodes have different characteristics. Although this test cannot serve as a proper classifier, it answers our question that overlapping nodes have, in fact, different properties than the rest of the nodes in a blockchain network.

5.4.2 Attacking Overlapping Network Entities

The final part of our research question *RQ3* asks how the presence of overlapping nodes can impact the resilience of blockchain overlays to targeted attacks. In order to answer this, we repeat the test of the previous Section (5.3.2) with a small variation. From each set $S_{c \in C}^t$, we remove all G_c^t sets. This new set, $S_{c \in C}^{\prime t}$, contains all nodes that participate in more than one networks at the same time timestamp.

We then sort the unique elements of $S_{c \in C}^{\prime t}$ in descending order based on their maximum normalized betweenness centrality. Since a node can participate in more than one network, we sort the nodes based on the highest value they have across all snapshots at time t. We use the min-max method to normalize the betweenness centrality values for each snapshot. After sorting the nodes, we proceed to remove them from each snapshot S_c^t at the same time t. The nodes are removed in the same order from all snapshots.

The results of targeting overlapping nodes first, are plotted in Figures 5.4 and 5.5 with red circles. The plot reports the average change in the largest connected component over all snapshots S_c^t. Clearly, this strategy is less effective compared to the strategies used earlier, which target top central nodes within a specific network. However, it provides the benefit of attacking multiple networks simultaneously. An interesting finding is that Litecoin is more susceptible to this kind of attack compared to attacks focused on single blockchain node metrics. This is partly explained by the fact that Litecoin has one of the highest percentages of overlapping nodes (see Fig. 5.11a).

Closer inspection of the data at hand shows that an attacker is able to shrink the largest connected component of Bitcoin Cash, Bitcoin, Zcash, and Litecoin networks by 70%, 40%, 25% and 20% respectively. This demonstrates that, by targeting overlapping nodes, a powerful adversary can mount a partitioning attack against *4 different networks at the same time*.

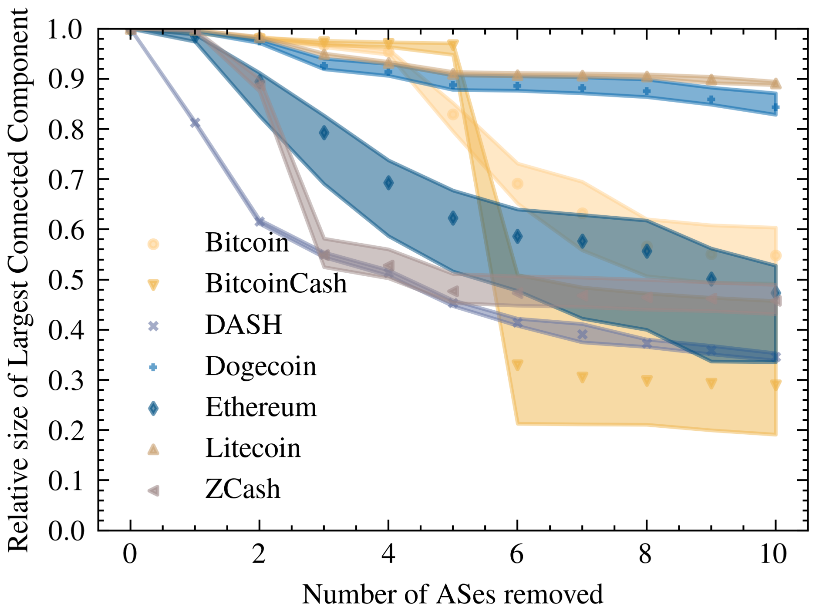


Figure 5.12: Size of the largest connected component of all networks when selected ASes are disrupted.

Another effect of overlapping nodes is shown in the strategy described next. Similarly to the selection performed in Section 5.3.3, we calculated the geometric mean of out-degree of all networks, for each AS, across all timestamps. That is, for each Autonomous System we took into consideration all nodes from all chains and calculated the geometric mean of their out-degree. We then remove each AS and calculate the effect on each network. Removal of an AS simultaneously removes all nodes (from all networks) that reside on that particular AS. The results of this selection strategy are plotted in Figure 5.12. The significance of overlapping nodes is profound. A disruption in just 6 ASes could have considerable effects in five networks at the same time.

5.5 Longitudinal Evolution

Research question *RQ4* asks about the temporal behavior of the networks and how they change over time. This section explores this issue.

Permissionless blockchain networks are open to participation, and nodes may freely come and go. This independent arrival and departure of the nodes creates the collective effect known as churn. Churn in Bitcoin has been studied by Imtiaz *et al.* [56]. A recent study by Kiffer *et al.*, explores Ethereum's overlay network in detail, including churn [61]. Stutzbach and Rejaie conducted an in-depth study of churn in P2P networks [94]. In their work, they recognize that one of the most basic properties of churn is the *session length* distribution, which describes how long each node participates in the system each time it connects. A *session* begins when a node joins the network and ends when the node disconnects.

In this section, we analyze the node dynamics observed in our collected datasets, focusing on the churn characteristics of the networks of interest. Following the methods of [94], we examine the session length distributions.

Limitations As mentioned in a previous chapter (see Section 4.3.1), the snapshots collected are two hours apart. All peers that appear in a snapshot are considered to be online for the entire period. This poses a minimum cap to our longitudinal granularity. It does not allow us to capture short-lived connections, i.e., we cannot know whether a node connected and disconnected multiple times within a two-hour period. Since disconnections during a snapshot period are missed, two sessions may look like one longer session. Moreover, very short sessions might not be observed at all, contributing to a bias towards longer-lived connections.

Additionally, due to maintenance operations, snapshots 95 and 253 were not purged to disk in a timely manner (see Section 4.3.1). This resulted in snapshots with a duration of six and nine hours, respectively. [2] We decided not to include these snapshots in the analysis presented in this section. Furthermore, to better capture any evolution dynamics, we split our dataset in 4 periods, excluding the two aforementioned snapshots. Each of the four periods has an approximate duration of 5 and a half days:

[2]snapshots corresponding to time-stamps 2020-07-03T07:03:22 and 2020-07-14T14:12:12.

- *period-0 (p0):* data from 2020-06-27 to 2020-07-02.
- *period-1 (p1):* data from 2020-07-03 to 2020-07-08.
- *period-2 (p2):* data from 2020-07-09 to 2020-07-14.
- *period-3 (p3):* data from 2020-07-14 to 2020-07-20.

In our longitudinal analysis, we decided to use the subset of reachable peers of each snapshot, i.e., peers that respond to our requests. This is preferred, since we cannot distinguish between peers that are offline and peers that are alive but behind NATs or firewalls. Finally, during the last days of our measurement period, Zcash's network protocol was updated to a new version, incompatible with the previous one. In effect, we could not capture any node in the Zcash overlay after the 16th of July 2020.[3]

5.5.1 Distribution of Session Length

In Figures 5.14-5.15, we plot the complementary cumulative distribution function (CCDF) of session length across the three periods defined above, for each overlay.

The distributions of session lengths in Dash, Dogecoin, Litecoin, and Zcash remain consistently similar across all three periods, indicating a stable pattern over time. This consistency is also observed among the distributions of Dogecoin, Litecoin, and Zcash, highlighting the uniformity of session behaviors across these networks. In essence, the data suggests that session length distributions do not undergo substantial changes over time, and this trend is shared among the various networks analyzed.

In contrast, in Bitcoin and BitcoinCash, we observe that session lengths change over time. Nevertheless, across the two systems, the distributions are consistent, with Bitcoin having a small percentage of nodes with sessions longer than BitcoinCash.

In Ethereum, we observe that most sessions are short. In fact, fewer than half of the sessions are less than six hours. A similar observation was also made in [55]. There exists an obvious shift in session length distribution between the first and the second period, which remains unchanged during the third period. Finally, Dash has the higher percentage of long session lengths with 80% of session lengths lasting for more than 4 days.

[3]Zcash-Heartwood: `https://z.cash/upgrade/`

Prior studies[56] suggest that session lengths exhibit a behavior similar to a heavy-tailed distribution. To test how well the session length distribution of each network snapshot can be modeled by a *power-law* (*PL*), *log-normal* (*LN*), *power-law with exponential cutoff* (*PLEC*) or *stretched exponential* (*SE*), we calculate the best fit using the *powerlaw* package available by Alstott *et al.* [4]. In Table 5.8, we report the best fit for each period, for all networks. Dash is best modeled by a stretched exponential in all periods, while Dogecoin and Litecoin are best fitted by a power-law with exponential cutoff. The rest of the networks are mostly fitted by a power-law with exponential cutoff.

To summarize, blockchain networks differ in their temporal characteristics, with some networks sharing some similarities. The largest networks in our study, namely Bitcoin, BitcoinCash, and Ethereum, have observable differences in their session-length distributions between different periods. In these networks, while most sessions are short (less than a day), some sessions are a week long. The observed data in most networks are better described by power-law distributions with exponential cutoff.

