Peer-to-Peer Security

Mohamed Ibrahim Atallah
Prof. Dr. Ayman Abdel-Hamid

College of Computing and Information Technology
Arab Academy for Science, Technology and Maritime Transport

15th September 2022

Abstract

Developments to peer-to-peer systems have been revolutionary in many ways, Napster, Gnutella and BitTorrent were some of the first P2P application that gave the world a glimpse of their true potential, later with the inception of distributed ledger by Bitcoin, that brought a way to solve the double spending problem. P2P also brought a way to combat censorship and provide anonymization services to those who need it via Tor. This report attempts to survey the security implications of P2P.

1 Introduction

Peer-to-Peer (P2P) systems present a unique set of security challenges, while some of these challenges are new and unique to P2P systems, most of them are in fact the same security challenges we're already aware of but amplified due to the inherent complexity of P2P. In the traditional client-server systems we always had a central authority to trust. To this end, the target of any potential attack is the server and any disputes in the system are resolved by the server as well, and so we had a single point of both trust and failure. On the other hand in P2P systems, peers have mostly similar roles they can be clients or servers an early example of this is BitTorrent [5], where peers can be clients (i.e., downloaders, *leechers*), servers (i.e., uploaders, *seeders*), or simultaneously uploading pieces of files they have and downloading pieces of files they don't have. Effectively being both the client and the server at the same time. We can quickly see the attractiveness of this model, the burden of sharing files is no longer carried out by some central resource but shared among peers to achieve a common goal.

The emergence of blockchain and cryptocurrencies initiated by Bitcoin [8], sparked major interest in P2P systems, with major companies touting what is so called "The Metaverse" to be the future, governed by blockchain technologies. However, before that dystopian nightmare becomes a reality we need to understand its security implications.

Contributions of this report are the following: we explore these security challenges in the domain of P2P systems, major innovations and how it addresses these challenges we also provide an in-depth review of recent works in P2P security regarding the following topics:

- Peer Discovery
- Message Propagation
- Consensus

2 Background

This section presents the conceptual framework, on which we will base our understanding of P2P systems. Let us begin with a formal definition of what is a P2P system derived from the ideas we discussed so far. A P2P system is a system in which peers have symmetric functions, where how performant the system is determined by the peers and their shared resources, rather than a central resource, and so a single word can describe such system **decentralization**. Now we can categorize P2P systems as such [15]:

- Structured: peers are organized in a deterministic topology (rings, binary trees, grids, etc.)
- Unstructured: peers maintain an ad-hoc list of neighbors

2.1 Peer Discovery

Secure peer discovery is the process of finding identifiers of other nodes (peers) in the network, while maintaining the following invariants [12]:

1. Peer discovery is unbiased (a form of active attacks)
2. Avoiding information leakage (a form of passive attacks)

2.1.1 Centralized Discovery

One solution to our discovery task is to utilize a central directory authority (including replicas) [5, 17] that keeps track of all active peers in the network, whenever a new node joins the network it registers itself with that authority, and gets a list of nodes to establish connections. However, this approach is flawed in many aspects: first, it proposes a trust bottleneck as observed in Tor [17] if a single directory authority lied, it could make clients believe for a time a distorted view of the network, second, a scaling bottleneck as directory require tracking of more peers with a high rate of node churn, third, it presents a target for attacks on the system [18, 10]

