

*u<sup>b</sup>*

---

*b*

**UNIVERSITÄT  
BERN**

# Seminar Cryptography and Data Security

## Exploring the Bitcoin P2P network

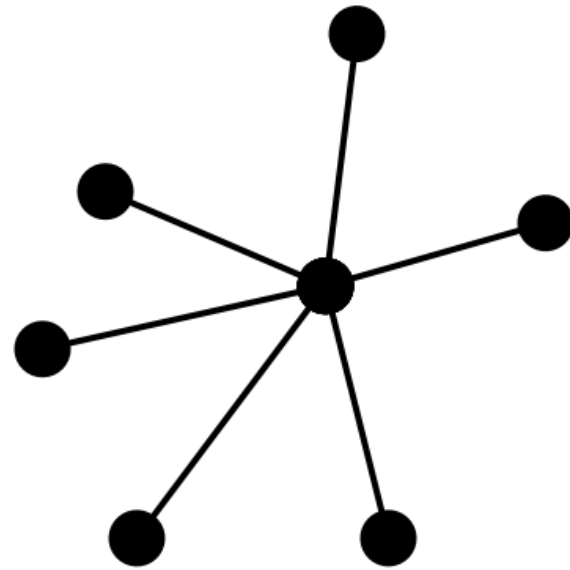
**Chris Rüttimann & Thushjandan Ponnudurai**

Bern, 6. April 2022

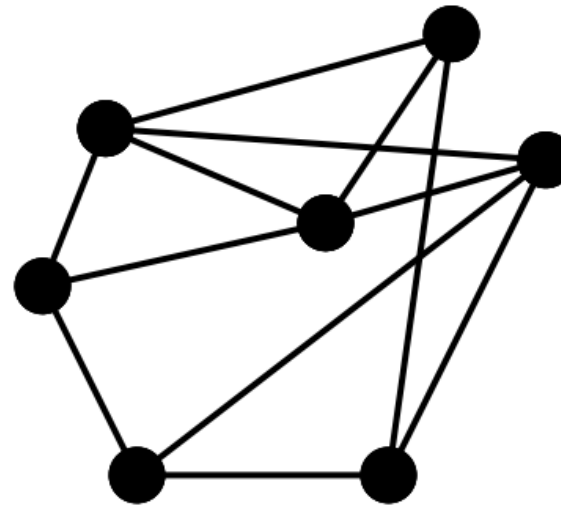
# Outline

- Peer to Peer fundamentals
- How Bitcoin P2P networks work
- Bitcoin P2P protocol
- Deanonymization attacks
  - Mitigations
- Our Bitcoin mainnet network analysis

# Peer to Peer Network



Server based



P2P

# Bitcoin P2P Protocol

## Public P2P networks

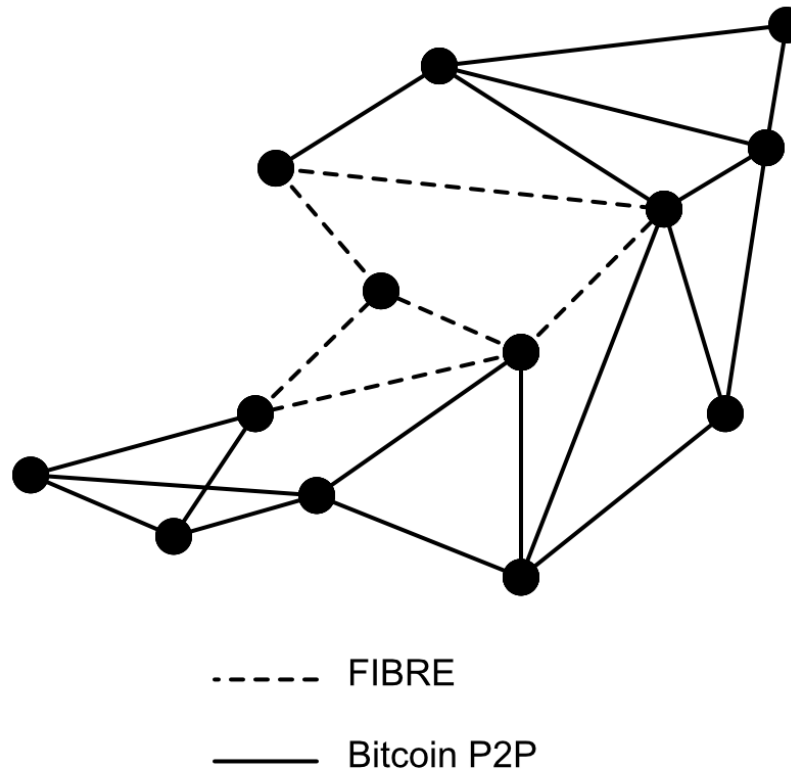
- Different P2P networks
  - Mainnet
  - Testnet
  - signet
  - Regtest (local)
- Differentiation over TCP ports, magic bytes and use cases

# Bitcoin P2P Protocol

## Private Relay Network

- FIBRE (Fast Internet Bitcoin Relay Engine)
  - UDP
  - Error Correction
- Bitcoin Relay Network
  - Predecessor of FIBRE

# Bitcoin P2P Protocol



# Bitcoin P2P Protocol

## How to find peers?

- DNS Record

- Bitcoin Main Network:

- <https://github.com/bitcoin/bitcoin/blob/master/src/chainparams.cpp#L121>

- `vSeeds.emplace_back("seed.bitcoin.sipa.be."); // Pieter Wuille, only supports x1, x5, x9, and xd`

- `vSeeds.emplace_back("dnsseed.bluematt.me."); // Matt Corallo, only supports x9`

- addr messages

- Sent as response to getaddr message

- Sent by peers to announce new peers



# Bitcoin P2P Protocol

## Version message

- Information about transmitting node
- Needs to be exchanged before other messages
- Followed by a verack message

