# An open measurement dataset on the Bitcoin P2P Network

Jean-Philippe Eisenbarth *, Thibault Cholez *, Olivier Perrin *
*Universite de Lorraine, CNRS, Inria,
*LORIA, F-54000 Nancy, France*
Email: {jean-philippe.eisenbarth, thibault.cholez, olivier.perrin}@loria.fr

***Abstract*—The Bitcoin blockchain is managed by an underlying peer-to-peer network. This network is responsible for the propagation of transactions carried out by users via the blocks (which contain the validated transactions), and to ensure consensus between the different nodes. The quality and safety of this network are therefore particularly essential. In this work, we present an open dataset on the peers composing the Bitcoin P2P Network that was made following a well defined and reproducible methodology. We also provide a first analysis of the dataset on three criteria: the number of public nodes and their client version and geographical distribution.**

***Index Terms*—Blockchain, Bitcoin p2p network, network measurement, p2p protocol, network security**

## I. Introduction

During the last decade, cryptocurrencies have gained in popularity. Even if their total market capitalization is prone to much variation, they are currently valued to more than $937 billion [1]. With more than 18 million of circulating tokens valued to $602 billion, Bitcoin is the most acknowledged cryptocurrency. It is based on a decentralized blockchain which is a distributed transaction ledger whose records are append-only and publicly auditable. This ledger is managed by a underlying peer-to-peer (p2p) network that agrees on a common order of the transactions using a distributed consensus. There is no trusted party to regulate this system. In its initial white paper [2], Satoshi Nakamoto argues that this system prevents attackers to alter the ledger and double spend tokens for as long as an attacker does not control 51% or more of the computational power in the p2p network. Such a combination of a public ledger and a p2p network that participates in the consensus and can audit it is called a permissionless blockchain.

Due to the popularity and success of Bitcoin, its protocols, network and security have been widely studied [3], [4], [5], [6], [7] [8] [9] [10]. To the best of our knowledge, even if some authors indicate their methodology, either their dataset are not publicly available or the source of the software used (crawlers, scripts to build the statistics, algorithms...) are not given, preventing reproducible results. None also proved that the crawler they used actually converges and is sound.

In this paper, we present our open dataset [11] composed of snapshots of the Bitcoin network and make a comprehensive analysis of it. During one month, we performed daily crawls on the nodes composing the network and gathered information about them. Our dataset [11] and all related tools [12], more precisely the crawler to make it and the scripts to analyze it are also open source, thus making our study fully reproducible and expendable, by opposition to previous studies in the domain. From our dataset, we present a first couple of metrics that characterize the network: the size of the network, the clients' distribution and the geographical localization of peers.

The rest of the paper is organized as follows. Section II presents some background on the Bitcoin network and its protocol. In Section III, we present our crawling strategy to make the dataset that is then analyzed in Section IV. Finally, Section V concludes the paper.

## II. Background

In this paper, we focus on the implementation of the Bitcoin Core client [13], which is the official implementation, documented in the Bitcoin's community documentation [14].

There are four main types of messages in the Bitcoin's p2p network:

- network discovery messages
- transactions or blocks announcements
- transactions or blocks requests
- transactions or blocks deliveries

We will focus on the network discovery messages (also called overlay network protocol). Now we will describe how the Bitcoin p2p network works.

There are two types of nodes in the Bitcoin p2p network: the full nodes and the lightweight nodes. A full node downloads every block and transaction and verifies that they respect all the Bitcoin's consensus rules. If it is not the case, the invalid blocks or transactions are rejected, no matter the network's global state about them. In order to use Bitcoin in a trustless and private way, a user needs to run a full node wallet. In contrast, a lightweight client only downloads the block headers to verify the authenticity of the transactions. This method is called Simplified Payment Verification (SPV). This type of client relies upon full nodes which become trusted third parties.

Essentially, when a full node starts up, it needs to connect to other active peers of the network to synchronize its local blockchain state with the one the network agreed upon. At

978-3-903176-32-4 

the very first startup, a node doesn't know any other peers of the network. Most likely, it will use the DNS seeders to learn about them. Basically the DNS seeders continuously crawl the network to create a list of active reachable peers. When they receive a DNS request from a new node, they reply with a subset of chosen IP addresses they know. The list of DNS hostnames to query is hardcoded in the Bitcoin client (e.g. seed.bitcoin.sipa.be, dnsseed.bluematt.me, dnsseed.bitcoin.dashjr.org, etc.).

