May 6, 2021

# 1   Bitcoin peer-to-peer protocol

## 1.1   Connection and peer discovery

In order to establish a connection two peers need to exchange version messages on a TCP channel, by default on port 8333, in an handshake fashion. The node that first sends a `version` message is said to be establishing an *outgoing connection*. The receiving peer will eventually send back a `verack` message followed by another `version` message. This latter node is said to be establishing an *ingoing connection*. Once the second `version` message is answered with a `verack` the connection is set up. Each node is able to establish up to 8 outgoing connections and 117 ingoing connections, for a total of 125 connections. In order to keep its connections active, each peer sends a message to each neighbour at least once every thirty minutes. If more ninety minutes pass without receiving anything from a neighbour, the connection is dropped. Each node always tries to keep eight outgoing connections active, therefore upon dropping an outgoing connection a node tries to connect to another address from its peer cache.

Nodes on fresh bootstrap make use of hardcoded *DNS seeds* as their first mean to discover other peers. DNS seed servers are maintained by community members and answer queries providing a list of addresses that can be either dynamically gathered through a periodic scan of the network or manually updated by server administrators. As a fallback option the user is able through command line to specify a list of addresses the client can connect to. Were both these two options to fail the client has a hardcoded list of peers it can directly connect to, though this is considered the last resort for bootstrapping peers. Nodes on startup that were previously in the network shall first lookup for peer names in their local address database, implemented as "peer.dat". The database contains the address of each peer the node has come to know during its lifetime in the network. If the node has disconnected for a time too long, many of the addresses in the database may have become outdated or unreachable. A node that cannot connect to any address in the peer database or has spent up to eleven seconds trying to connect unsuccessfully to at least one of the peers in the database behaves as on fresh bootstrap and resolves to query a DNS first. The use of a local address database, also called *peer cache*, provides reconnecting peers a fully-decentralized way to join the network and is the first line of defence against *fake bootstrap attacks*; the topic along with other security concerns will be discussed in Chapter 2.

The peer discovery process after the first connection of a node is carried on through the exchange of `addr` messages containing the address and port number of other peers in the network. Once a connection has been set up the two nodes exchange `addr` messages,

providing each other with addresses from their local peer database. Furthermore every twenty-four hours each node sends to its neighbours an `addr` message containing only its own address in order to advertise itself in the network. `addr` messages can contain at most one thousand addresses; additionally those containing ten or fewer addresses are relayed. This behaviour greatly contributes to the gossiping of self-advertisement messages even though messages are relayed only to a small subset of neighbours, namely to a couple of peers.

## 1.2 Peer database structure

The local peer database or cache serves the purpose of storing the addresses a node has come to know from `addr` messages and DNS seeds. The database is constituted of the `new` and the `tried` tables. The latter contains addresses of peers with whom the node has successfully established a connection in the past. It consists of 64 buckets that can store up to 64 unique addresses. Buckets are selected through a hash on the IP address. The `new` table contains the addresses the node has not connected to and is therefore larger: it has 256 buckets of 64 addresses each.

# 2 Bitcoin network security concerns

Bitcoin's decentralized, uncontrolled environment is open to a wide range of attacks, many of which can be applied to other cryptocurrencies as well. For this reason the following attacks could be considered attack paradigms, with the same rationale but different implementation amongst different cryptocurrency systems.

Attacks like *double-spending* aim to exploit systems deficiences: a malicious user tries to spend two times, in different transactions, the same set of coins. Although proof-of-work consensus makes double-spending infeasible, the attack can be still executed with other approaches that manipulate peers, as with network partitions, or were other conditions to be met [6]. Double-spending also can be carried out were an attacker to hold the majority of the computing power, this condition is also known as *51%* or *majority* attack [10]. With the majority of the network's hash rate an attacker would also be able to fork the blockchain to its advantage and prevent transactions and blocks from being verified, therefore this represents the worst security threat for any cryptocurrency. Blockchain forks can be exploited to engineer block races as in *selfish mining* attacks, were the malicious user intentionally holds private his mined blocks [5],[7]. If the attacker successfully keeps his private chain longer then the public one it can release his mined blocks, thus claiming the rewards for himself and wasting other miners' computing power.

These are just a few attacks that can be driven onto Bitcoin: this chapter serves only introductory purposes. In the following sections there is more on network attacks, which are the focus of this thesis. The interested reader can find complete security overviews in the works of Conti et al., Saad et al. and Wang et al. [1], [9], [12].

## 2.1 Network security

Peer caches have been widely implemented so far as they are a fast and fully-decentralized way to rejoin the network after a disconnection. They allow P2P systems to overcome other major issues at bootstrap time related to presence of a single point of failure, i.e. bootstrap servers, and are the most scalable and efficient solution when compared to other decentralized mechanisms such as *address probing* [3]. Nevertheless, the use of peer caches introduces new security concerns for the network: the manipulation of the peer database performed by one or more cooperating malicious nodes can easily lead to the monopolization of a node, or group of nodes, connections', thus creating a network partition. This condition, often referred to as the *eclipsing* of node, is inherently dangerous for it leaves the victim open to *double-spending*, *selfish mining* and *51%* attacks.