# Bitcoin P2P Protocol

## Version message

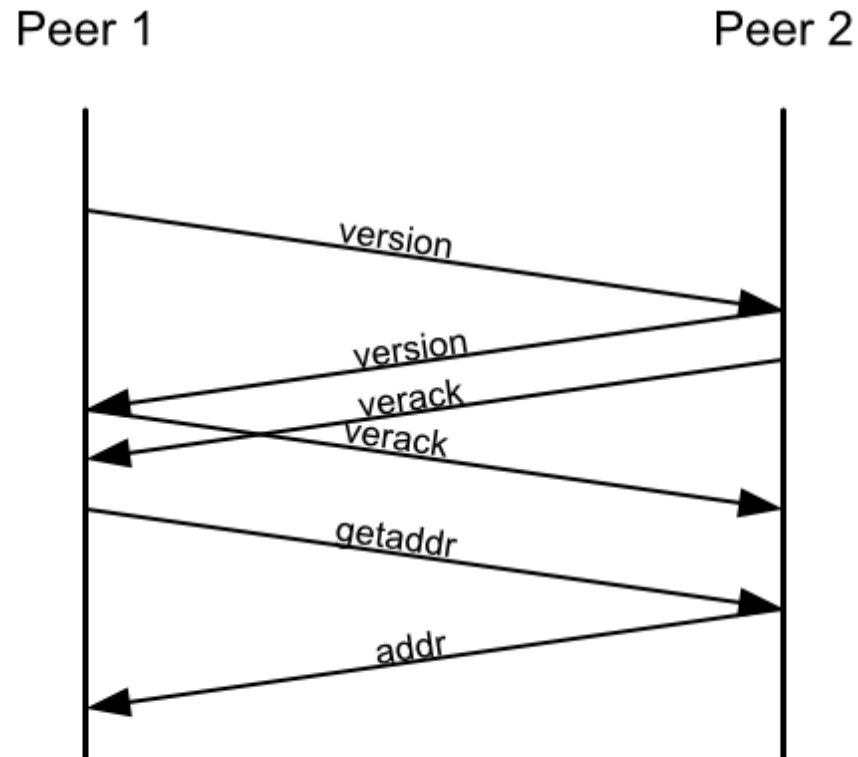
0b110907	# Magic
76657273696f6e0000000000	# Command
56000000	# Length
a71a4acd	# Checksum
7f110100	# Version
0000000000000000	# Services
9298396200000000	# Timestamp
0000000000000000	# addr_recv services
00000000000000000000ffff	# addr_recv IPv6 part
c0a80014	# addr_recv IPv4
075b	# port_recv
0000000000000000	# from services
000000000000000000000000	# sender IPv6 part
00000000	# sender IPv4 part
0000	# sender port
ab1e7b09b53138b4	# Nonce
00	# Agent Bytes
52190900	# Starting Height
00	# relay

Header

Payload

# Bitcoin P2P Protocol

## Connecting to peers



# Bitcoin P2P Protocol

## Demo

# Bitcoin P2P Protocol

## Addrman

- A node maintains 256 buckets, which can save up to 64 address
  - Total 16'384 addresses
- GetAddr returns only a subset
  - 23% from addrman
  - Max 1000 addresses
- Endpoints pseudo randomly from addrman selected
  - Nodes from different ASN or address range (/16 for IPv4) are chosen
- A node can maintain up to 125 connections (default)
  - 10 outbound connections

# Bitcoin P2P Protocol

## Misbehavior

- IPs can be banned for 24h for rogue behaviour
  - Penalty score
- Not following the protocol
  - Invalid data
    - Trying to build invalid chain
    - Valid header with invalid tx
  - Flooding
    - Ex. Sending more than 1000 addresses in addr msg

# Discovering Bitcoin's Network Topology

## Motivation / Attacks

- Who is operating a bitcoin node?
- Link user pseudonyms to IP addresses where the transactions are generated
- Where is the weakest point in the chain to attack?
  - Split the network

# Discovering Bitcoin's Network Topology

## Motivation / Research

- Is the Bitcoin network really decentralized?
  - Super nodes
  - Point of failures
  - Private peering agreement between miners (private relay)
- Metrics should be exposed by the Bitcoin software itself



# Discovering Bitcoin's Network Topology

## Coinscope

- Addrman maintains a timestamp to every known address
- Timestamp of active connections are updated within 20mins
- Timestamp of a new address doesn't change once added to addrman

# Discovering Bitcoin's Network Topology

## Coinscope - Mitigations

The screenshot shows a GitHub pull request interface. At the top, the title is "Reduce fingerprinting through timestamps in 'addr' messages. #5860". Below the title, it says "Merged" and "laanwj merged 1 commit into bitcoin:master from sipa:addrfinger on 17 Mar 2015". The interface includes tabs for "Conversation" (14), "Commits" (1), "Checks" (0), and "Files changed" (2). The conversation section shows two comments: one from "sipa" dated 6 Mar 2015, which is suggested by "@jonasnick", and another from "fanquake" dated 8 Mar 2015. A commit history entry shows "sipa" force-pushing the "addrfinger" branch from commit "d84013a" to "54f1a8b" 7 years ago. On the right side, there are sections for "Reviewers" (No reviews), "Assignees" (No one assigned), and "Labels" (P2P).

# Coinscope

## Mitigation

```
src/addrman.cpp
@@ -252,28 +252,29 @@
252 252 // we just overwrote an entry in vTried; no need to update nTried
253 253 info.finfoTried = true;
254 254 return;
255 255 }
256 256
257 257 void CAddrMan::Good_(const CService& addr, int64_t nTime)
258 258 {
259 259     int nId;
260 260     CAddrInfo* pinfo = Find(addr, &nId);
261 261
262 262     // if not found, bail out
263 263     if (!pinfo)
264 264         return;
265 265
266 266     CAddrInfo& info = *pinfo;
267 267
268 268     // check whether we are talking about the exact same CService (including same port)
269 269     if (info != addr)
270 270         return;
271 271
272 272     // update info
273 273     info.nLastSuccess = nTime;
274 274     info.nLastTry = nTime;
275 275 - info.nTime = nTime;
276 275     info.nAttempts = 0;
277 276 + // nTime is not updated here, to avoid leaking information about
278 277 + // currently-connected peers.
```