A node has two databases in which it stores IP addresses of other peers of the network: the `new` and the `tried` tables. These two tables are organized into buckets, 1024 buckets for the `new` table and 256 for the `tried` table. Each bucket can contain a maximum of 64 entries. The `new` table contains the addresses of peers to which the node has not yet tried to connect to. The `tried` table contains the addresses of peers to which the node has been able to connect successfully and that are known as reachable.

After the local database has been seeded with some active peers, the node tries to connect to 8 peers by default using unencrypted TCP connections (called outbound connections). Additionally, the node can be configured (port forwarding, access enabled in the firewall, etc.) to accept connections (called inbound connections) from other peers, up to 117 by default. Such a node — called a *listening node* — may issue unsolicited `ADDR` messages to advertise its neighbors that it accepts inbound connections. These neighbors may relay this message to their own neighbors and the distribution may carry on this way. A node could also request explicitly to discover other active peers in the network by sending a `GETADDR` message to its neighbors. To determine how many addresses a node needs to answer, it calculates $m$ the minimum between 23% of the number of nodes contained in all the buckets and a maximum number set to 2,500. Then, it responds with `ADDR` messages containing information (ip, port, timestamp) of the $m$ peers retrieved randomly from the buckets (a single `ADDR` message can contain up to 1000 buckets entries information). At maximum, a node would send 3 messages, each containing respectively 1000, 1000 and 500 addresses from its tables. This is how the ADDR messages are documented for the Bitcoin Core client but we noticed that it is no longer the case for the versions released after 2014 ($>$ v0.10.0). Since then, a node sends at maximum 1000 addresses in one message. The list of addresses that contains 23% of the number of the nodes is cropped to 1000. The reason is not documented and not clear, even inside the community and they plan to remove this limitation in the future [15].

Also, a node maintains its connections up-to-date by verifying periodically the state of the nodes it is connected to by issuing application-level `PING` messages and waiting for the `PONG` responses. The two types of connection (inbound and outbound) are used to disseminate transaction and block messages (announcement, request and delivery).

## III. Measurement strategy

### A. Crawling methodology

From February 7–March 2, 2020 we operated a crawling node on the Bitcoin's main network. We forked and improved the crawler used by the bitnodes.io website and we published it on Github [12]. The methodology involves first a crawler that tries to concurrently establish connections to the peers of the Bitcoin network and second, another program that runs in parallel and that keeps track of the connectivity of the discovered nodes. This program connects to all the nodes returned by the crawler and sends periodically (every 60 seconds) a PING message to the connected nodes to verify if they are still connected. When a crawl is finished, the nodes that are tagged as reachable by this program are exported with additional information such as geolocalization, hostname, Bitcoin client used, among others.

It is important to note that our infrastructure lacked an IPv6 link and a TOR proxy to test and integrate these nodes. Thus, all the discovered nodes are IPv4 nodes. They accounted for approximately 75% of the network whereas IPv6 and Tor nodes accounted for 10% and 15% of the network respectively. Please note that in the dataset we publicly released [11], IP addresses are pseudo-anonymized following ICANN recommendations [16] which offers an acceptable compromise between utility and privacy. More precisely a salted hash is provided and the last byte of the address is set to 0.

Also, nodes that are running a Bitcoin protocol version older than 70001 were not taken into account during a crawl because of compatibility issues with prior versions of the protocol. It corresponds to the Bitcoin Core v0.8.0 client (user agent Satoshi:0.8.0) and below (before February, 2013). Pappalardo et al. [17] showed that already in 2016 the vast majority of nodes in the network were running a protocol version more recent than 70001. Consequently, we consider that the crawler could reasonably ignore the nodes that are running an older version of the protocol.

### B. Technical setup

The machine we used is an Ubuntu 18.04 LTS computer with an Intel Xeon Processor E5-2420 v2 6 cores, 16GB RAM and a 1 Gb/s network link that established connections to all the IPv4 reachable peers — i.e. listening nodes — of the Bitcoin peer-to-peer network. To handle the large number of network connections needed, we had to increase the opened files limit of our GNU/Linux system to 1,000,000. We also set some network kernel settings as specified below. The machine operated the crawling node 24/7 for the whole period.

```
net.core.rmem_default=33554432
net.core.wmem_default=33554432
net.core.rmem_max=33554432
net.core.wmem_max=33554432
net.ipv4.tcp_rmem=10240 87380 33554432
net.ipv4.tcp_wmem=10240 87380 33554432
```