Table 5.8: Session Length best-fit.
PL:power-law; *PLEC*: power-law with exponential cutoff; *SE*: stretched exponential.

	Bitcoin	Bitcoin Cash	Dash	Dogecoin	Ethereum	Litecoin	Zcash
period-0	SE	SE	SE	PLEC	PLEC	PLEC	SE
period-1	PLEC	PLEC	SE	PLEC	PL	PLEC	PLEC
period-2	PLEC	PLEC	SE	PLEC	PL	PLEC	SE
period-3	PLEC	PLEC	SE	PLEC	PLEC	PLEC	SE

Correlation between node degree and session length

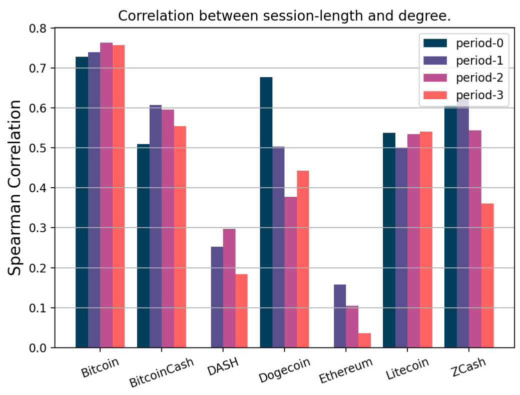
To answer whether a node's session-length correlates with its degree, we calculated the Spearman correlation [64], implemented by the SciPy [99] package. The Spearman rank-order correlation coefficient is a non-parametric measure of the monotonicity of the relationship between two data vectors. In Figure 5.13a we plot the calculated correlation coefficients between the duration of the node session and the degree of the node for each overlay, per period. All correlation coefficients are statistically significant with p-values lower than 0.02, except for the first periods of Dash and Ethereum, which are not plotted. The coefficients vary between the four periods and their values point towards moderate to strong correlation in all overlays. In Ethereum and Dash we observe the lowest correlations. We further study the correlation in Ethereum by breaking each period into groups of *low* and *high* degree nodes and computing the correlation coefficients per group. The resulting correlation coefficients for higher-degree nodes are again positive and higher, between 0.26 and 0.52. The coefficients for the set of lower-degree nodes are negative, indicating a nonlinear correlation between up-time and node degree in Ethereum. Similar observations are made in Dash as well, with high-degree nodes having strong positive correlation and low-degree nodes with moderate negative correlation.

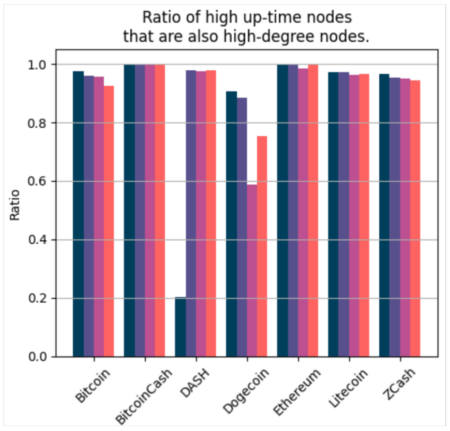
Further inspection of the data reveals that nodes which are connected longer are almost always high-degree nodes. This is somewhat expected as the longer a node is connected, the more peers it is able to discover. In Figure 5.13b, we plot the ratio of high session-length nodes that are in the top 10% high-degree nodes. For each snapshot, we extract the top 10% nodes ranked by their out-degree. We then set the least up-time of these nodes as a threshold. By selecting the nodes with an up-time longer or equal to that threshold, we calculate the Jaccard similarity of between the two sets of nodes, namely *Long-session nodes* and *High-degree nodes*. This leads us to expect similar results in the networks' resilience. The observed relation between a node's up-time and degree could be used to locate nodes of interest in settings that employ topology hiding features, as is the case with the latest Bitcoin reference client

5.5.2 Targeting high-uptime nodes

In our next experiment, we wanted to determine whether targeting peers who are frequently online could be a potential threat to blockchain overlays. To do this, we selected the nodes for removal based on how long they had been online on the network. To reduce computation times, the top 2500 nodes of each network were selected. This method of selection has the advantage of not relying on any knowledge of the topology of the network, and instead focuses solely on how often a peer is observed to be online. This makes it an effective strategy for addressing the use of topology hiding techniques in blockchain overlays, as it does not rely on any information about the network's topology. The results of this experiment are presented in Figures 5.16-5.17.

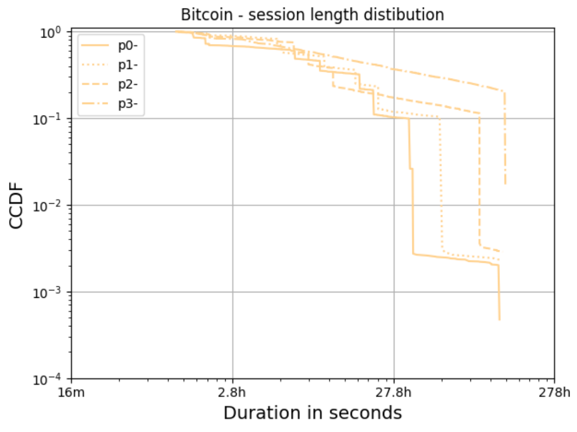
While it might be expected that targeting nodes with high uptime, which is strongly correlated with node degree, would result in greater disruption, our findings reveal a less severe impact, particularly evident in the case of Bitcoin. Although nodes with high uptime often exhibit high connectivity, there exist nodes with even higher connectivity that are less frequently connected. In other words, nodes with the highest connectivity do not necessarily possess the highest uptime; yet they are more likely to remain online for longer periods than the observed average. This observation is supported by comparing of the size of the largest connected component in Figure 5.4 with Figure 5.16. It is important to note that although this strategy is less severe, it remains effective, resulting in an observable reduction in the size of the largest connected component. Interestingly, in the context of ZCash, focusing on high uptime nodes appears to be even more effective than targeting nodes with high degrees.



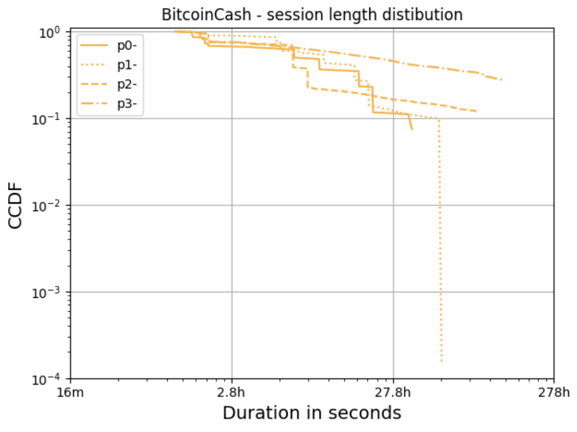


(a) **(b)**

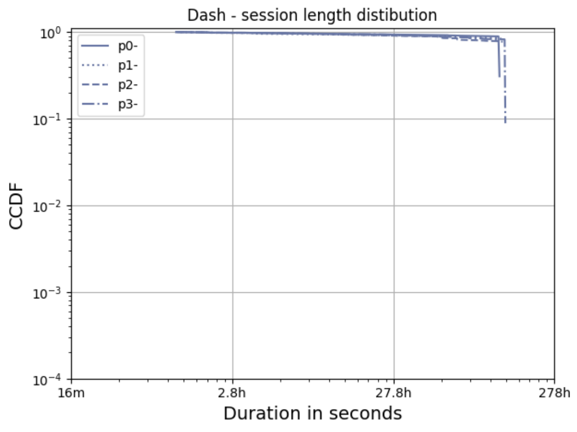
Figure 5.13: (a) Spearman Correlation between session-length and node degree. (b) Percentage of high up-time nodes that are also high-degree nodes.