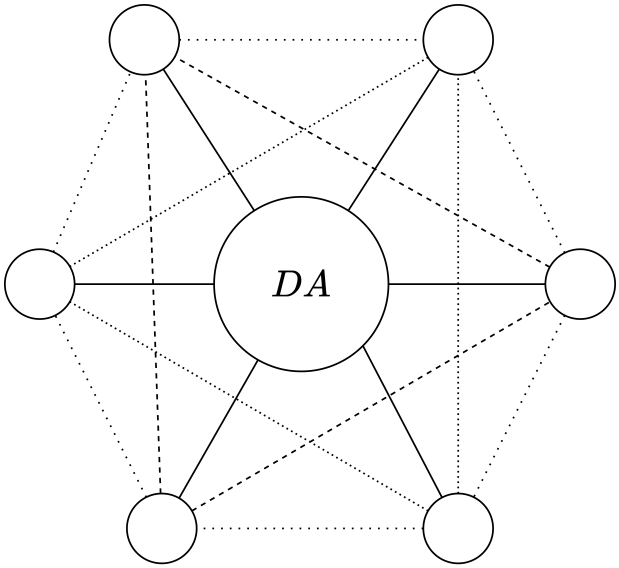

Figure 1: Centralized solution showing nodes different connections achieved by using a Directory Authority (DA)

2.1.2 Structured P2P Discovery: DHT

Distributed hash table (DHT) is purely P2P solution, that doesn't rely on a central resource for discovery. Chord [16] is one of the prominent DHT systems used for discovery, where every node gets assigned a **key**, produced from applying a cryptographic hash function to a unique node identifier (e.g., IP Address). Peers in the system should not be able to influence the key they get assigned (the result of the hash function), and other peers in the system should easily verify the

authenticity of the assigned key through applying the same hash function.

Peers then keep track of their immediate successor (first node with a higher key value) and predecessor and a finger table (FT), each finger entry consist of a starting key value and the immediate successor node of that key, FT provide convenient shortcuts to resolve key lookups. In fact, Chord key lookup has worst case complexity of $\mathcal{O}(\log n)$ [16] where n is the number of nodes, if the lookup key doesn't exist in our best guess finger node we know of from our FT, that node would forward our key lookup request to best finger node it knows of from its FT, and so on, and so forth until the lookup is resolved. However, this method of looking up a key has one major drawback is that it violates one of our node discovery invariants. Here information leakage occurs, intermediate nodes are aware of our lookup target.

A suggested solution to this problem would be request the FT of other peer instead of sending them our lookup target. This approach is still vulnerable to both passive and active attacks:

1. A malicious node may manipulate their own FT by omitting honest nodes in favour of colluding malicious nodes.

2. Information leakage may still occur if the fraction of malicious node is high enough such that malicious nodes can utilize the Chord structure to deduce information about our lookup target.

2.2 Blockchain

A widespread (and notorious [2]) application of P2P is blockchain and first of which was Bitcoin [8]. Interactions between peers on a blockchain is expressed through transactions. A transaction is message between two parties X and Y that includes the digital signature

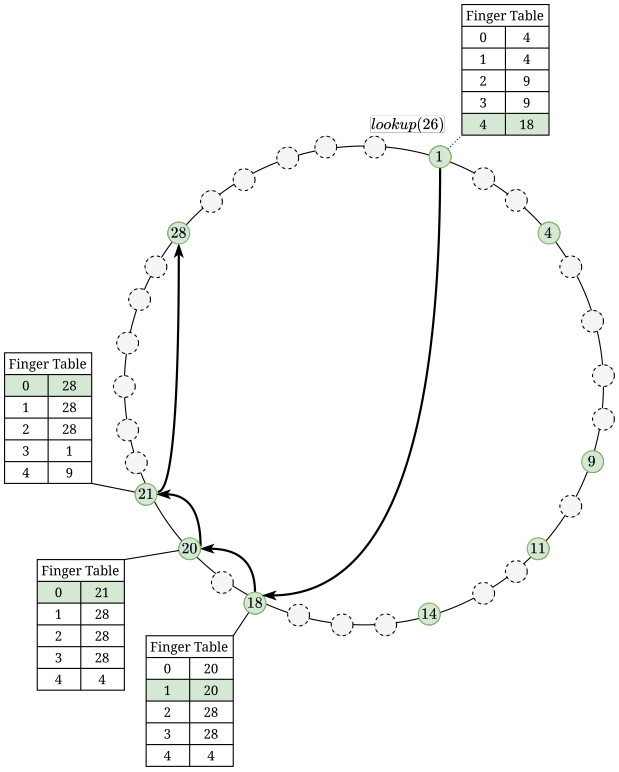

Figure 2: Chord key lookup

[14] of X and the hash of the previous transaction forming a *chain* that latest of which transfers the ownership of some digital asset to Y. The problem remains how can Y verify that X didn't double spend that digital asset in an earlier transaction? In absence of a central authority the solution is to publicly announce transactions to the network. Now only the earliest transaction is the one that counts and later transactions are deemed invalid.