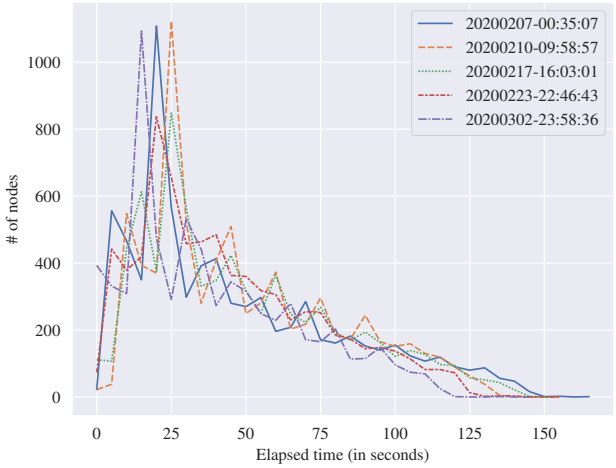


Fig. 1. Number of new reachable nodes discovered throughout five crawls.

```
net.ipv4.ip_local_port_range=2000 65500
```

### C. Validation

In order to validate the quality and the consistency of our measurements, we want to assess that a crawl ends when all possible peers are discovered. A crawl proceeds as follows:

1) the crawling node is bootstrapped by receiving a small subset of the reachable nodes from known DNS seeders which addresses[1] are hardcoded in the client ;
2) it tries to establish a connection to the newly discovered nodes ;
3) then it sends `GETADDR` messages to all the nodes it is connected to ;
4) the crawling node disconnects from a node once it received the `ADDR` messages in response to its `GETADDR` message ;
5) it loops back to step 2 until the crawling node does not discover new nodes (one it did not process yet) in a certain amount of time.

More precisely, the newly discovered nodes are stored as pending and the crawling node tries to establish a connection to them. These nodes are retrieved from the `GETADDR` sent by the nodes the crawler is connected to. This list of addresses is filtered in order to dismiss stale nodes (lastly seen online more than 8 hours from the current time of crawl) for efficiency purpose (older nodes are not likely to be connected). A crawl stops when there are no pending nodes left (checked every 10 seconds). In practice a crawl did not exceed 3 minutes and the program was set to restart a new crawl every 4 minutes. As shown in Fig. 1, a crawl stops learning new nodes after 2 to 3 minutes and the number of new nodes discovered over time is decreasing after a peak at the beginning.

[1]dnsseed.bitcoin.dashjr.org, dnsseed.bluematt.me, seed.bitcoin.sipa.be, seed.bitcoinstats.com, seed.bitcoin.sprovoost.nl and seed.bitnodes.io

TABLE I
THE NUMBER OF NODES DISCOVERED FOR ONE CRAWL AND THE TOTAL NUMBER OF REACHABLE NODES AT ONE POINT (THE TIMEZONE IS UTC+1)

| | Minimum | Maximum | Average | Median |
|---|---|---|---|---|
| Discovered nodes | 5,506 (2020-02-11 12:26) | 7,623 (2020-02-9 22:14) | 7,218 | 7,310 |
| Reachable nodes | 6,856 (2020-02-17 14:32) | 8,103 (2020-02-10 2:03) | 7,783 | 7,870 |

Also, we deployed five independent ground-truth Bitcoin Core nodes (straight v0.20.1). They joined the p2p network and four of them had approximately 30 active connections whereas the fifth node had roughly 70 active connections. The crawler was able to find all of them. The methodology of the crawler can reasonably be considered effective in finding almost all accessible nodes in the Bitcoin p2p network.

## IV. First Analysis of the Bitcoin's Network Dataset

In this section, we analyze several metrics that are derived from the crawls of the Bitcoin network.

### A. Number of nodes

As shown in Table I, the crawler was able to discover in average $7,218$ unique IPv4 nodes in the p2p network during a single crawl. The minimum number of reachable nodes found was $6,856$[2] and the maximum 8,103. The program responsible to keep the connection opened with all the previously discovered nodes was connected in average to $7,783$ nodes throughout the period. The difference between the discovered and reachable nodes can be explained by the fact that a small portion of the nodes can be missed by a particular crawl but found by the others and the program that keep track of the reachable nodes is regularly updated with the newly discovered node by the crawler.