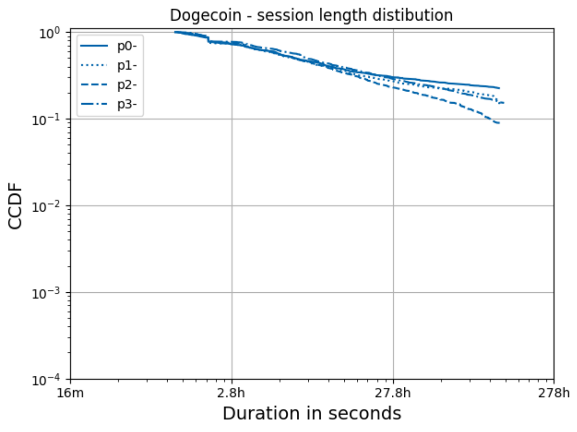
(a) Bitcoin



(b) Bitcoin Cash

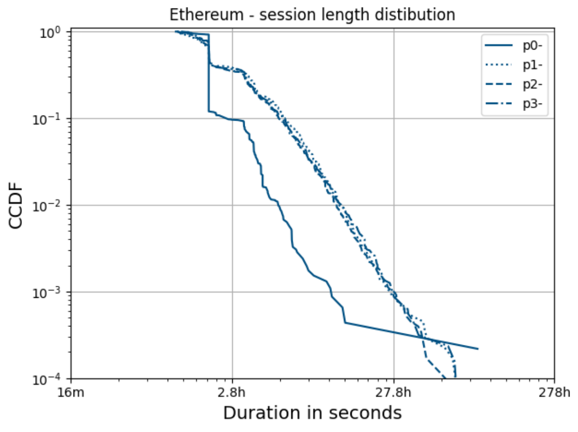


(c) Dash

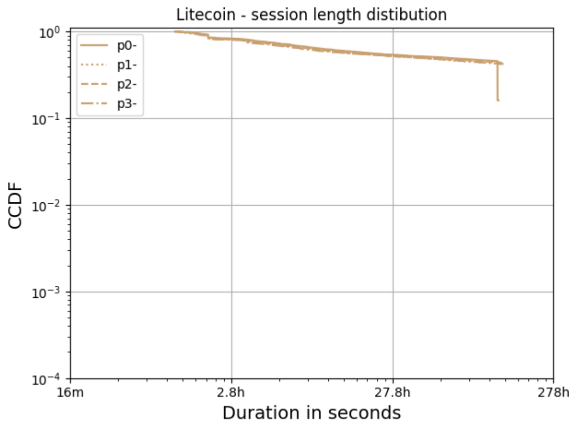


(d) Dogecoin

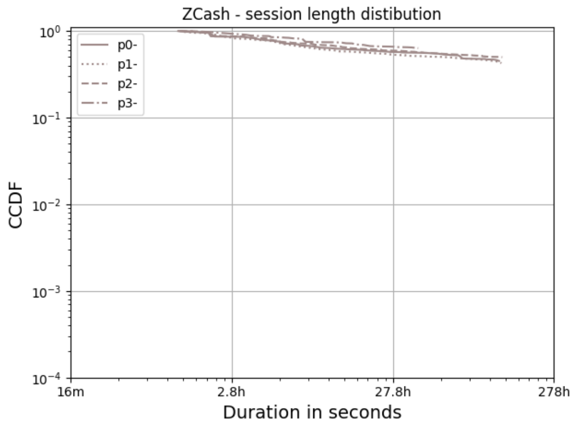
Figure 5.14: CCDF of session lengths per network studied.



(a) Ethereum



(b) Litecoin



(c) Zcash

Figure 5.15: CCDF of session lengths per network studied.

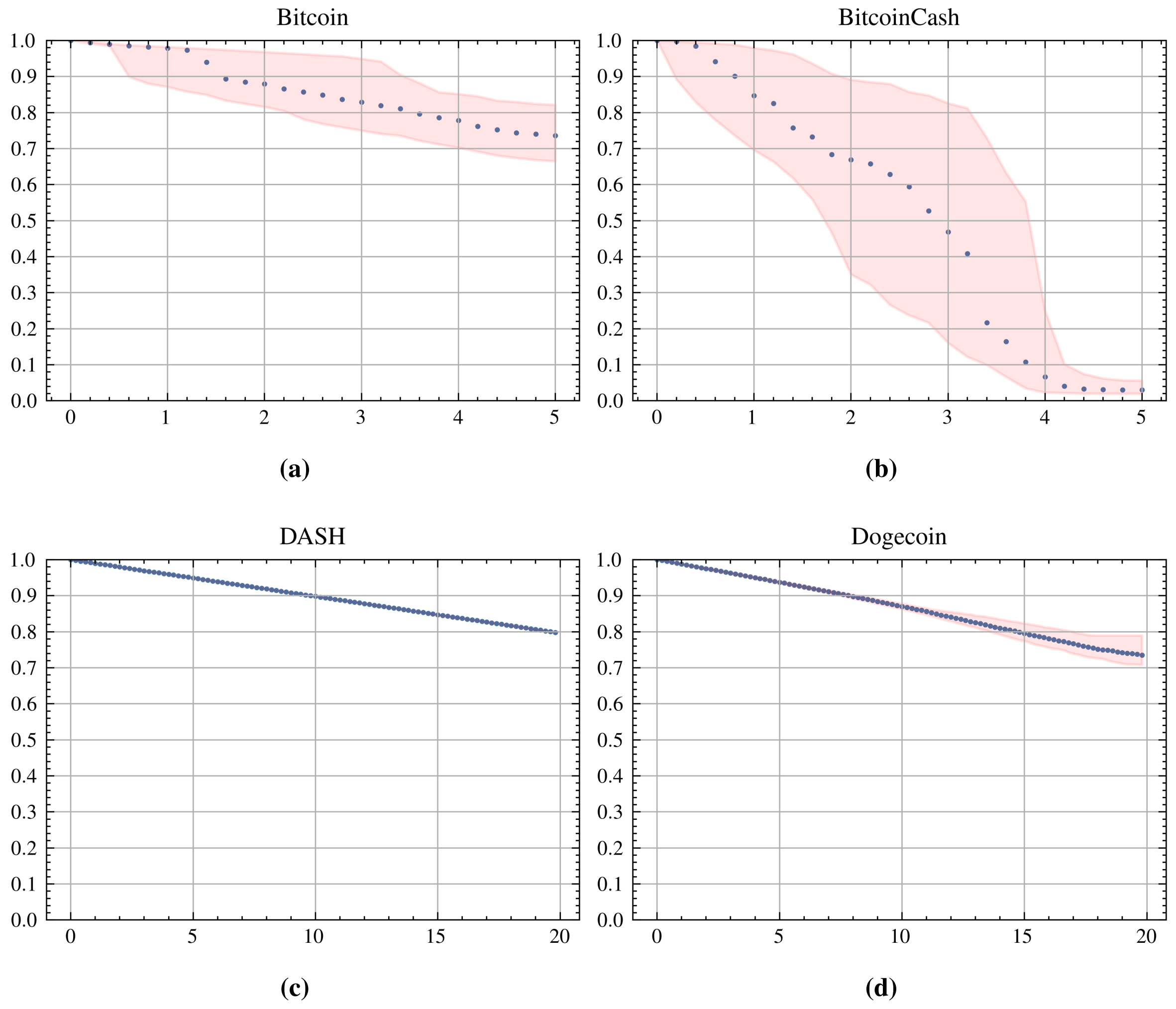
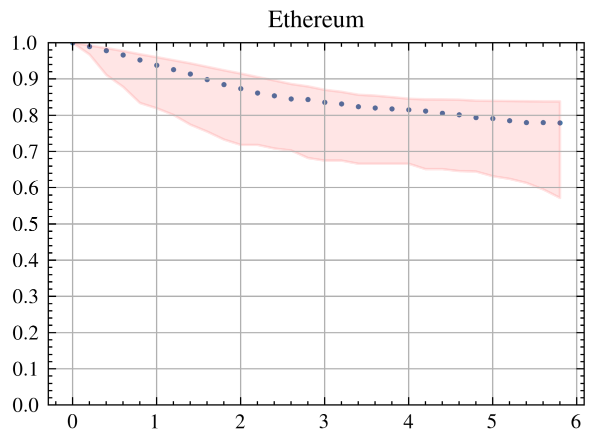
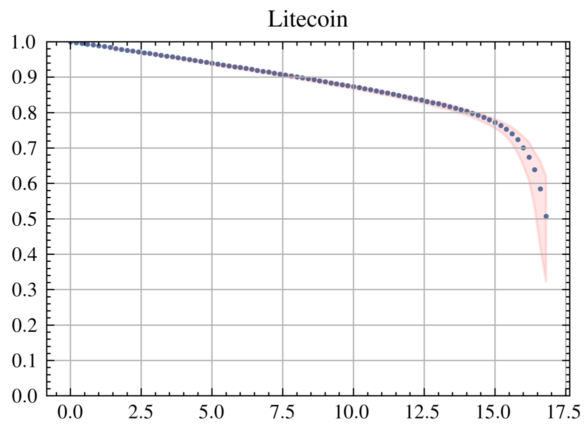


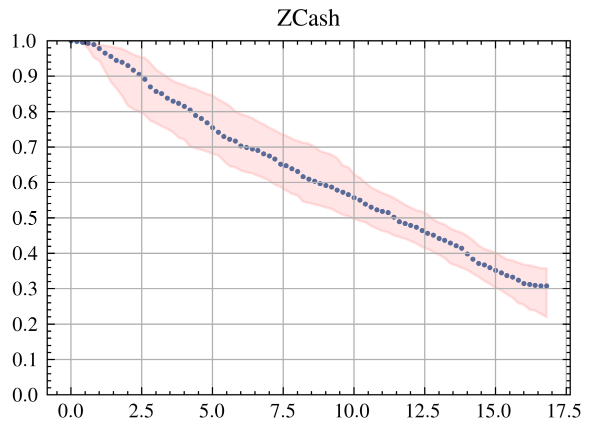
Figure 5.16: Targeting high-uptime nodes. Y-axis reports relative size of the largest connected component. X-axis represents percentage of nodes removed. Points correspond to the median value across all snapshots. The shaded area indicates values between 1st-3rd quartile.