# Coinscope

## Mitigation

```
271 271
272 272 // update info
273 273 info.nLastSuccess = nTime;
274 274 info.nLastTry = nTime;
275 - info.nTime = nTime;
276 275 info.nAttempts = 0;
276 + // nTime is not updated here, to avoid leaking information about
277 + // currently-connected peers.
277 278
```

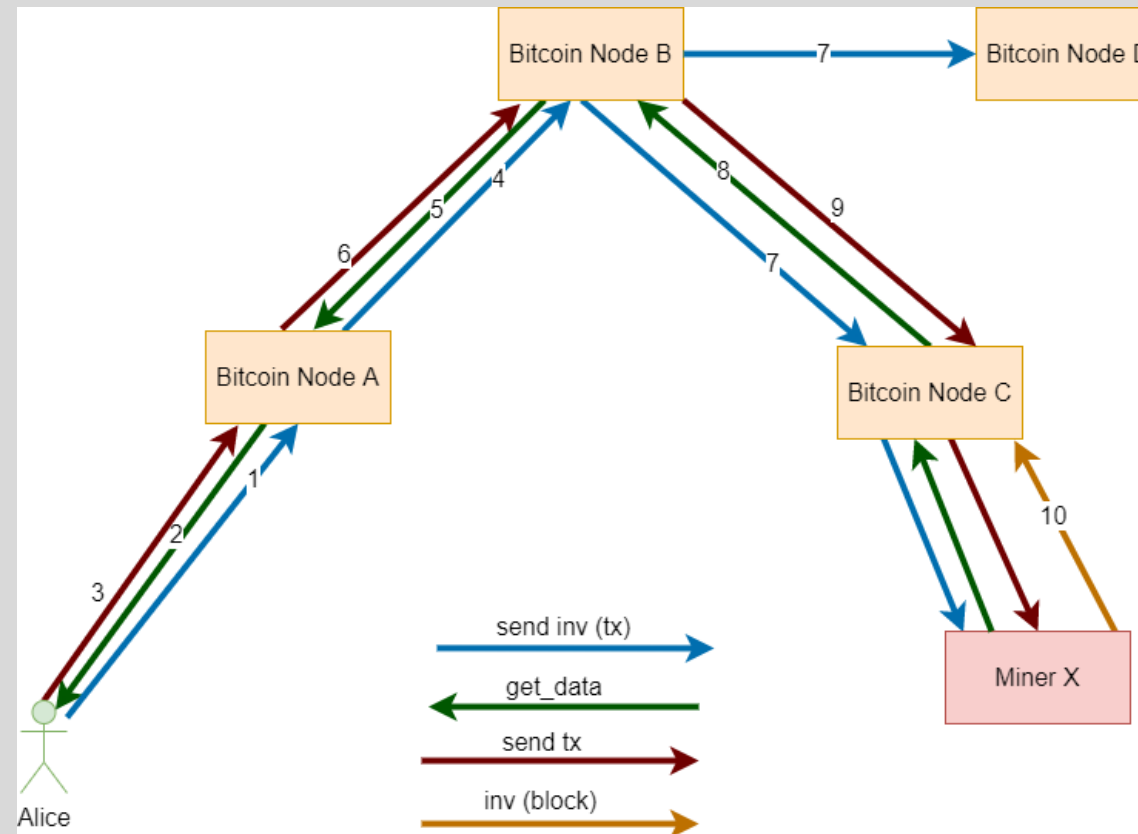
# Discovering Bitcoin's Network Topology

## TxProbe

- **inv** message
  - Used to announce new blocks or transactions
  - Contains only hashes of blocks or transactions
- **get\_data** message
  - Used to request certain block or transaction
  - Uses hashes to request certain item

# Discovering Bitcoin's Network Topology

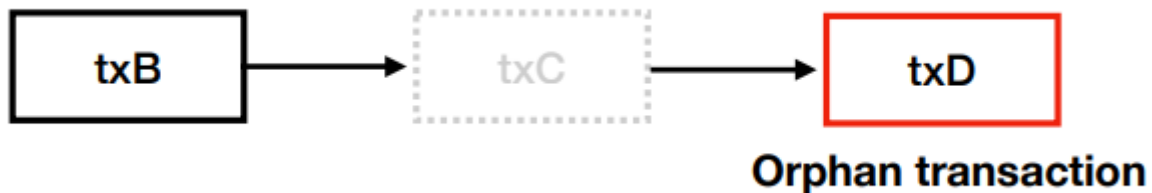
## Relay mechanism



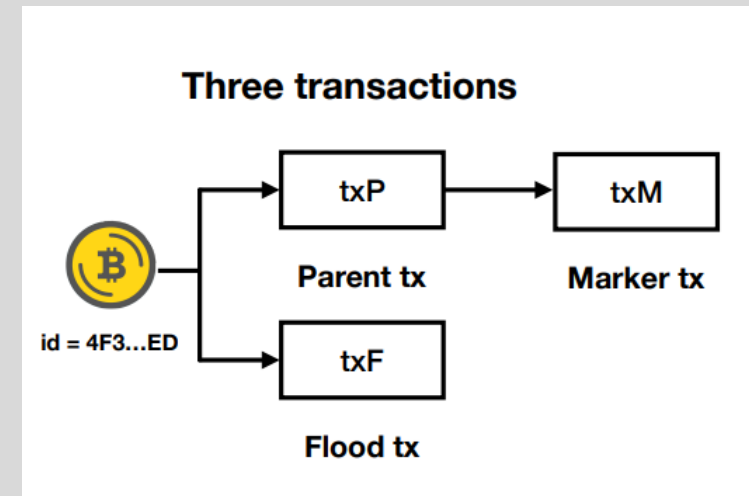
# Discovering Bitcoin's Network Topology