We need some sort of system to come to an agreement on the truth of the chronological order of transactions, with the lack of trust in P2P environment makes this a truly challenging task as all peers can be potential attackers.

We also need a way to commit transactions to the network, associating them to a specific timestamp, in order to prevent replay attacks.

Blocks are a collection of verified transactions and acts a form of commit of transac-

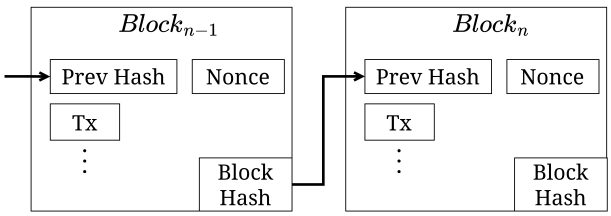

Figure 3: A simplified overview of blocks, block hash is the output of the previous hash, the nonce value and the transactions (Tx) contained within that block.

tions to the networks, that is, when a transaction is added to a block it cannot be modified or reverted, since blocks also contain a hash of the previous block forming a *chain*, and thus the name, blocks should preserve the chronological order of transactions.

2.2.1 Nakamoto Consensus

The breakthrough of Bitcoin was its consensus mechanisms, Proof-of-work (PoW) and the longest chain rule together form agreement among untrustworthy peers. Although the idea of PoW wasn't a new as it was first introduced as a mean to protect against spam e-mails[6], it was the first of its kind to be employed in a P2P environment to reach consensus.

PoW constitutes that peers work on "mining" blocks by collecting and verifying transactions into a block and then work on generating a new hash for that block by changing a nonce value to get a hash that starts with a certain number of consecutive 0 bits determined by a *mining difficulty* parameter, determined by the number of miners active on the network. Once that nonce value is found, the block is now considered *mined* and the miner then broadcasts that block to the network and peers can then easily verify that block, and move on to mining the next block.

This process requires a substantial amount of bandwidth and computing power, of course there must be an incentive to motivate the miners to keep mining (validating the network) and ensuring it operation, this incentive comes in the form of an extra transaction added to the mined block to be awarded for the miner who successfully mines that block.

The fork choice mechanism: Longest chain rule, further reinforces PoW by providing a mean for resolving chain conflicts, if a node receives two contradicting versions of the chain with an equal height from the genesis [1] that node only needs to wait for the chain that gets a newly mined block and considers that chain to be the truth of the network, that is the chain with the most computational power spent to proof its validity through PoW.

There are significant downsides of both PoW and the longest chain rule: first PoW depends on draining scarce physical resources in the form of electricity and mining hardware (GPUs, ASICs, etc.) in order to earn the mining reward. This model poses a grave environmental threat [19], second the longest chain rule doesn't provide finality on a chain just the longest chain that exists, and so an adversary can secretly work on an alternate chain, effectively removing transactions that were previously deemed valid and purpose that alternative chain when it becomes longer than the original one.

2.2.2 Proof-of-Stake

Proof-of-Stake (PoS) an alternative consensus mechanism to PoW [1], PoS replaces miners with validators, a validator gets the right to forge a new block and that validator gets rewarded the transaction fees contained in that block. In order to ensure that a validator would uphold the integrity of the

[1] The very first block published on the chain

network by not approving fraudulent transactions, validators must first place stake in the network in the form of cryptocurrency that is higher than the transaction fees, if a validator approves a fraudulent transaction they get penalized by getting part of their stake revoked. The idea here is to have higher stake than the transaction fees so if a validator misbehaves they'd lose more than they would gain from attempting to manipulate the network. Validators get chosen to forge new blocks based on the amount of stake they've placed into the network, the higher the stake the more likely it is a peer will become a validator of the next block.