(a)



(b)



(c)

Figure 5.17: Targeting high-uptime nodes. Y-axis reports relative size of the largest connected component. X-axis represents percentage of nodes removed. Points correspond to the median value across all snapshots. The shaded area indicates values between 1st-3rd quartile.

Chapter 6

Discussion and Conclusion

The unique properties of blockchain systems, such as immutability, public availability and verifiability, decentralization of trust, and disintermediation, make them highly desirable for a wide range of applications. Although the most successful application of blockchains so far has been in the realm of digital currencies, their potential extends far beyond this. Blockchain solutions have already been suggested for a variety of practical uses, and it is believed that they will continue to penetrate more industries in the future.

Blockchain technology is often promoted as being more secure and resilient due to its decentralized structure. Studies have indicated that security and resilience cannot be assumed to be guaranteed, as numerous attacks have been proposed in the literature. The fundamental element of any blockchain system is the underlying peer-to-peer (P2P) network that connects the active parties of the system. However, the properties of these networks are often overlooked and have not been studied in depth. The security and correctness of any blockchain application ultimately depend on the resilience of the underlying network.

To enhance their security, blockchain systems rely on topology hiding techniques. The fact that blockchains have different application requirements than previous P2P implementations (content discovery and delivery) makes the case for different network protocols. Instead, the blockchain community has relied so far on repurposing previous P2P implementations.

This thesis had the overarching goal of investigating the network layer of blockchain overlays and assessing their susceptibility to targeted attacks. Furthermore, we sought to broadly explore the ecosystem of blockchains and compare the structural properties of different networks. This is one of a few works that simultaneously study multiple blockchain networks. To accomplish this, we conducted a large measurement experiment and analyzed the gathered data, drawing insights and conclusions. We have developed a simple network monitor that can probe several blockchain overlays concurrently, allowing us to identify a large number of potential connections. The monitor is practical and scalable, as it does not require high connectivity in each network and incurs no transaction processing costs. For our analysis, we used synthesized graphs that incorporate all possible connections that could exist in the network, effectively addressing the challenges imposed by topology hiding. In particular, this study stands as a pioneering effort with a distinct focus on the resilience of blockchain networks. Additionally, we explored the ramifications of the observed temporal dynamics and the interdependencies of these networks.

In the following, we provide answers to the Research Questions laid out in the Introduction, summarize the main insights of this work, discuss its limitations, and provide possible future directions.

6.1 Answers to research questions

To aid the reader, we re-state the research questions from the introduction in this section. These questions have guided our investigation and are now addressed here.

6.1.1 RQ1:

Do all blockchain overlays share similar properties? What are their structural and network characteristics, and how do they compare to other well-known networks such as the Internet AS topology, the Web, and Online Social Networks?

From the collected data set, we observe that the synthesized graphs vary significantly in their densities. Their dynamics are manifested by significant variations found in the degree distributions per snapshot. The graphs are well connected, and their degree distributions belong to the exponential family. Despite their low diameters and small average shortest path length, we did not find evidence that these graphs satisfy the small-world property. In fact, we show that the actual networks do not satisfy this property as well.

The distribution of their clustering coefficients is similar to that of other real networks and diverges from random networks of comparable size and average degree. However, the structure we observe sets them apart from traditional technological or social networks.

One finding was that the collected synthesized graphs have different graph properties and indicate toward the conclusion that these graphs do not share a similar structure. This was somewhat surprising, since most of these networks share the same network protocols for node discovery and peer selection. One factor that may be playing a significant role in this discrepancy is the existence of unreachable peers. Their existence has been recognized, but their significance has not been studied in depth. In this work, we show that unreachable peers exist in all networks in our study, although at different percentages per network. They play an important role in blockchain systems, as demonstrated in previous work. Our work also uncovered a relation between the characteristics of unreachable peers and the networks' tolerance against targeted attacks.

6.1.2 RQ2:

To what extent are blockchain overlays prone to random failures and targeted attacks? How does the concentration of nodes within the same Autonomous Systems impact the resilience of the system?

Blockchain overlays exhibit resilience against random failures but are susceptible to targeted attacks. This observation is consistent with the behavior observed in other real-world networks, including the Web and the Internet. Interestingly, some of the networks studied here can experience significant disruptions with the removal of fewer than ten highly connected nodes, revealing an unexpected vulnerability within major cryptocurrency networks.

Remarkably, all overlays are vulnerable to at least one of the presented attack strategies, even when the abundance of network connections available to participating peers is taken into account.

Furthermore, we show that centralization of nodes in AS is another vulnerability of blockchain overlays. While disrupting multiple ASes simultaneously may be challenging, it remains achievable for determined state-nation adversaries. In fact, AS incidents are not uncommon, with nearly 13,000 BGP hijacking incidents reported in the first quarter of 2023.[1] Notably, the average duration of such attacks is close to one hour. The results presented could further enhance the potential of BGP partitioning attacks against blockchains, by focusing on ASes that host highly connected nodes.

[1]Qrator Labs. *Q1 2023 DDoS Attacks and BGP Incidents*. 2023. URL: `https://blog.qrator.net/en/q1-2023-ddos-attacks-and-bgp-incidents_171/`.

The primary goal of this question was to diagnose the problems rather than to provide comprehensive mitigation strategies. Future research can certainly build on the findings of this study by delving into mitigation strategies and defensive measures. These could include approaches such as network design improvements, enhanced peer discovery mechanisms, or consensus algorithm modifications to mitigate the impact of targeted attacks. However, such strategies are complex and often require in-depth analysis and experimentation, which could be the focus of subsequent research endeavors.

6.1.3 RQ3:

What is the connection between different blockchains when peers and links are simultaneously participating in multiple networks? How does this relationship affect the resilience of the blockchain ecosystem?

Throughout our analysis, we consistently observed a notable presence of overlapping nodes between different blockchain networks. Our rigorous statistical analysis provides strong evidence that these overlapping nodes exhibit distinct properties and metric distributions compared to the remaining nodes within a particular overlay.

In particular, our investigation revealed a significant degree of node overlap among Bitcoin, Bitcoin Cash, Zcash, and Litecoin networks. This intriguing finding suggests that a focused targeted attack on a subset of these shared nodes could potentially disrupt multiple networks simultaneously. By simulating such an attack, we demonstrate that this is, in fact, feasible. Our methodology demonstrates that a determined and methodical attacker, armed with knowledge of the central entities within the topology, could strategically target a small number of nodes to effectively impede message propagation across multiple blockchain networks.

This underscores the urgency of adopting measures to enhance the robustness of these networks or considering the implementation of open topology protocols, rather than relying solely on topology hiding mechanisms. By revealing the susceptibility of such overlays to coordinated attacks, our study emphasizes the need to strengthen network defenses and enhance resilience for the ongoing integrity and functionality of blockchain systems.

6.1.4 RQ4:

How do blockchain networks evolve over time? Are the longitudinal characteristics of these networks related to the other network properties of their peers?

In examining the temporal aspects of blockchain overlays, we uncover a dynamic nature inherent to these networks. Larger networks exhibit significant variations in the distribution of session lengths over time, while smaller networks tend to manifest more predictable distributions.

Moreover, our findings reveal a noteworthy positive correlation between a node's up-time and its degree. This discovery has significant implications, providing potential adversaries with a strategic advantage. By targeting nodes with high up-time, adversaries can exploit this correlation to execute effective attacks, bypassing the need for accurate knowledge of the network's topology.

6.2 Implications

The implications drawn from our study of blockchain overlay networks are far-reaching and hold significant relevance for the field of blockchain technology, network resilience, and security. Our findings highlight the diverse and distinct characteristics exhibited by different blockchain networks, challenging the notion of a one-size-fits-all approach in analyzing and understanding their properties. This insight underscores the importance of tailoring strategies and solutions to the specific context of each blockchain network, rather than assuming uniform behavior based on a single network's characteristics.

The inability to categorize these blockchain networks within established frameworks demonstrates the novel and unique nature of these systems. This poses a compelling challenge for future research in developing new models and theories that accurately capture the structural and behavioral aspects of blockchain overlays. Such efforts could contribute to a deeper understanding of the interplay between network dynamics, participant behavior, and emergent properties in the context of blockchain ecosystems.

Furthermore, the identified vulnerabilities and susceptibility of major blockchain networks to targeted attacks have significant implications for the security and resilience of these systems. These findings emphasize the need for proactive measures to mitigate potential threats and improve the robustness of blockchain networks against adversarial strategies. Addressing these vulnerabilities could involve the development of novel defense mechanisms, distributed consensus protocols, or network optimization strategies that strengthen network resilience while maintaining the core principles of blockchain technology.