## TxProbe

- Orphans transactions are used to probe connections between nodes



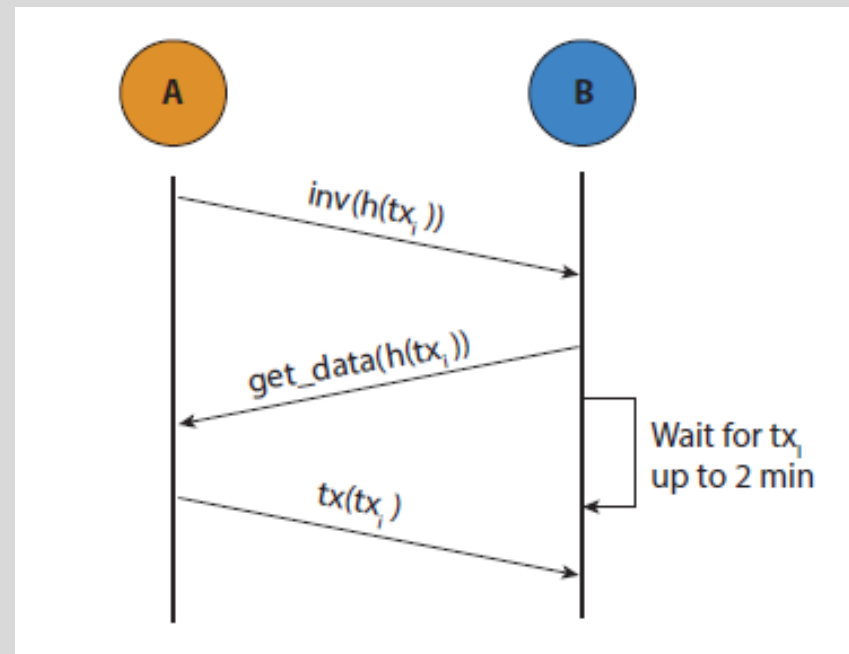
Source: Delgado-Segura, Sergi, et al. "TxProbe: Discovering Bitcoin's Network Topology Using Orphan Transactions." Crossref, <https://srgi.me/resources/slides/CESC19-TxProbe.pdf>.



Source: Delgado-Segura, Sergi, et al. "TxProbe: Discovering Bitcoin's Network Topology Using Orphan Transactions." Crossref, <https://srgi.me/resources/slides/CESC19-TxProbe.pdf>.

# Discovering Bitcoin's Network Topology

## TxProbe



Source: Delgado-Segura, Sergi, et al.

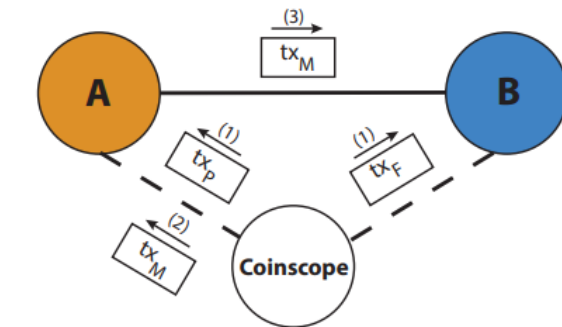
"TxProbe: Discovering Bitcoin's Network Topology Using Orphan Transactions."

Crossref, [https://doi.org/10.1007/978-3-030-32101-7\\_32](https://doi.org/10.1007/978-3-030-32101-7_32).



# Discovering Bitcoin's Network Topology

## TxProbe



A's mempool

tx<sub>P</sub> (1)  
tx<sub>M</sub> (2)

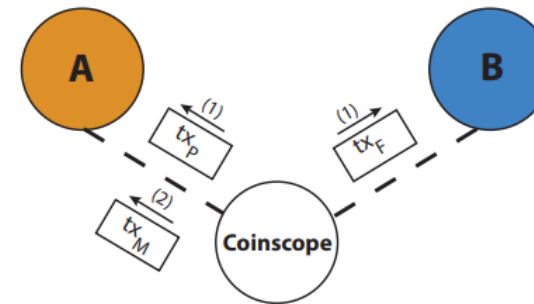
B's mempool

tx<sub>F</sub> (1)

B's MapOrphanTransactions

tx<sub>M</sub> (3)

(a) Basic positive edge inferring technique between two nodes.



A's mempool

tx<sub>P</sub> (1)  
tx<sub>M</sub> (2)

B's mempool

tx<sub>F</sub> (1)

B's MapOrphanTransactions

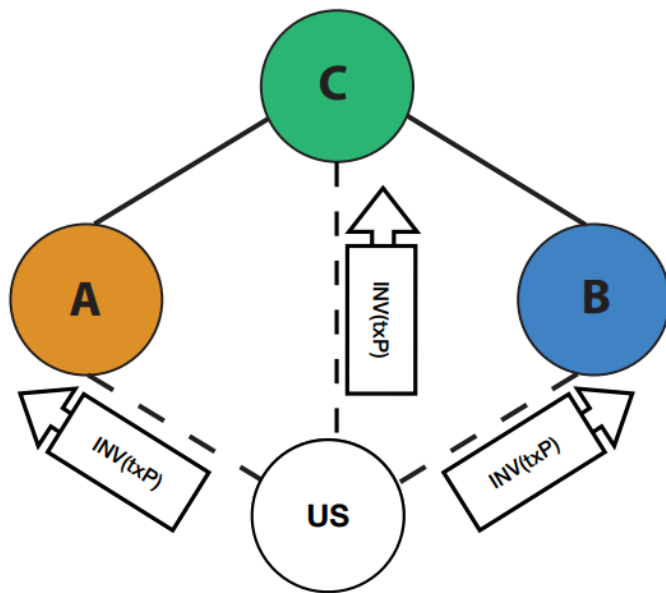
∅

(b) Basic negative edge inferring technique between two nodes.