We also have an overall picture of the total number of nodes referenced in the p2p network (not only the reachable ones). As usual, the crawler sends a single `GETADDR` message to a newly discovered reachable node, that responds with 1000 nodes contained in all its buckets (see II). We consider here the unique nodes discovered per crawl with no consideration regarding their reachability, and we found that there are 200,000 unique nodes in average retrieved from the buckets of all reachable nodes. Due to overlapping in the buckets of all the nodes, we can deduct that this number, yet an approximate, should be representative of the average size of the whole p2p network (public and non public nodes).

[2]To calculate the minimum we have omitted the data for the 21 February because we had to restart the program and it needed several crawls to connect to the accurate number of reachable nodes of the p2p network. If we had kept these data, the minimum, maximum, average and median would have been $6,393$ (2020-02-21 10:04 UTC+1, date of the restart), $8,103$ (unchanged), $7,785$, and $7,869$, respectively.

### B. Client distribution

The reachable nodes we discover are all public nodes (i.e. the user that deployed a node did properly configure the firewall and/or port forwarding if necessary) but we cannot conclude on whether they are full or lightweight nodes with only the information that they are public.

To answer this, we classified the public nodes discovered according to which client (also named `user agent`) they used. The official full node client `Bitcoin Core` (its user agent is `Satoshi`) is the most widely deployed client we found, with up to 98.36% of the nodes that are running this client. There are dozens of versions for this client in use. Throughout the whole period of crawl, there are three versions that are the most popular ones: `0.18.0` (May, 2019), `0.18.1` (August, 2019) and `0.19.0.1` (November, 2019). Their relative order varies during the period but these three versions alone represent approximately 65% of the crawled network.

There are other clients that are present in the dataset but in a very small percentage (less than 1%). Their user agents are `bcoin`, `btcd` and `bitcoin_unlimited`. We also found even less significant number of other clients that we could not classify because we could not find trustful information about them. Their user agents are for example: `Aurum`, `CKCoind`, `therealbitcoin.org`, `BitflateCore`, `MultiChain`...

We can deduct that a very vast majority (approximately 99%) of the discovered public nodes are full nodes and this majority is running the official Bitcoin client.

### C. Geographical distribution

The previous section showed that public nodes are full nodes, and we can wonder if they are geographically stable during a whole day.

We show in Fig. 2 the number of reachable nodes on February 15th in the world divided in three areas: Europe / Africa / Middle East, North / South America and Asia / Oceania. We used the time zones as the criteria to gather the areas. As we can see, the number of up nodes over time is quite constant in the three areas. The public full nodes are not shut down by nightfall. They are most likely running on servers rather than on users' personal computers because the periodic daily churn per zone inherent to p2p file sharing networks cannot be seen here. We can also notice that, unlike the geographical distribution of mining nodes, there are more public full nodes deployed in the European continent or in the North America than in Asia. For example, in our entire dataset at any given time, there are in average approximately 3900 nodes hosted in Europe, 2150 in North America and 1200 nodes hosted in Asia. The absence of daily pattern as well as a balanced geographical distribution are good properties for distributed systems making them more robust to failures.

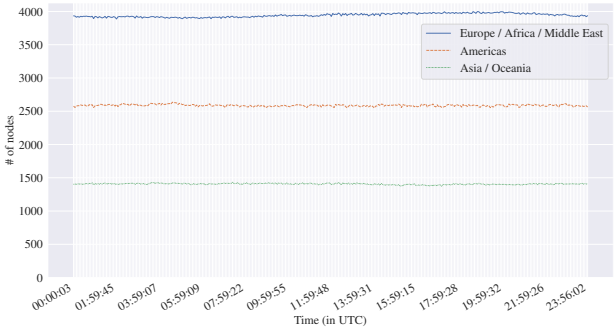


Fig. 2. Geographical distribution of the public nodes over one day

## V. Conclusion and Future Work

We presented our an open measurement dataset on the Bitcoin p2p network. The dataset contains in-vivo measurements over one month of the Bitcoin's public nodes p2p network. We assessed our crawler soundness and made it available as well as the scripts we used to perform analysis in an attempt to provide facilities to reproduce and extend our study. In this paper, we also provided a first insight on the dataset, showing that the size of the Bitcoin p2p network is very stable with approximately 7800 reachable nodes, that more than 98% of the peers execute the official client, and that peers composing the network are well balanced throughout the world.

Nevertheless, it is important to note that the analysis we showed in this paper concerns the public nodes (namely reachable full nodes) in the Bitcoin p2p network and it cannot be trivially generalized to all the nodes in the network (wallets, lightweight nodes, miners...). Yet, these public nodes are crucial, since they act as a backbone of the complete Bitcoin network: wallets and lightweight clients need the public full nodes to connect to the network and retrieve the blocks and transactions.