In the past, more robust solutions for P2P networks have been proposed that prioritize security and resilience. A recent example is the work of Fanzoni *et al.* [35]. However, blockchain networks have been relying on topology hiding techniques to enhance security. Moreover, blockchain systems have different requirements from previous P2P implementations, so new innovative protocols are needed. One characteristic of blockchain systems is that they often need to handle large amounts of data and handle transactions in a decentralized manner. This means that the network protocols used must be able to handle these requirements efficiently and securely. To address these challenges, developers and solution architects must carefully consider their network implementation when building blockchain-based applications and not solely rely on the distributed nature of blockchains for security and resilience. This includes evaluating different protocols and approaches and selecting the ones that best meet the specific needs and requirements of their applications. By doing so, they can ensure that their blockchain systems are secure, resilient, and able to handle the demands of their intended use cases.

Miners and wallet providers could also enhance their services by employing additional mechanisms to ensure greater network presence and connectivity. Developers and protocol designers, based on the findings of this work, can create more realistic models for simulations and optimize the design and implementation of more robust P2P protocols.

Finally, looking towards future research prospects, our proposed unified model and methodology can serve as a valuable foundation for forthcoming investigations in this dynamic field.

6.3 Limitations

While the proposed method may not be accurate in capturing active links between participating peers, it captures with high probability the majority of all possible links available to all reachable nodes. We argue that this is both a weakness and a strength. On the one hand, with an inaccurate network topology, one cannot draw safe conclusions about the information dynamics of the network and how specific aspects may be improved. It also limits our ability to extract precise network metrics to correctly categorize and characterize networks. On the other hand, having the complete set of all available connections that can be realized in the network offers a more robust foundation for studying the network's resilience. This is because if a graph of potential links can be partitioned, the actual network with fewer links would also be disrupted (cf. Section 4.2). Additionally, the proposed method not only avoids the difficulties involved in achieving accurate topology recovery but also makes the results more resistant to measurement errors. Nevertheless, in our methodology validation section, we outline how the calculated metrics can serve as a bound to the actual network characteristics (see Section 4.7.2). Thus, the calculated metrics can provide useful insight into the network properties.

Note: The latest version of Bitcoin Core includes changes that affect address advertisements. This version uses cached responses to `getadrr` requests; for a period of 24 hours, all `getadrr` requests from any peer are served by a fixed `addr` reply containing up to 1000 peers. This further impedes the collection of `ADDRMAN` contents.[2]. If the adoption of the latest client version prevails, the scraping of `peers.dat` files would take several days to complete. In that case, we could speed up the crawling process by using multiple vantage points in different networks.[3]. Moreover, a motivated adversary can still infer most critical nodes based on their temporal characteristics, as we have shown in Section 5.5.1.

Likewise, the connectivity among peers categorized as unreachable remains uncertain and shrouded in obscurity. This prompts the question of whether these peers may indeed have connections among themselves. If such a situation were to exist, particularly given the substantial count of unreachable peers compared to reachable, it could potentially result in the presence of a significant connected component within the unreachable region, thus eluding detection through the employed methodology.

The work of Wang and Pustogarov [101] which suggests that a significant proportion of unreachable peers are associated with wallet applications that do not accept incoming connections, partially addresses this limitation. This finding aligns with the observation that some nodes in blockchain networks, such as lightweight wallets, may intentionally limit their connections to enhance privacy and reduce network load and also aligns with our observations.

Given the nature of blockchain P2P networks and the intentional topology hiding mechanisms, it is challenging to capture the complete connectivity accurately. However, the methodology

[2]Bitcoin Core. *Cache responses to GETADDR to prevent topology leaks.* 2021. URL: `https://github.com/bitcoin/bitcoin/pull/18991`.

[3]Bitcoin Core Developers. *PR18991 implementation.* 2020. URL: `https://github.com/bitcoin/bitcoin/blob/master/src/net.h#L1075`.

applied in this study, which prioritizes potential connections and maintains uniform treatment of all links, yields valuable insights into the structural features and possible vulnerabilities of these networks. It offers a foundation for understanding common patterns and weaknesses in blockchain P2P networks that can be valuable in enhancing their resilience and security.

In summary, while there may be undetected connections among unreachable peers, the methodology used in this research aims to provide a structural assessment of blockchain overlay networks based on available information. Related work [101] suggests that connections within the unreachable region of the Bitcoin protocol are rare, since the significant majority of unreachable peers are wallet clients. Taking into account the similar nature of the networks chosen for this study (cryptocurrencies), we assume that this pattern likely extends to the other networks included in our analysis.

6.4 Future Research

The characteristics of blockchain overlays share similarities with the core-periphery network structure, as outlined by Borgatti and Everett in [13]. The intuitive conception of the core-periphery structure entails a dense, cohesive core and a sparse, unconnected periphery. In blockchain overlays, we observe a highly connected and reachable core with unreachable peers in the periphery. This structure can be explained by the pigeonhole principle: a very large number of unreachable peers seek to connect to a smaller number of reachable nodes. Therefore, it is inevitable that multiple unreachable peers establish connections with the same reachable peer in the connected core of the network. Exploring the potential presence of the core-periphery structure within such overlays and assessing the feasibility of using it as a more precise model for these networks is an intriguing avenue for further investigation.

The results obtained from our work suggest that there exists a relationship between the presence of unreachable nodes and the resilience properties of an overlay. Further exploration of this aspect could provide valuable insights and inform future developments.

Another interesting direction for future work would be to use analytical methods to explain the findings of the resilience study and create a mathematical model based on the observed network topology. This could improve our understanding of the resilience of blockchain networks, guide more secure protocols, and help predict and prevent vulnerabilities or attacks.

Future research should focus on developing defensive strategies to enhance the resilience and security of blockchain overlay networks. While this study identified vulnerabilities, its main objective was diagnostic rather than providing comprehensive mitigation strategies. However, identifying vulnerabilities is often a critical first step in improving network resilience. By understanding the weaknesses, researchers and practitioners can develop targeted and effective mitigation strategies. A future line of research and a more extensive exploration of possible solutions could definitely build upon the presented work, to mitigate potential network threats.

6.5 Concluding remarks

The conclusion of this thesis could be summarized in the following sentence: "*If you are building a blockchain based application or innovative solution, you better think about the network implementation twice*".

We have explored the resilience characteristics of seven distinct blockchain networks. Our methodology, simple yet effective, enables the simultaneous measurement of these networks in a practical and efficient manner. By treating all network elements uniformly and avoiding distinctions based on their role or position in the network, we have gained a comprehensive view of blockchain overlay networks across different systems. This approach, characterized by its inclusivity, simplicity, and flexibility, has broadened our analytical scope and simplified our data processing.

The findings revealed vulnerabilities to targeted attacks and revealed hidden interconnections among networks. These interdependencies could potentially be exploited by a powerful adversary to disrupt multiple networks simultaneously. The most notable outcome of our study is the surprising discovery that removing a small number of highly connected nodes can significantly partition major cryptocurrency networks, even considering the abundance of network connections available to peers. The robustness of our results comes from considering all potential links within the network, not only active connections. Furthermore, despite the shared protocols governing most blockchain overlays, their graph properties exhibit variations, possibly due to diverse participation and temporal user behavior.

This thesis offers critical insights into the nature and characteristics of blockchain overlay networks, leading to implications that span the domains of network analysis, blockchain technology, and security. As the blockchain landscape continues to evolve, these implications serve as a valuable foundation for further research, innovation, and the development of strategies to promote more secure and resilient blockchain systems.

Bibliography

[1] Réka Albert and Albert-László Barabási. “Statistical mechanics of complex networks”. In: *Reviews of modern physics* 74.1 (2002).

[2] Mohammed J. F. Alenazi and James P. G. Sterbenz. “Comprehensive comparison and accuracy of graph metrics in predicting network resilience”. In: *DRCN*. IEEE, 2015.

[3] Mohammed J. F. Alenazi and James P. G. Sterbenz. “Evaluation and comparison of several graph robustness metrics to improve network resilience”. In: *RNDM@WMNC*. IEEE, 2015.

[4] Jeff Alstott, Ed Bullmore, and Dietmar Plenz. “powerlaw: A Python Package for Analysis of Heavy-Tailed Distributions”. In: *PLoS ONE* 9.1 (Jan. 2014). Ed. by FabioEditor Rapallo. DOI: 10.1371/journal.pone.0085777. URL: http://dx.doi.org/10.1371/journal.pone.0085777.

[5] Yusuke Aoki et al. “SimBlock: A Blockchain Network Simulator”. In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (2019), pp. 325–329. URL: https://api.semanticscholar.org/CorpusID:59316461.

[6] Maria Apostolaki, Aviv Zohar, and Laurent Vanbever. “Hijacking Bitcoin: Routing Attacks on Cryptocurrencies”. In: *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2017.

[7] Asaph Azaria et al. “MedRec: Using Blockchain for Medical Data Access and Permission Management”. In: *OBD*. IEEE Computer Society, 2016.

[8] Moshe Babaioff et al. “On bitcoin and red balloons”. In: *EC*. ACM, 2012.

[9] Albert-László Barabási et al. *Network science*. Cambridge University Press, 2016. URL: https://http://www.networksciencebook.com/.