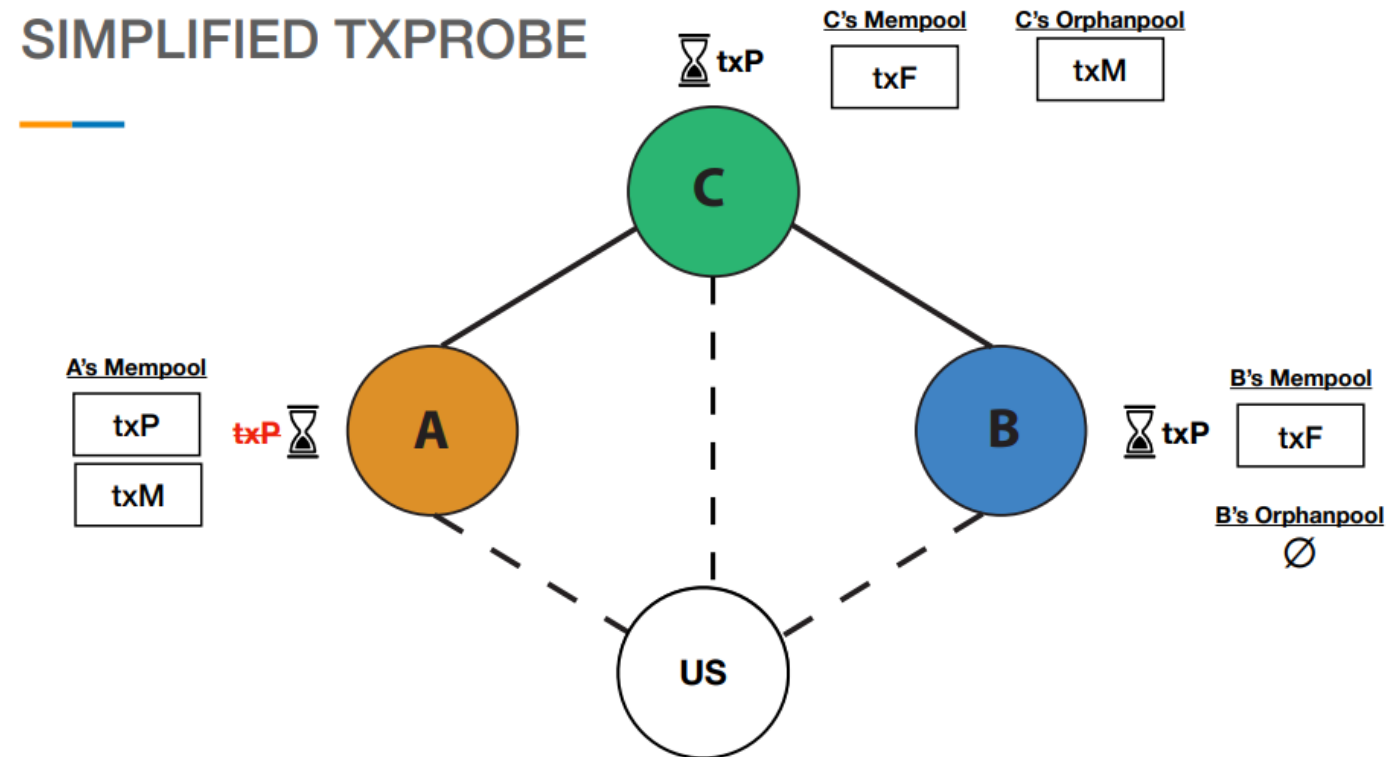
# Discovering Bitcoin's Network Topology

## TxProbe

### INVBLOCKING



### SIMPLIFIED TXPROBE



# TxProbe

## Mitigations

Orphan eviction is not actually random #15121

**Closed** sr-gi opened this issue on 7 Jan 2019 · 2 comments

sr-gi commented on 7 Jan 2019

When orphan transactions are evicted from the MapOrphanTransactions pool they are chosen based on their transaction id.

[bitcoin/src/net\\_processing.cpp](#)  
Lines 785 to 791 in 378fd4a

p2p: Add 2 outbound block-relay-only connections #15759

**Merged** fanquake merged 9 commits into [bitcoin:master](#) from [sdaftuar:2019-03-blocksonly-edges](#) on 7 Sep 2019

Conversation 153 Commits 9 Checks 0 Files changed 6

sdaftuar commented on 5 Apr 2019 · edited

Transaction relay is optimized for a combination of redundancy/robustness as well as bandwidth minimization -- as a result

randomize GETDATA(tx) request order and introduce bias toward outbound #14897

**Merged** sipa merged 1 commit into [bitcoin:master](#) from [naumenkogs:master](#) on 8 Feb 2019

Conversation 115 Commits 1 Checks 0 Files changed 6

naumenkogs commented on 8 Dec 2018 · edited by MarcoFalke

This code makes executing two particular (and potentially other) attacks harder.

Reviewers: jamesob, jonasschnelli

## TxProbe

### Mitigation – Orphan pool

- Bug: Transaction with lowest hash was evicted
  - Not truly random
- Fix: Select a random position in a list of orphaned transactions

## TxProbe

### Mitigation - InvBlock

- Bug: Not accepting any tx from other nodes after sending get\_data to a node
- Fix: Randomized fetch order
  - Outbound connections are preferred over inbound connections