2.2.3 GHOST

Greedy Heaviest-Observed Sub-Tree (GHOST) is an alternative fork choice algorithm [13], similar to Bitcoin's longest chain rule, with one major difference is that it doesn't rely on the longest chain but rather choosing the chain associated with the most weight, rather than just taking the longest one.

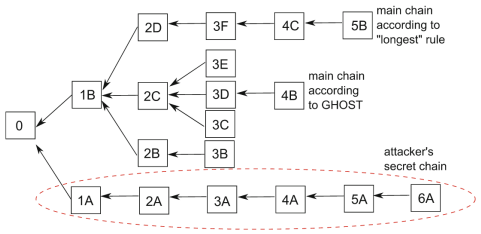

Figure 4: An attacker's chain is able to switch the longest chain, but not the chain selected by GHOST [13]

2.2.4 Casper FFG

Casper the Friendly Finality Gadget (FFG) provides "finality" utility to blockchains. By finality, we mean that the peers can come to consensus on blocks that are finalized and would never change, and these blocks serve as **checkpoints** to the correct canonical chain, prior to that the only checkpoint a blockchain had is its genesis block now it. Without checkpoints an attack can secretly fork a parallel chain to the canonical chain and publish it when it satisfies the fork choice algorithm (See Figure 4).

Casper FFG puts blocks in three states:

1. Proposed: The state at which a block has been proposed to the network

2. Justified: In this state a block has been agreed upon via consensus and considered a part of the chain

3. Finalized: In this state a block is changed into a checkpoint, so that it can no longer be revoked or modified.

3 Analysis

In this section we give an overview of recent works and advancements in P2P security.

3.1 GuardedGossip

GuardedGossip is a proposed protocol for node discovery in Tor [12]. GuardedGossip builds on Chord DHT and enhances its security properties and defends against both active and passive attacks. In section 2.1.2 we discussed that Chord is vulnerable to both passive attacks. A way to defend against passive attacks is via gossiping, by which nodes contact random nodes in the network, informing them of their existance in the network as well as all the nodes that it knows off. This prevents information leakage. However, it's vulnerable to active attacks. GuardedGossip combines gossiping and DHT to build to grant immunity against active attacks and significant reduce the possibility of passive attacks through uncertainty.

In addition to chord standard shortcut table (**FT**), every node maintains extra information about the network summarized in the following:

- Gossip List: contains nodes that sent us gossip requests.
- Witness List: contains nodes that we have "seen" during network operations along with a timestamp.
- Guarded List: contains nodes that are safe to establish connections with.

Now the protocol workflow goes as the following: a node is picked from the gossip list and added to the witness list, then fetches the FT of the picked node and entries from that FT go through two checks:

1. **Bound Checking**: compares the estimated average node density that is the difference between a finger entry and its corresponding optimal ID (as if the entire chord ID space is occupied) we denote that density by d, the same process is applied to the fetched FT, again we denote it by d_g bound checking can then be expressed through the following constraint: $d_g < \gamma d$ where γ is the *finger table tolerance factor*.
2. **Witness List Checking**: the witness list is used to detect malicious manipulation in the fetched FT whether some nodes were skipped.

If any of the aforementioned checks failed the entire finger table is discarded; on the contrary if the fetched FT passed only a randomly selected set of nodes are added to the guarded list generating uncertainty. Finally, the entirety of the passed FT is added to the witness list to be utilized for future checks. Furthermore, the protocol maintains its lists by removing random entries from both gossip and guarded lists generating even more uncertainty, and for the witness list entries are deleted based on their timestamp.