[10] Sami Ben Mariem, Pedro Casas, and Benoit Donnet. "Vivisecting Blockchain P2P Networks: Unveiling the Bitcoin IP Network". In: *ACM CoNEXT Student Workshop*. 2018.

[11] Joseph Bonneau et al. "SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies". In: *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2015.

[12] Lashon B Booker. *The effects of observation errors on the attack vulnerability of complex networks*. Tech. rep. MITRE CORP MCLEAN VA, 2012.

[13] Stephen P. Borgatti and Martin G. Everett. "Models of core/periphery structures". In: *Soc. Networks* 21 (2000).

[14] Andrei Z. Broder et al. "Graph structure in the Web". In: *Comput. Networks* 33.1-6 (2000).

[15] Simone Casale-Brunet et al. "Networks of Ethereum Non-Fungible Tokens: A graph-based analysis of the ERC-721 ecosystem". In: *Blockchain*. IEEE, 2021.

[16] Panagiotis Chatzigiannis et al. "Diversification across mining pools: optimal mining strategies under PoW". In: *J. Cybersecur.* 8.1 (2022).

[17] Wubing Chen et al. "A Survey of Blockchain Applications in Different Domains". In: *ICBTA*. ACM, 2018.

[18] Konstantinos Christidis and Michael Devetsikiotis. "Blockchains and Smart Contracts for the Internet of Things". In: *IEEE Access* 4 (2016).

[19] Aaron Clauset, Cosma Rohilla Shalizi, and Mark E. J. Newman. "Power-Law Distributions in Empirical Data". In: *SIAM Rev.* 51.4 (2009).

[20] Kyle Croman et al. "On Scaling Decentralized Blockchains - (A Position Paper)". In: *Financial Cryptography Workshops*. Vol. 9604. Lecture Notes in Computer Science. Springer, 2016.

[21] Erik Daniel, Elias Rohrer, and Florian Tschorsch. "Map-Z: Exposing the Zcash Network in Times of Transition". In: *LCN*. IEEE, 2019.

[22] Christian Decker and Roger Wattenhofer. "Information propagation in the Bitcoin network". In: *P2P*. IEEE, 2013.

[23] Sergi Delgado-Segura et al. "Cryptocurrency Networks: A New P2P Paradigm". In: *Mob. Inf. Syst.* 2018 (2018).

[24] Sergi Delgado-Segura et al. “TxProbe: Discovering Bitcoin’s Network Topology Using Orphan Transactions”. In: *Financial Cryptography*. Vol. 11598. Lecture Notes in Computer Science. Springer, 2019.

[25] Varun Deshpande, Hakim Badis, and Laurent George. “BTCmap: Mapping Bitcoin Peer-to-Peer Network Topology”. In: *PEMWN*. IEEE, 2018.

[26] Joan Antoni Donet Donet, Cristina Pérez-Solà, and Jordi Herrera-Joancomarti. “The Bitcoin P2P Network”. In: *Financial Cryptography Workshops*. Vol. 8438. Lecture Notes in Computer Science. Springer, 2014.

[27] Maya Dotan et al. “SOK: cryptocurrency networking context, state-of-the-art, challenges”. In: *ARES*. ACM, 2020.

[28] Jean-Philippe Eisenbarth, Thibault Cholez, and Olivier Perrin. “A Comprehensive Study of the Bitcoin P2P Network”. In: *BRAINS*. IEEE, 2021.

[29] Ittay Eyal and Emin Gün Sirer. “Majority is not enough: bitcoin mining is vulnerable”. In: *Commun. ACM* 61.7 (2018).

[30] Julius Fechner, Balakrishnan Chandrasekaran, and Marc X. Makkes. “Calibrating the performance and security of blockchains via information propagation delays: revisiting an old approach with a new perspective”. In: *SAC*. ACM, 2022.

[31] Sebastian Feld, Mirco Schönfeld, and Martin Werner. “Analyzing the Deployment of Bitcoin’s P2P Network under an AS-level Perspective”. In: *ANT/SEIT*. Vol. 32. Procedia Computer Science. Elsevier, 2014.

[32] Miroslav Fiedler. “Algebraic connectivity of graphs”. In: *Czechoslovak mathematical journal* 23.2 (1973).

[33] Bryan Ford and Rainer Böhme. “Rationality is Self-Defeating in Permissionless Systems”. In: *CoRR* abs/1910.08820 (2019). arXiv: `1910.08820`. URL: `http://arxiv.org/abs/1910.08820`.

[34] Federico Franzoni and Vanesa Daza. “SoK: Network-Level Attacks on the Bitcoin P2P Network”. In: *IEEE Access* 10 (2022). DOI: `10.1109/ACCESS.2022.3204387`.

[35] Federico Franzoni, Xavier Salleras, and Vanesa Daza. “AToM: Active topology monitoring for the bitcoin peer-to-peer network”. In: *Peer-to-Peer Netw. Appl.* 15.1 (2022).

[36] Linton C Freeman. “A set of measures of centrality based on betweenness”. In: *Sociometry* (1977).

[37] Scott Freitas et al. “Graph Vulnerability and Robustness: A Survey”. In: *IEEE Trans. Knowl. Data Eng.* 35.6 (2023).

[38] Juan A. Garay and Aggelos Kiayias. “SoK: A Consensus Taxonomy in the Blockchain Era”. In: *CT-RSA*. Vol. 12006. Lecture Notes in Computer Science. Springer, 2020.

[39] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. “The Bitcoin Backbone Protocol: Analysis and Applications”. In: *EUROCRYPT (2)*. Vol. 9057. Lecture Notes in Computer Science. Springer, 2015.

[40] Diego Garlaschelli and Maria I. Loffredo. “Patterns of Link Reciprocity in Directed Networks”. In: *Physical Review Letters* 93.26 (Dec. 2004). ISSN: 1079-7114. DOI: 10.1103/physrevlett.93.268701. URL: http://dx.doi.org/10.1103/PhysRevLett.93.268701.

[41] Adem Efe Gencer et al. “Decentralization in Bitcoin and Ethereum Networks”. In: *Financial Cryptography*. Vol. 10957. Lecture Notes in Computer Science. Springer, 2018.

[42] Arthur Gervais et al. “Is Bitcoin a Decentralized Currency?” In: *IEEE Secur. Priv.* 12.3 (2014).

[43] Arthur Gervais et al. “On the Security and Performance of Proof of Work Blockchains”. In: *CCS*. ACM, 2016.

[44] Arthur Gervais et al. “Tampering with the Delivery of Blocks and Transactions in Bitcoin”. In: *CCS*. ACM, 2015.

[45] Sarada Prasad Gochhayat et al. “Measuring Decentrality in Blockchain Based Systems”. In: *IEEE Access* 8 (2020).

[46] Matthias Grundmann, Hedwig Amberg, and Hannes Hartenstein. “On the Estimation of the Number of Unreachable Peers in the Bitcoin P2P Network by Observation of Peer Announcements”. In: *CoRR* abs/2102.12774 (2021).

[47] Matthias Grundmann, Max Baumstark, and Hannes Hartenstein. “On the Peer Degree Distribution of the Bitcoin P2P Network”. In: *ICBC*. IEEE, 2022.

[48] Matthias Grundmann, Till Neudecker, and Hannes Hartenstein. “Exploiting Transaction Accumulation and Double Spends for Topology Inference in Bitcoin”. In: *Financial Cryptography Workshops*. Vol. 10958. Lecture Notes in Computer Science. Springer, 2018.

[49] Matthias Grundmann et al. “Short Paper: What Peer Announcements Tell Us About the Size of the Bitcoin P2P Network”. In: *Financial Cryptography*. Vol. 13411. Lecture Notes in Computer Science. Springer, 2022.

[50] Lewis Gudgeon et al. “SoK: Layer-Two Blockchain Protocols”. In: *Financial Cryptography*. Vol. 12059. Lecture Notes in Computer Science. Springer, 2020.

[51] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. “Exploring Network Structure, Dynamics, and Function using NetworkX”. In: *Proceedings of the 7th Python in Science Conference*. Ed. by Gaël Varoquaux, Travis Vaught, and Jarrod Millman. Pasadena, CA USA, 2008.

[52] Weifeng Hao et al. “BlockP2P: Enabling Fast Blockchain Broadcast with Scalable Peer-to-Peer Network Topology”. In: *GPC*. Vol. 11484. Lecture Notes in Computer Science. Springer, 2019.

[53] Frank Harary. “The Maximum Connectivity of a Graph”. In: *Proceedings of the National Academy of Sciences* 48.7 (1962). ISSN: 00278424. (Visited on 06/10/2022).

[54] Ethan Heilman et al. “Eclipse Attacks on Bitcoin’s Peer-to-Peer Network”. In: *USENIX Security Symposium*. USENIX Association, 2015.

[55] Sebastian A. Henningsen et al. “Eclipsing Ethereum Peers with False Friends”. In: *EuroS&P Workshops*. IEEE, 2019.