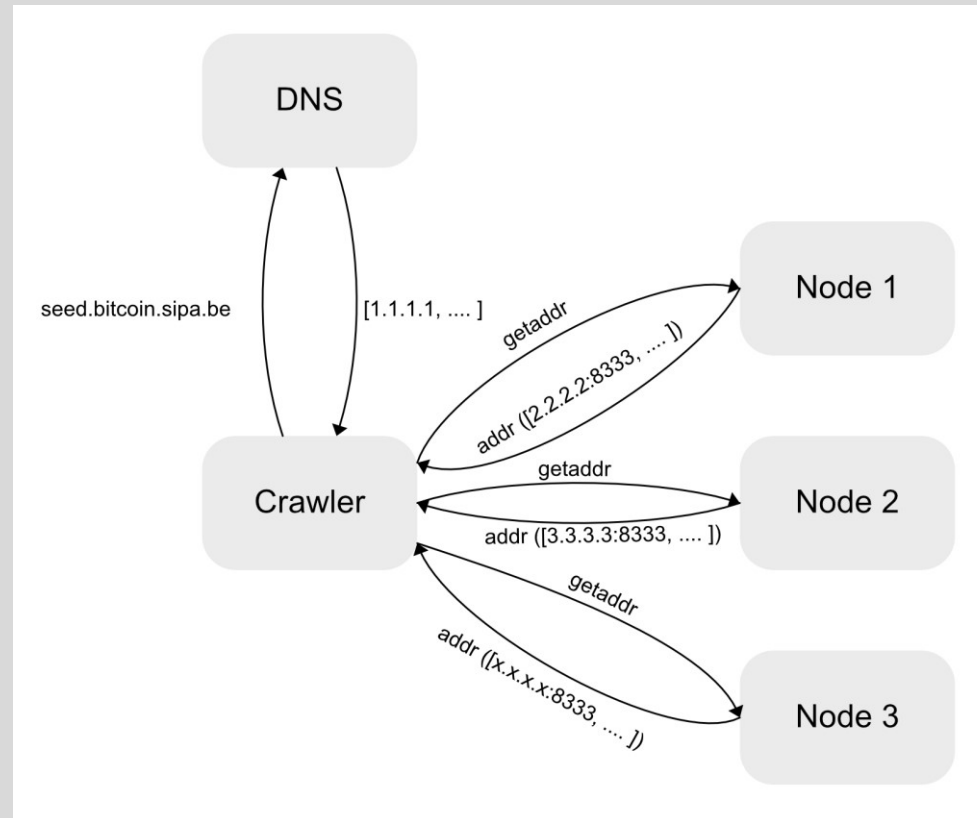
## TxProbe

### Mitigation – Outbound block-only relay

- Bug: Possible to infer the topology graph
- Fix: Additional 2 block-only relay outbound connections
  - Hide 2 additional connections from outside
  - Only block messages are relayed
  - Addr & tx message are not relayed

# Seminar Project

## Crawler



# Seminar Project

## Tools

- Python3
- MaxMind GeoLite2 location database
  - Accuracy 50km
- MaxMind GeoLite2 ASN database
- DNS resolution
- Bitcoin-seeder
  - <https://github.com/sipa/bitcoin-seeder>



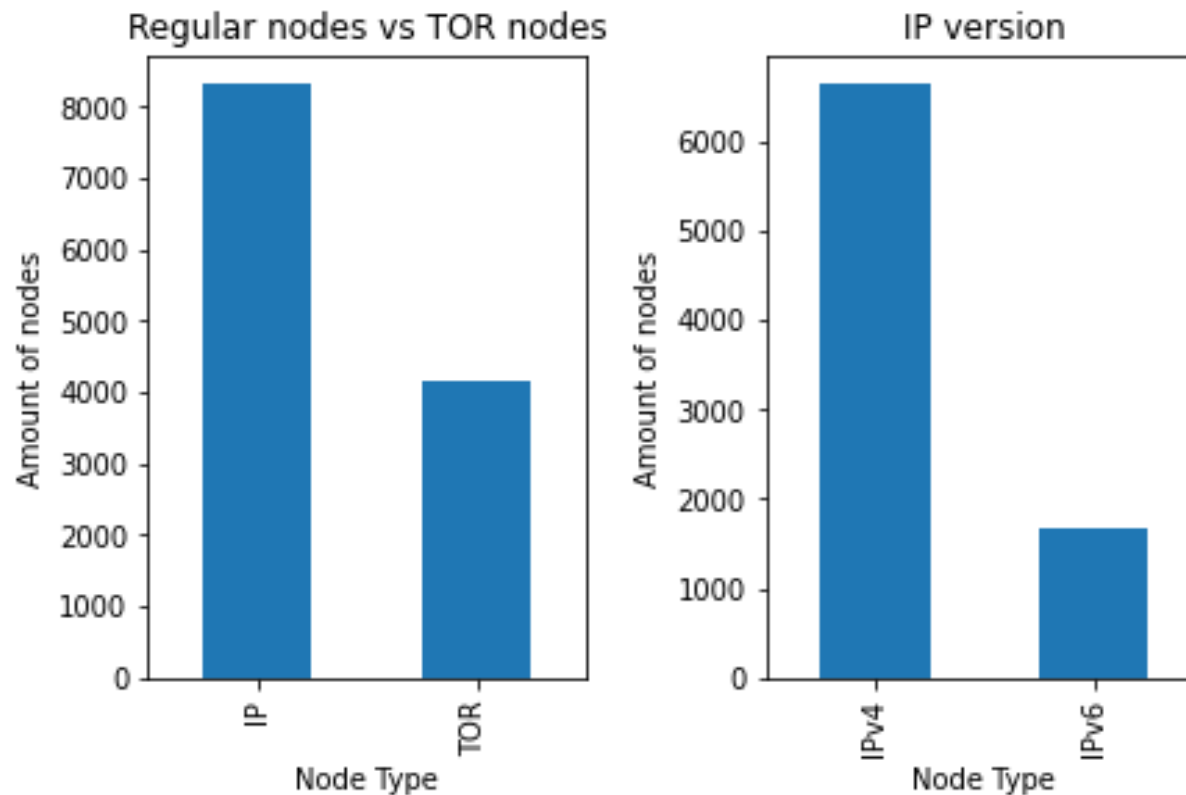
# Seminar Project

## Tools

```
sysop@btc01:~$ ps aux | grep dnsseed
dnsseed    71836  0.5  1.8 1031908 37320 ?        Ssl  Apr03   6:57 /usr/bin/dnsseed
sysop      104961  0.0  0.0   6432   720 pts/0    S+   12:16   0:00 grep --color=auto dnsseed
sysop@btc01:~$ sudo head -n 10 /var/lib/dnsseed/dnsseed.dump
# address                good  lastSuccess    %(2h)   %(8h)   %(1d)   %(7d)   %(30d)  blocks  svcs  version
176.209.115.111:8333      0     1649073958     99.99%  91.46%  55.97%  11.06%  2.70%   730389  00000408 70016 "/Satoshi:22.0.0/"
193.32.127.162:60969      0     1649073934     99.99%  91.46%  55.96%  11.05%  2.70%   730389  00000409 70016 "/Satoshi:22.0.0/"
95.168.168.108:8333       1     1649073909     99.99%  91.45%  55.95%  11.05%  2.70%   730389  00000409 70016 "/Satoshi:22.0.0/"
[2a05:d016:98f:5203:6f3a:54ce:5bd9:2770]:8333  1     1649073947     99.99%  91.45%  55.95%  11.05%  2.70%   730389  0000040d 70015 "/Satoshi:0.20.1/"
162.196.143.112:8333     1     1649073867     99.99%  91.45%  55.94%  11.05%  2.70%   730389  00000409 70016 "/Satoshi:22.0.0/"
5.9.51.253:8333          1     1649073955     99.99%  91.45%  55.94%  11.05%  2.70%   730389  00000409 70015 "/Satoshi:0.20.1/"
34.83.108.62:8333        1     1649073898     99.99%  91.45%  55.94%  11.05%  2.70%   730389  0000040d 70015 "/Satoshi:0.20.1/"
[2a02:c207:0:4971::1]:8333 1     1649073854     99.99%  91.44%  55.93%  11.05%  2.69%   730389  0000040d 70015 "/Satoshi:0.18.0/"
79.46.37.101:8333        1     1649073857     99.99%  91.43%  55.92%  11.04%  2.69%   730389  00000409 70016 "/Satoshi:22.0.0/"
sysop@btc01:~$
```