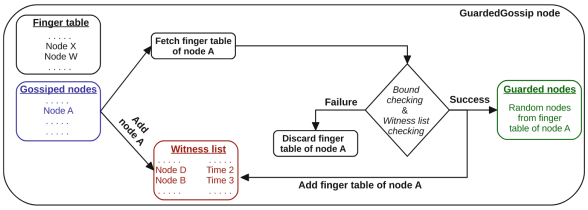

Figure 5: GuardedGossip protocol workflow [12]

3.2 GossipSub

GossipSub is a message propagation protocol [20], and is planned to be used in the Ethereum 2.0 Blockchain [9], GossipSub combines ideas from Gossip and PubSub protocols to achieve efficient and secure message propagation, in section 2.2 we discussed the importance of publishing transactions and blocks to the network to maintain network security through consensus.

It does so by first constructing a mesh of local nodes, that is a partial view of the network. Whenever a new message is received it's immediately forwarded to all nodes in the local mesh, as well as, sending meta-data about messages we have rather than the message itself to random nodes in the network. Furthermore, each node keeps a score of its local mesh nodes via a scoring function, the score is not shared with other nodes in the network but rather kept private. Now nodes can make more informed decisions on nodes to keep in its local mesh and nodes to prune from the mesh.

GossipSub provides an arsenal of defensive mitigation strategies to protect against the following attacks: Sybil, eclipse, censor, cold boot flash and covert flash.

1. **Controlled Mesh Maintenance**: The

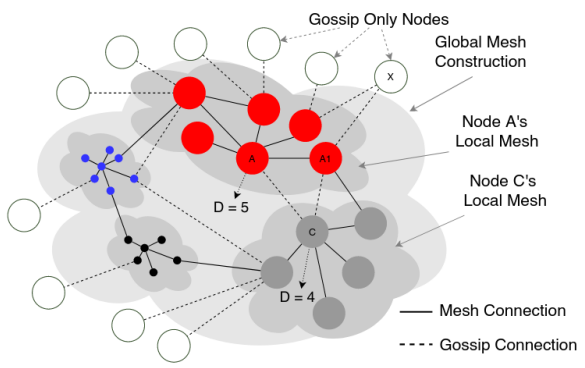

Figure 6: The GossipSub Mesh Construction [20]. D refers to the mesh degree (i.e., the number of nodes in the local mesh)

local mesh hash two thresholds D_{high} and D_{low} each node attempts to keep its mesh degree D between these two thresholds by pruning (removing) or grafting (adding) nodes based on the score formed by the scoring function during network operation.

2. **Opportunistic Grafting**: A node can get stuck in a mesh of bad performing peers and in this situation it might take a while before any new good performing peers are introduced, and so *opportunistic grafting* mechanism checks the median score of current mesh and attempts to graft nodes that have score higher than median of the current mesh, this allows nodes with bad performing peers to quickly recover.

3. **Flood Publishing**: newly published messages are sent to all known peers with a positive score that are subscribed to the topic.

4. **Adaptive Gossip Dissemination**: a gossip factor is chosen to determine how many peers that are not part of the local mesh receive gossip from the set of peers we know of the gossip factor was set to 25% so that every node has a 50% chance of being picked during a gossiping round. This mechanism improve resistance against eclipse attacks.

5. **Backoff on PRUNE**: Whenever a node is pruned in order to keep the mesh degree below D_{high} it gets a *backoff* during which it cannot send any GRAFT messages back to join that mesh again.

3.3 Saving Attack

Ethereum 2.0 proposed two new fork choice rules [7] based on GHOST, fresh-message-driven (FMD) and latest-message-driven (LMD). Gasper [4] is a time-slot based PoS consensus mechanism that user Casper FFG [3] together with LMD GHOST to choose the canonical chain of the network. Block proposal window is limited to a time **slot** every slot lasts for **12 seconds** and every **32 slot** make up an **epoch**. Before an epoch starts individual peers are assigned as validators to each time slot and a committee of validators is chosen to vote on the proposed block. The Saving Attack [11] shows the possibility of malicious peers to delay the finality thus disrupting the network and deviating the canonical chain from its current state to a modified state based on the blocks that had been "saved" by the attacker, [11] shows that FMD GHOST significantly reduces the impact of the saving attack.