## 2.2 Fake bootstrapping attack

Every P2P system needs some mean to let nodes join the network: in Bitcoin bootstrapping nodes need to rely on the information provided by some other peer in the network. In a fake bootstrapping attack the peers contacted while a node is still initiating its connections, and hence its knowledge of the network, flood the bootstrapping peer with malicious addresses. The results in Chapter 3 show how feeding bogus information to a joining node can seriously influence its view of the network and lead to network partitions and peer eclipsing. This condition would also increase the efficacy of other network-level attacks, such as a DDoS, and the success rate of spending and mining attacks [4].

Prevention is done by ensuring a node can contact a trustworthy peer when joining the network. Peer caches and centralized reliable bootstrap services serve this purpose, allowing joining nodes to make use of old neighbours while having central bootstrap servers as a fallback. Other solutions have been implemented and tested over time, though their drawbacks usually exceed the benefits. As an example randomly probing the address space, scouting for peers, decentralizes the bootstrap process and increases the variance in the network knowledge, but may not work if peers are behind NATs or firewalls and increases the latency for joining the network [3] [2]. In 2012, Touceda et al. discuss fake bootstrapping along with other security threats in their survey [11].

Bitcoin employs many countermeasures, and overcomes other bootstrap issues, with its peer cache implementation mixed with external mechanisms: each node contacts its stored addresses for successive connections and aims to establish multiple outgoing connections, thus not it does not rely on a single bootstrap node. On top of that, DNS seeds provide a reliable external bootstrap service and the user is given the ability to manually insert addresses to connect to [8].

## 2.3 Eclipse attack

During an eclipse attack the victim is flooded with a multitude of incoming connections from an attacker and fed with bogus network information, i.e. peer addresses, that will fill up the peer cache. Upon restarting, with high probability the victim will form all of its eight outgoing connections with malicious IP addresses, thus finding itself eclipsed. As

described in Section 2 this greatly enhances the efficacy and impact of other attacks. In 2015, Heilman et al. widely describe eclipse attacks and suggest many countermeasures to harden the network, a few of which were implemented [4]. As part of the adoption of these countermeasures, the number of buckets in `new` and `tried` has been increased, the addresses have become deterministically hashed to a single slot inside a bucket and the heavy bias towards addresses with fresher timestamps when choosing new connections has been removed.

## 2.4 Sybil attack

## 2.5 DDoS attack

Distributed Denial of Service (DDoS).

# 3 Results

# References

[1] Mauro Conti, E. Sandeep Kumar, Chhagan Lal, and Sushmita Ruj. A survey on security and privacy issues of bitcoin. *IEEE Communications Surveys Tutorials*, 20(4):3416–3452, 2018.

[2] C. Cramer, K. Kutzner, and T. Fuhrmann. Bootstrapping locality-aware p2p networks. In *Proceedings. 2004 12th IEEE International Conference on Networks (ICON 2004) (IEEE Cat. No.04EX955)*, volume 1, pages 357–361 vol.1, 2004.

[3] Jochen Dinger and Oliver P. Waldhorst. Decentralized bootstrapping of p2p systems: A practical view. In Luigi Fratta, Henning Schulzrinne, Yutaka Takahashi, and Otto Spaniol, editors, *NETWORKING 2009*, pages 703–715, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[4] Sharon Goldberg Ethan Heilman. Alison Kendler, Aviv Zohar. Eclipse attacks on bitcoin's peer-to-peer network. Cryptology ePrint Archive, Report 2015/263, 2015. https://eprint.iacr.org/2015/263.

[5] Ittay Eyal and Emin Gun Sirer. Majority is not enough: Bitcoin mining is vulnerable, 2013.

[6] Ghassan O. Karame, Elli Androulaki, and Srdjan Capkun. Double-spending fast payments in bitcoin. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, page 906–917, New York, NY, USA, 2012. Association for Computing Machinery.

[7] Tin Leelavimolsilp, Long Tran-Thanh, and Sebastian Stein. On the preliminary investigation of selfish mining strategy with multiple selfish miners, 2018.

[8] Mahmoud Mostafa. Bitcoin's blockchain peer-to-peer network security attacks and countermeasures. *Indian Journal of Science and Technology*, 13:767–786, 02 2020.

[9] Muhammad Saad, Jeffrey Spaulding, Laurent Njilla, Charles Kamhoua, Sachin Shetty, DaeHun Nyang, and Aziz Mohaisen. Exploring the attack surface of blockchain: A systematic overview, 2019.

[10] Sarwar Sayeed and Hector Marco-Gisbert. Assessing blockchain consensus and security mechanisms against the 51 *Applied Sciences*, 9(9), 2019.

[11] Diego Suarez Touceda, Jose M. Sierra, Antonio Izquierdo, and Henning Schulzrinne. Survey of attacks and defenses on p2psip communications. *IEEE Communications Surveys Tutorials*, 14(3):750–783, 2012.

[12] Hao Wang, Chunpeng Ge, and Zhe Liu. On the security of permissionless blockchain systems: Challenges and research perspective. In *2021 IEEE Conference on Dependable and Secure Computing (DSC)*, pages 1–8, 2021.