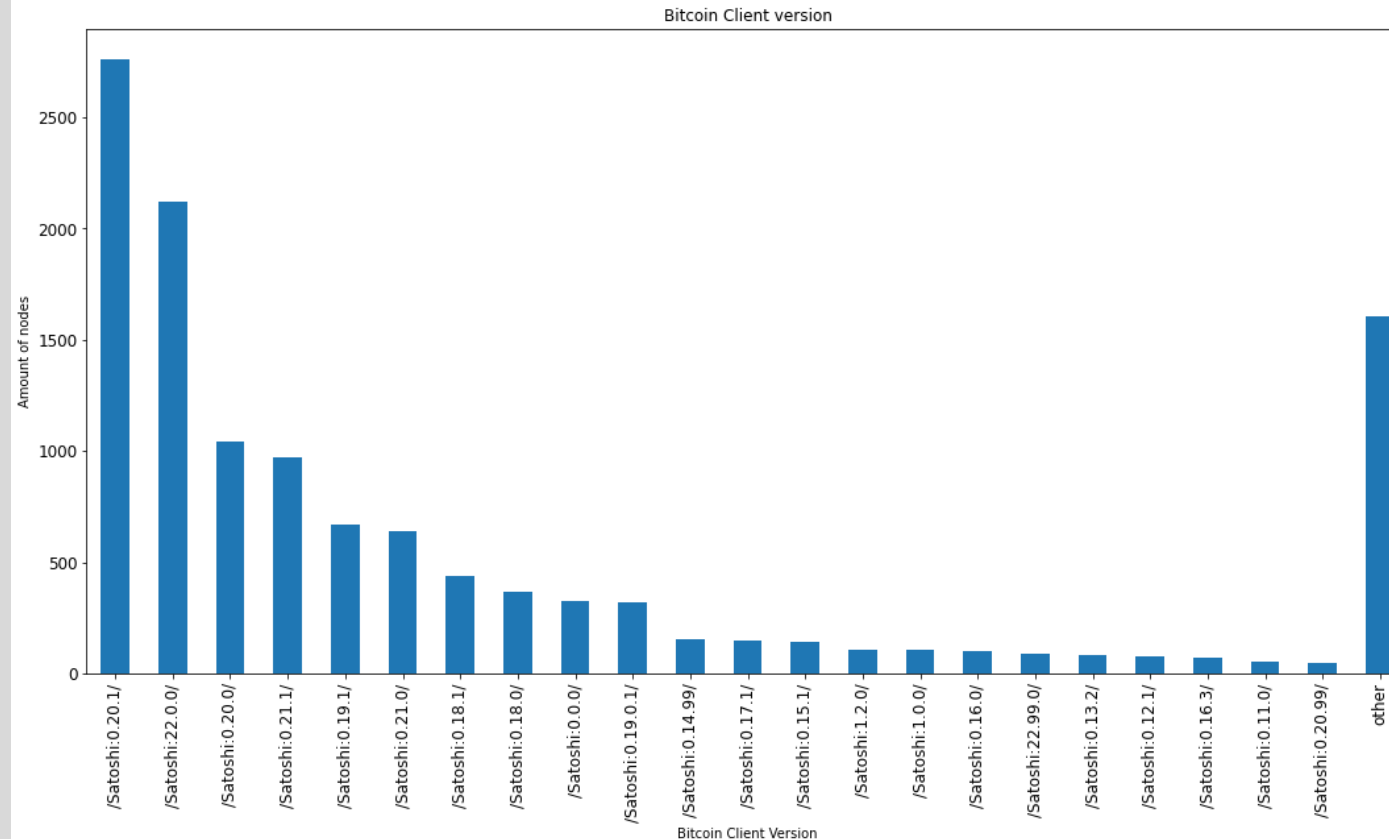
# Seminar Project

## Reachability of Bitcoin nodes



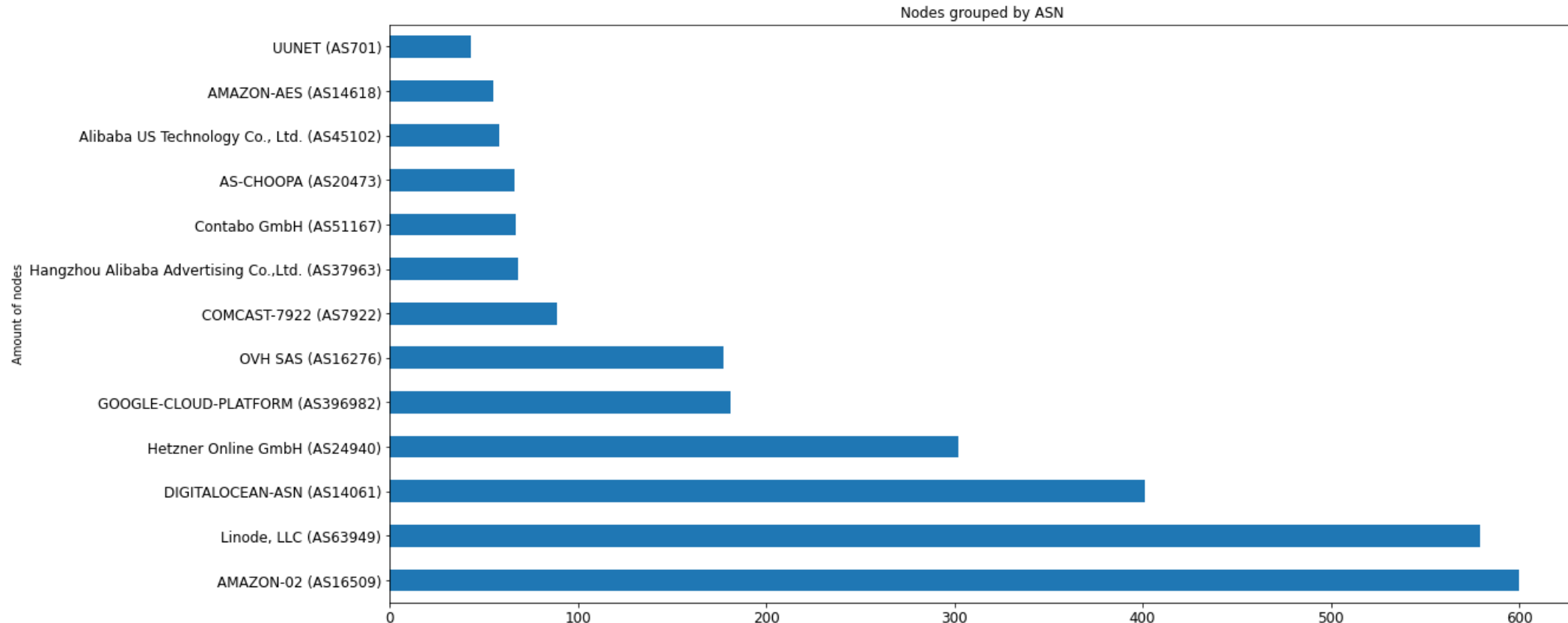