We already started to further analyze the p2p network data with other metrics that can be derived from our dataset such as the churn rate, the popularity of peers, the list of software vulnerabilities that affect clients' versions or their distribution among the reachable nodes. In our future work, we will pursue our investigation of our dataset to gain even more knowledge on the Bitcoin p2p network.

**Acknowledgement**

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 830927 .

## References

[1] CoinMarketCap, "Global Charts," [accessed on January 22nd 2021]. [Online]. Available: https://coinmarketcap.com/charts/

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[3] C. Decker and R. Wattenhofer, "Information propagation in the Bitcoin network," in *IEEE P2P 2013 Proceedings*. Trento, Italy: IEEE, Sep. 2013, pp. 1–10. [Online]. Available: http://ieeexplore.ieee.org/document/6688704/

[4] A. Biryukov, D. Khovratovich, and I. Pustogarov, "Deanonymisation of Clients in Bitcoin P2P Network," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '14. Scottsdale, Arizona, USA: Association for Computing Machinery, Nov. 2014, pp. 15–29. [Online]. Available: https://doi.org/10.1145/2660267.2660379

[5] A. Miller, J. Litton, A. Pachulski, N. Gupta, D. Levin, N. Spring, and B. Bhattacharjee, "Discovering bitcoin's public topology and influential nodes," 2015.

[6] M. A. Imtiaz, D. Starobinski, A. Trachtenberg, and N. Younis, "Churn in the Bitcoin Network: Characterization and Impact," in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. Seoul, Korea (South): IEEE, May 2019, pp. 431–439. [Online]. Available: https://ieeexplore.ieee.org/document/8751297/

[7] S. Feld, M. Schönfeld, and M. Werner, "Analyzing the Deployment of Bitcoin's P2P Network under an AS-level Perspective," *Procedia Computer Science*, vol. 32, pp. 1121–1126, 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S187705091400742X

[8] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *Proceedings of the 24th USENIX Conference on Security Symposium*, ser. SEC'15. Berkeley, CA, USA: USENIX Association, 2015, pp. 129–144.

[9] A. E. Gencer, S. Basu, I. Eyal, R. van Renesse, and E. G. Sirer, "Decentralization in Bitcoin and Ethereum Networks," in *Financial Cryptography and Data Security*, S. Meiklejohn and K. Sako, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, vol. 10957, pp. 439–457. [Online]. Available: http://link.springer.com/10.1007/978-3-662-58387-6_24

[10] V. Deshpande, H. Badis, and L. George, "BTCmap: Mapping Bitcoin Peer-to-Peer Network Topology," in *2018 IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN)*. Toulouse: IEEE, Sep. 2018, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8548904/

[11] J.-P. Eisenbarth, "Bitcoin p2p network study dataset overview." [Online]. Available: http://concordia-btc-p2p.lhs.loria.fr/index.html

[12] ——, "Github jpeisenbarth/bitnodes forked from ayeowch/bitnodes." [Online]. Available: https://github.com/jpeisenbarth/bitnodes/tree/add_statistics

[13] The Bitcoin Core developers, "Github bitcoin/bitcoin," Bitcoin, [Accessed on March 20th 2020]. [Online]. Available: https://github.com/bitcoin/bitcoin

[14] The Bitcoin community, "Satoshi Client Node Discovery," [Accessed on March 10th 2020]. [Online]. Available: https://en.bitcoin.it/wiki/Satoshi_Client_Node_Discovery

[15] ——, "Cache responses to GETADDR to prevent topology leaks by naumenkogs · Pull Request #18991 · bitcoin/bitcoin." [Online]. Available: https://github.com/bitcoin/bitcoin/pull/18991

[16] Internet Corporation for Assigned Names and Numbers, "Recommendations on Anonymization Processes for Source IP Addresses Submitted for Future Analysis," Aug. 2018. [Online]. Available: https://www.icann.org/en/system/files/files/rssac-040-07aug18-en.pdf

[17] G. Pappalardo, T. Di Matteo, G. Caldarelli, and T. Aste, "Blockchain inefficiency in the Bitcoin peers network," *EPJ Data Science*, vol. 7, no. 1, p. 30, Dec. 2018. [Online]. Available: https://epjdatascience.springeropen.com/articles/10.1140/epjds/s13688-018-0159-3