[56] Muhammad Anas Imtiaz et al. “Churn in the Bitcoin Network: Characterization and Impact”. In: *IEEE ICBC*. IEEE, 2019.

[57] Swami Iyer et al. “Attack robustness and centrality of complex networks”. In: *PloS one* 8.4 (2013).

[58] Marco Alberto Javarone and Craig Steven Wright. “From Bitcoin to Bitcoin Cash: a network analysis”. In: *CRYBLOCK@MobiSys*. ACM, 2018.

[59] Dimitris Karakostas, Aggelos Kiayias, and Christina Ovezik. “SoK: A Stratified Approach to Blockchain Decentralization”. In: *CoRR* abs/2211.01291 (2022). DOI: 10.48550/arXiv.2211.01291. arXiv: 2211.01291. URL: https://doi.org/10.48550/arXiv.2211.01291.

[60] Arijit Khan. “Graph Analysis of the Ethereum Blockchain Data: A Survey of Datasets, Methods, and Future Work”. In: *Blockchain*. IEEE, 2022.

[61] Lucianna Kiffer et al. “Under the Hood of the Ethereum Gossip Protocol”. In: *Financial Cryptography (2)*. Vol. 12675. Lecture Notes in Computer Science. Springer, 2021.

[62] Seoung Kyun Kim et al. “Measuring Ethereum Network Peers”. In: *Internet Measurement Conference*. ACM, 2018.

[63] Jon Kleinberg and Steve Lawrence. “The structure of the Web”. In: *Science* 294.5548 (2001).

[64] S. Kokoska and D. Zwillinger. *CRC Standard Probability and Statistics Tables and Formulae, Student Edition*. CRC Press, 1999.

[65] Seungjin Lee and Hyoungshick Kim. “On the robustness of Lightning Network in Bitcoin”. In: *Pervasive Mob. Comput.* 61 (2020).

[66] Jure Leskovec and Rok Sosic. “SNAP: A General-Purpose Network Analysis and Graph-Mining Library”. In: *ACM Trans. Intell. Syst. Technol.* 8.1 (2016).

[67] Kai Li et al. “TopoShot: uncovering Ethereum’s network topology leveraging replacement transactions”. In: *Internet Measurement Conference*. ACM, 2021.

[68] Yitao Li et al. “Dissecting Ethereum Blockchain Analytics: What We Learn from Topology and Geometry of the Ethereum Graph?” In: *SDM*. SIAM, 2020.

[69] Damien Magoni. “Tearing down the Internet”. In: *IEEE J. Sel. Areas Commun.* 21.6 (2003).

[70] Priya Mahadevan et al. “The internet AS-level topology: three data sources and one definitive metric”. In: *Comput. Commun. Rev.* 36.1 (2006).

[71] Yuval Marcus, Ethan Heilman, and Sharon Goldberg. “Low-Resource Eclipse Attacks on Ethereum’s Peer-to-Peer Network”. In: *IACR Cryptol. ePrint Arch.* (2018).

[72] Petar Maymounkov and David Mazières. “Kademlia: A Peer-to-Peer Information System Based on the XOR Metric”. In: *IPTPS*. Vol. 2429. Lecture Notes in Computer Science. Springer, 2002.

[73] Andrew Miller et al. *Discovering bitcoin’s network topology and influential nodes*. Tech. rep. University of Maryland, 2015.

[74] Alan Mislove et al. “Measurement and analysis of online social networks”. In: *Internet Measurement Conference*. ACM, 2007.

[75] Arvind Narayanan et al. *Bitcoin and Cryptocurrency Technologies - A Comprehensive Introduction*. Princeton University Press, 2016.

[76] Christopher Natoli et al. “Deconstructing Blockchains: A Comprehensive Survey on Consensus, Membership and Structure”. In: *CoRR* abs/1908.08316 (2019).

[77] Kartik Nayak et al. “Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack”. In: *EuroS&P*. IEEE, 2016.

[78] Till Neudecker. *Characterization of the Bitcoin Peer-to-Peer Network (2015-2018)*. Tech. rep. 1. Karlsruher Institut für Technologie (KIT), 2019. 29 pp. DOI: `10.5445/IR/1000091933`.

[79] Till Neudecker, Philipp Andelfinger, and Hannes Hartenstein. “A simulation model for analysis of attacks on the Bitcoin peer-to-peer network”. In: *IM*. IEEE, 2015.

[80] Till Neudecker, Philipp Andelfinger, and Hannes Hartenstein. “Timing Analysis for Inferring the Topology of the Bitcoin Peer-to-Peer Network”. In: *UIC/ATC/ScalCom/CBDCom/IoP/SmartWorl* IEEE Computer Society, 2016. DOI: `10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0070`.

[81] M. E. J. Newman, Stephanie Forrest, and Justin Balthrop. “Email networks and the spread of computer viruses”. In: *Phys. Rev. E* 66 (3 Sept. 2002). DOI: `10.1103/PhysRevE.66.035101`. URL: `https://link.aps.org/doi/10.1103/PhysRevE.66.035101`.

[82] Mark E. J. Newman. “Mixing patterns in networks.” In: *Physical review E* 67.2 (2003).

[83] A. Pinar Ozisik et al. “Graphene: efficient interactive set reconciliation applied to blockchain propagation”. In: *SIGCOMM*. ACM, 2019.

[84] Sehyun Park et al. “Nodes in the Bitcoin Network: Comparative Measurement Study and Survey”. In: *IEEE Access* 7 (2019).

[85] Colin Percival and Simon Josefsson. “The scrypt Password-Based Key Derivation Function”. In: *RFC* 7914 (2016). DOI: `10.17487/RFC7914`. URL: `https://doi.org/10.17487/RFC7914`.

[86] Diego F. Rueda, Eusebi Calle, and Josè-Luis Marzo. “Robustness Comparison of 15 Real Telecommunication Networks: Structural and Centrality Measurements”. In: *J. Netw. Syst. Manag.* 25.2 (2017).

[87] Muhammad Saad, Songqing Chen, and David Mohaisen. “SyncAttack: Double-spending in Bitcoin Without Mining Power”. In: *CCS*. ACM, 2021.

[88] Muhammad Saad et al. “Exploring the Attack Surface of Blockchain: A Comprehensive Survey”. In: *IEEE Commun. Surv. Tutorials* 22.3 (2020).

[89] Muhammad Saad et al. “Partitioning Attacks on Bitcoin: Colliding Space, Time, and Logic”. In: *ICDCS*. IEEE, 2019.

[90] Ashish Rajendra Sai et al. “Taxonomy of Centralization in Public Blockchain Systems: A Systematic Literature Review”. In: *Inf. Process. Manag.* 58 (2020).

[91] Christian H Sanabria-Montaña and Rodrigo Huerta-Quintanilla. “Generalization of Clustering Coefficient on Lattice Networks Applied to Criminal Networks”. In: *International Journal of Computer and Information Engineering* 11.7 (2017).

[92] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. “Optimal Selfish Mining Strategies in Bitcoin”. In: *Financial Cryptography*. Vol. 9603. Lecture Notes in Computer Science. Springer, 2016.

[93] Georgos Siganos, Sudhir Leslie Tauro, and Michalis Faloutsos. “Jellyfish: A conceptual model for the as Internet topology”. In: *J. Commun. Networks* 8.3 (2006).

[94] Daniel Stutzbach and Reza Rejaie. “Understanding churn in peer-to-peer networks”. In: *Internet Measurement Conference*. ACM, 2006.

[95] Qawi K. Telesford et al. “The Ubiquity of Small-World Networks”. In: *Brain Connect.* 1.5 (2011).

[96] Natkamon Tovanich et al. “An Empirical Analysis of Pool Hopping Behavior in the Bitcoin Blockchain”. In: *IEEE ICBC*. IEEE, 2021.

[97] Muoi Tran et al. “A Stealthier Partitioning Attack against Bitcoin Peer-to-Peer Network”. In: *IEEE Symposium on Security and Privacy*. IEEE, 2020.

[98] Florian Tschorsch and Björn Scheuermann. “Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies”. In: *IEEE Commun. Surv. Tutorials* 18.3 (2016).

[99] Pauli Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020).

[100] Dan J. Wang et al. “Measurement error in network data: A re-classification”. In: *Soc. Networks* 34 (2012).

[101] Liang Wang and Ivan Pustogarov. “Towards Better Understanding of Bitcoin Unreachable Peers”. In: *CoRR* abs/1709.06837 (2017).

[102] Taotao Wang et al. “Ethna: Analyzing the Underlying Peer-to-Peer Network of Ethereum Blockchain”. In: *IEEE Trans. Netw. Sci. Eng.* 8.3 (2021).

[103] Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications*. Structural Analysis in the Social Sciences. Cambridge University Press, 1994. DOI: 10.1017/CBO9780511815478.

[104] Hassler Whitney. “Congruent Graphs and the Connectivity of Graphs”. In: *American Journal of Mathematics* 54.1 (1932). ISSN: 00029327, 10806377. URL: `http://www.jstor.org/stable/2371086` (visited on 06/10/2022).