4 Evaluation

In this section we present our results that we achieved while trying to replicate the simulation experiments performed in [12], the experiment was replicated with a major difference the network size. Although, our experiment network size is small we were able to achieve the same results, which proves that GuardedGossip is applicable in small networks as well as being able to scale.

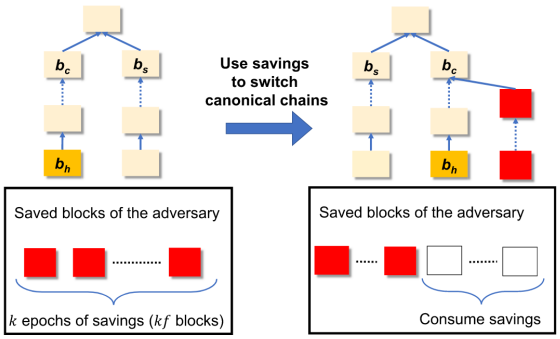

Figure 7: Outline of the saving attack [11]. b_c denotes the canonical chain, b_s denotes the secondary chain (that is the best candidate chain after the canonical one) and b_h denotes the current head block

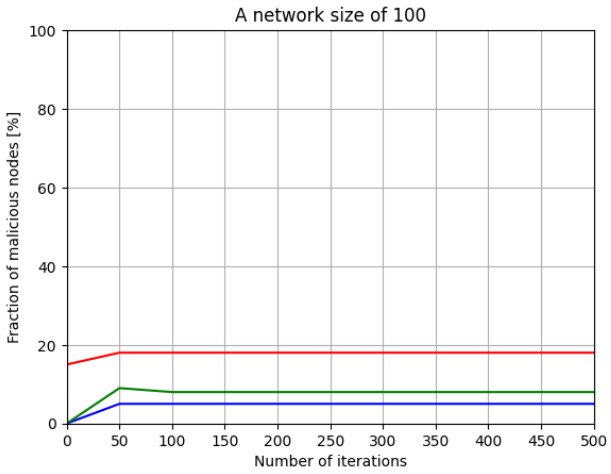

Figure 8: Here we show that our results match the result from [12] in a network of relatively low size.

5 Conclusion and Future Work

Pure P2P solutions are the way forwards, whenever we propose any form of hierarchy it automatically presents a target for attacks. The area of PoS based blockchains is still new and the first major blockchain to widely adopt it is Ethereum which will attest to the security and performance of PoS for more blockchain applications to adopt, or perhaps we need a better consensus algorithm? One thing that is certain PoW is not sustainable and demands an urgent replacement.

On the other hand we can see the potential that a protocol such as GuardedGossip [12] has in scaling Tor while preserving its security. However, it still has its flaws, as malicious peers can still find their into the guarded nodes list, extending the protocol with a scoring function like in [20] might have the potential to significantly reduce the percentage of malicious nodes in the guarded list or perhaps even eliminate their existence in the network? We leave these questions for future works to further investigate.

References

[1] Iddo Bentov, Ariel Gabizon and Alex Mizrahi. "Cryptocurrencies Without Proof of Work". In: *Financial Cryptography and Data Security*. Ed. by Jeremy Clark et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 142–157. ISBN: 978-3-662-53357-4.

[2] *Bitcoin Press Release by the Central Bank of Egypt*. 2016. URL: `https://www.cbe.org.eg/en/Pages/HighlightsPages/Bitcoin%20Press%20Release.aspx` (visited on 13/09/2022).