[105] Jun Wu et al. “ATTACK VULNERABILITY OF COMPLEX NETWORKS BASED ON LOCAL INFORMATION”. In: *Modern Physics Letters B* 21 (2007).

[106] Yongxiang Xia, Jin Fan, and David Hill. “Cascading failure in Watts–Strogatz small-world networks”. In: *Physica A: Statistical Mechanics and its Applications* 389.6 (2010). ISSN: 0378-4371. DOI: `https://doi.org/10.1016/j.physa.2009.11.037`.

[107] Jiale Yang et al. “Detectable, Traceable, and Manageable Blockchain Technologies BHE: An Attack Scheme against Bitcoin P2P Network”. In: *WCMC* (2022).

[108] Timothy Tzen Vun Yap et al. “Exploratory graph analysis of the network data of the Ethereum blockchain”. In: *F1000Research* 10 (2021).

[109] Ian Young. “Dogecoin: A Brief Overview & Survey”. In: *SSRN* (2018). URL: `http://dx.doi.org/10.2139/ssrn.3306060`.

[110] Chen Zhao and Yong Guan. “A Graph-Based Investigation of Bitcoin Transactions”. In: *IFIP Int. Conf. Digital Forensics*. Vol. 462. IFIP Advances in Information and Communication Technology. Springer, 2015.

[111] Guy Zyskind, Oz Nathan, and Alex Pentland. “Decentralizing Privacy: Using Blockchain to Protect Personal Data”. In: *IEEE Symposium on Security and Privacy Workshops*. IEEE Computer Society, 2015.

Online Sources

[112] Atish Kulkarni and M. Leeuwen. *Master Thesis: Graph pattern mining for blockchain networks*. 2021. URL: `https://api.semanticscholar.org/CorpusID:240353620`.

[113] BitcoinCash. *Bitcoin Cash*. 2021. URL: `https://www.bitcoincash.org`.

[114] bitcoinj. *bitcoinj Public Repository*. 2017. URL: `https://github.com/bitcoinj/bitcoinj`.

[115] bitnodes.io. *Global Bitcoin Nodes Distribution*. 2020. URL: `https://bitnodes.io` (visited on 09/18/2020).

[116] *Blockchain Technology Market Size, Share & Growth [2030]*. 2023. URL: https://www.fortunebusinessinsights.com/industry-reports/blockchain-market-100072.

[117] Vitalik Buterin. *Ethereum: A next-generation smart contract and decentralized application platform*. 2014. URL: https://github.com/ethereum/wiki/wiki/White-Paper.

[118] GENERAL BYTES. *Bitrafael Public Repository*. 2017. URL: https://github.com/GENERALBYTESCOM/bitrafael_public.

[119] Bitcore Client. *Bitcore Client Package*. 2017. URL: https://www.npmjs.com/package/bitcore-client/v/8.19.0.

[120] CoinMarketCap. *CoinMarketCap*. 2021. URL: https://coinmarketcap.com.

[121] Bitcoin Core. *0.20.1 Release Notes*. 2021.

[122] Bitcoin Core. *Bitcoin Core, addrman implementation*. 2021. URL: https://github.com/bitcoin/bitcoin/blob/d0d256536cdfb1443067fb7cc0a19d647f
src/addrman.cpp#L41.

[123] Bitcoin Core. *Cache responses to GETADDR to prevent topology leaks*. 2021. URL: https://github.com/bitcoin/bitcoin/pull/18991.

[124] Bitcoin Core Developers. *Bitcoin core 0.10.1 release notes*. https://github.com/bitcoin/bitcoin/blob/v0.10.1/doc/release-notes.md. 2015. URL: https://github.%20com/bitcoin/bitcoin/blob/v0.10.1/doc/release-notes.md (visited on 01/17/2021).

[125] Bitcoin Core Developers. *Bitcoin Core integration/staging tree*. 2021. URL: https://github.com/bitcoin/bitcoin.

[126] Bitcoin Core Developers. *Bitcoin P2P Network*. 2021. URL: https://developer.bitcoin.org/devguide/p2p_network.html.

[127] Bitcoin Core Developers. *Bitcoin Stochastic address manager*. 2020. URL: https://github.com/bitcoin/bitcoin/blob/be3af4f31089726267ce2dbdd6c9c153bb
src/addrman.h#L108.

[128] Bitcoin Core Developers. *Do not add random inbound peers to addrman.PR8594*. 2020. URL: https://github.com/bitcoin/bitcoin/pull/8594#issue-173338388.

[129] Bitcoin Core Developers. *Ignore GETADDR msg from inbound connections*. 2020. URL: https://github.com/bitcoin/bitcoin/blob/37e9f07996d3a7504ea54180d18 src/net_processing.cpp#L3567.

[130] Bitcoin Core Developers. *PR18991 implementation*. 2020. URL: https://github.com/bitcoin/bitcoin/blob/master/src/net.h#L1075.

[131] Ethereum. *Ethereum peer-to-peer networking specifications*. 2014. URL: https://github.com/ethereum/devp2p.

[132] Go Ethereum. *Getting Started with Geth*. 2022. URL: https://geth.ethereum.org/docs/getting-started.

[133] Parity Ethereum. *Parity Ethereum*. 2022. URL: https://github.com/openethereum/parity-ethereum.

[134] Daniel Diaz Evan Duffield. *Dash: A Payments-Focused Cryptocurrency*. 2018. URL: https://github.com/dashpay/dash/wiki/Whitepaper.

[135] Bitcoin Fees. *bitcoinfees.earn.com*. 2021. URL: https://bitcoinfees.earn.com.

[136] Bitcoin Forum. *Interesting : Bitcoin ABC try to fork ... the Bitcoin Blockchain.* 2017. URL: https://bitcointalk.org/index.php?topic=2063734.0.

[137] Bitcoin Forum. *UASF nodes wrongly reporting IP*. 2017. URL: https://bitcointalk.org/index.php?topic=1954151.0.

[138] Daira Hopwood et al. *Zcash Protocol Specification*. 2020. URL: https://coinpare.io/whitepaper/zcash.pdf.

[139] Thaddeus Dryja Joseph Poon. *The Bitcoin Lightning Network:Scalable Off-Chain Instant Payments*. 2016. URL: https://lightning.network/lightning-network-paper.pdf.

[140] Luke Dash Jr. *Bitcoin Historical Node Count*. 2022. URL: https://luke.dashjr.org/programs/bitcoin/files/charts/historical.html.

[141] Qrator Labs. *Q1 2023 DDoS Attacks and BGP Incidents*. 2023. URL: https://blog.qrator.net/en/q1-2023-ddos-attacks-and-bgp-incidents_171/.

[142] Litecoin. *Litecoin*. 2021. URL: https://litecoin.org.

[143] Billy Markus and Jackson Palmer. *Dogecoin*. 2013.

[144] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. 2009. URL: http://bitcoin.org/bitcoin.pdf.

[145] Raiden Network. *What is the Raiden Network?* 2017. URL: https://raiden.network/101.html.

[146] Jonas Nick. *Guessing Bitcoin's P2P Connections*. 2015. URL: https://jonasnick.github.io/blog/2015/03/06/guessing-bitcoins-p2p-connections/.

[147] Aristodemos Paphitis, Nicolas Kourtellis, and Michael Sirivianos. *A First Look into the Structural Properties of Blockchain P2P Overlays*. 2023. DOI: 10.6084/m9.figshare.23522919. URL: 10.6084/m9.figshare.23522919.

[148] Kevin Roose. *Is There a Cryptocurrency Bubble? Just Ask Doge*. 2017. URL: https://www.nytimes.com/2017/09/15/business/cryptocurrency-bubble-doge.html.

[149] Bruce Schneier. *Blockchain and Trust*. 2019. URL: https://www.schneier.com/blog/archives/2019/02/blockchain_and_.html.

[150] Yaron Singer. *CS 134 / Econ 1034: Networks. Class Notes*. 2017. URL: https://web.archive.org/web/20210415052757/http://www.hcs.harvard.edu/~cs134-spring2017/wp-content/uploads/2017/01/section2.pdf.

[151] SlushPool. *STRATUM V1 Docs*. 2022. URL: https://braiins.com/stratum-v1/docs.

[152] Jimmy Song. *Bitcoin Cash: What You Need to Know*. 2017. URL: https://jimmysong.medium.com/bitcoin-cash-what-you-need-to-know-c25df28995cf.

[153] trinity.ethereum.org. *The Trinity Ethereum Client*. 2021. URL: https://trinity.ethereum.org.

[154] Pieter Wuille. *Countermeasures against eclipse attacks*. 2015. URL: https://github.com/bitcoin/bitcoin/pull/5941.

[155] Pieter Wuille. *Replace global trickle node with random delays*. 2015. URL: https://github.com/bitcoin/bitcoin/pull/7125.

[156] Addy Yeow. *Bitnodes Network Crawler*. 2021. URL: https://github.com/ayeowch/bitnodes.