[3] Vitalik Buterin and Virgil Griffith. "Casper the Friendly Finality Gadget". In: *CoRR* abs/1710.09437 (2017). arXiv: `1710.09437`. URL: `http://arxiv.org/abs/1710.09437`.

[4] Vitalik Buterin et al. "Combining GHOST and Casper". In: *CoRR* abs/2003.03052 (2020). arXiv: `2003.03052`. URL: `https://arxiv.org/abs/2003.03052`.

[5] Bram Cohen. "Incentives build robustness in BitTorrent". In: *Workshop on Economics of PeertoPeer systems* 6 (June 2003).

[6] Cynthia Dwork and Moni Naor. "Pricing via Processing or Combatting Junk Mail". In: *Advances in Cryptology — CRYPTO' 92*. Ed. by Ernest F. Brickell. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 139–147. ISBN: 978-3-540-48071-6.

[7] *Ethereum 2.0 Specifications*. URL: https://github.com/ethereum/consensus-specs (visited on 13/09/2022).

[8] Satoshi Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: (Mar. 2009). URL: https://bitcoin.org/bitcoin.pdf.

[9] *Netoworking Layer — ethereum.org*. URL: https://ethereum.org/en/developers/docs/networking-layer/#gossip (visited on 13/09/2022).

[10] *New OONI & AFTE Report Details The State of Internet Censorship in Egypt*. 2018. URL: https://blog.torproject.org/egypt-internet-censorship/ (visited on 13/09/2022).

[11] Kai Otsuki, Ryuya Nakamura and Kazuyuki Shudo. "Impact of Saving Attacks on Blockchain Consensus". In: *IEEE Access* 9 (2021), pp. 133011–133022. DOI: 10.1109/ACCESS.2021.3115131.

[12] Andriy Panchenko et al. "GuardedGossip: Secure and Anonymous Node Discovery in Untrustworthy Networks". In: *Security and Privacy in Communication Networks*. Ed. by Joaquin Garcia-Alfaro et al. Cham: Springer International Publishing, 2021, pp. 123–143. ISBN: 978-3-030-90019-9.

[13] Yonatan Sompolinsky and Aviv Zohar. "Secure High-Rate Transaction Processing in Bitcoin". In: *Financial Cryptography and Data Security*. Ed. by Rainer Böhme and Tatsuaki Okamoto. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 507–527. ISBN: 978-3-662-47854-7.

[14] William Stallings. *Cryptography and network security : principles and practice*. eng. Seventh edition. Boston: Pearson, 2017. ISBN: 9780134444284.

[15] M. van Steen and A.S. Tanenbaum. *Distributed Systems*. 3rd ed. 2017. URL: distributed-systems.net.

[16] Ion Stoica et al. "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications". In: *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. SIGCOMM '01. San Diego, California, USA: Association for Computing Machinery, 2001, pp. 149–160. ISBN: 1581134118. DOI: 10.1145/383059.383071. URL: https://doi.org/10.1145/383059.383071.

[17] *Tor Directory Protocol, Version 3*. 2018. URL: https://gitweb.torproject.org/torspec.git/tree/dir-spec.txt (visited on 13/09/2022).

[18] *Torproject.org Blocked by GFW in China: Sooner or Later?* 2008. URL: https://blog.torproject.org/torprojectorg-blocked-gfw-china-sooner-or-later/ (visited on 13/09/2022).

[19] Harald Vranken. "Sustainability of bitcoin and blockchains". In: *Current Opinion in Environmental Sustainability* 28 (2017). Sustainability governance, pp. 1–9. ISSN: 1877-3435. DOI: https://doi.org/10.1016/j.cosust.2017.04.011. URL: https://www.sciencedirect.com/science/article/pii/S1877343517300015.

[20] Dimitris Vyzovitis et al. "GossipSub: Attack-Resilient Message Propagation in the Filecoin and ETH2.0 Networks". In: *CoRR* abs/2007.02754 (2020). arXiv: 2007.02754. URL: https://arxiv.org/abs/2007.02754.