# Seminar Project

## Bitcoin nodes grouped by Version



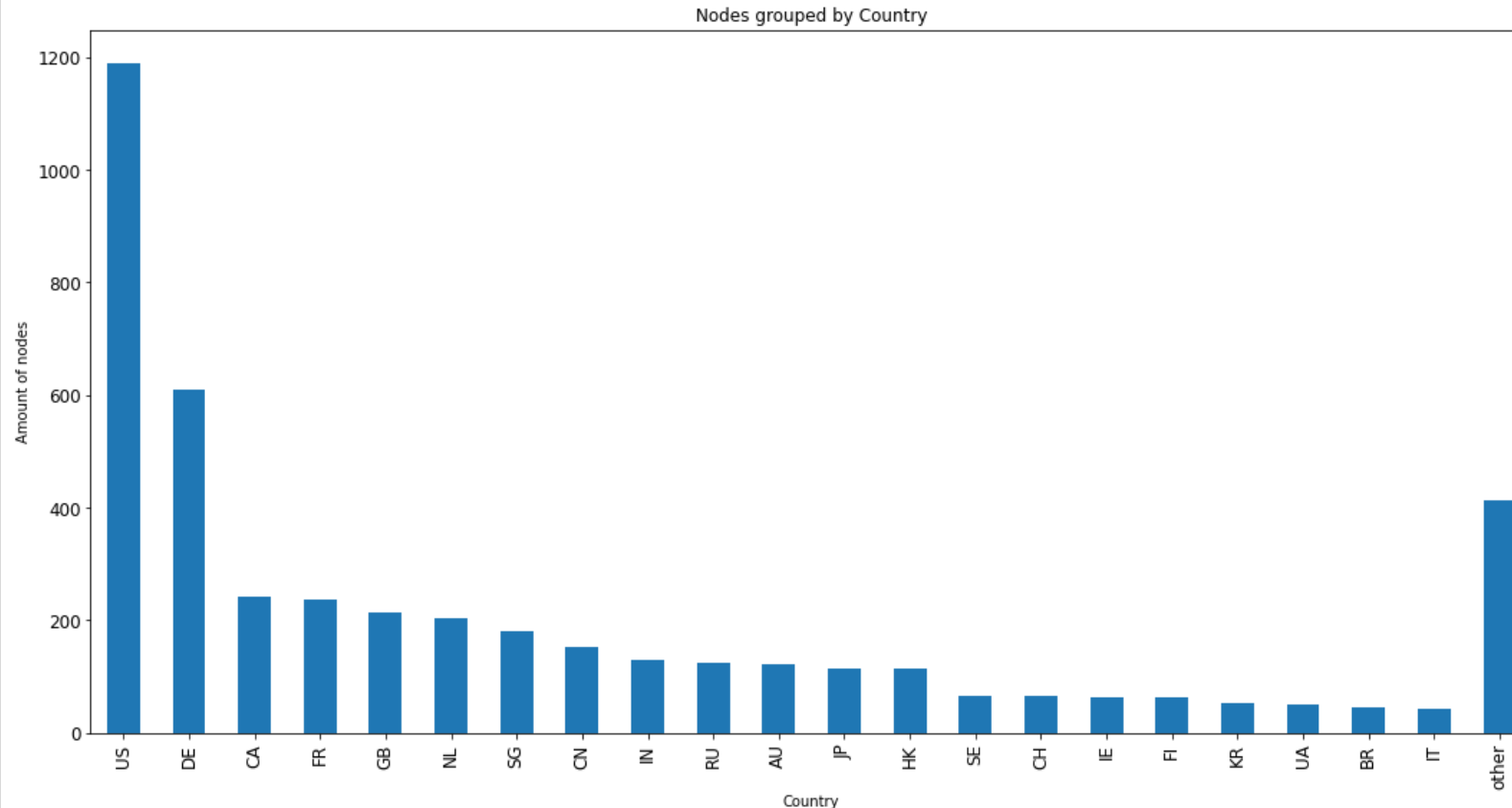
# Seminar Project

## Bitcoin nodes grouped by ASN



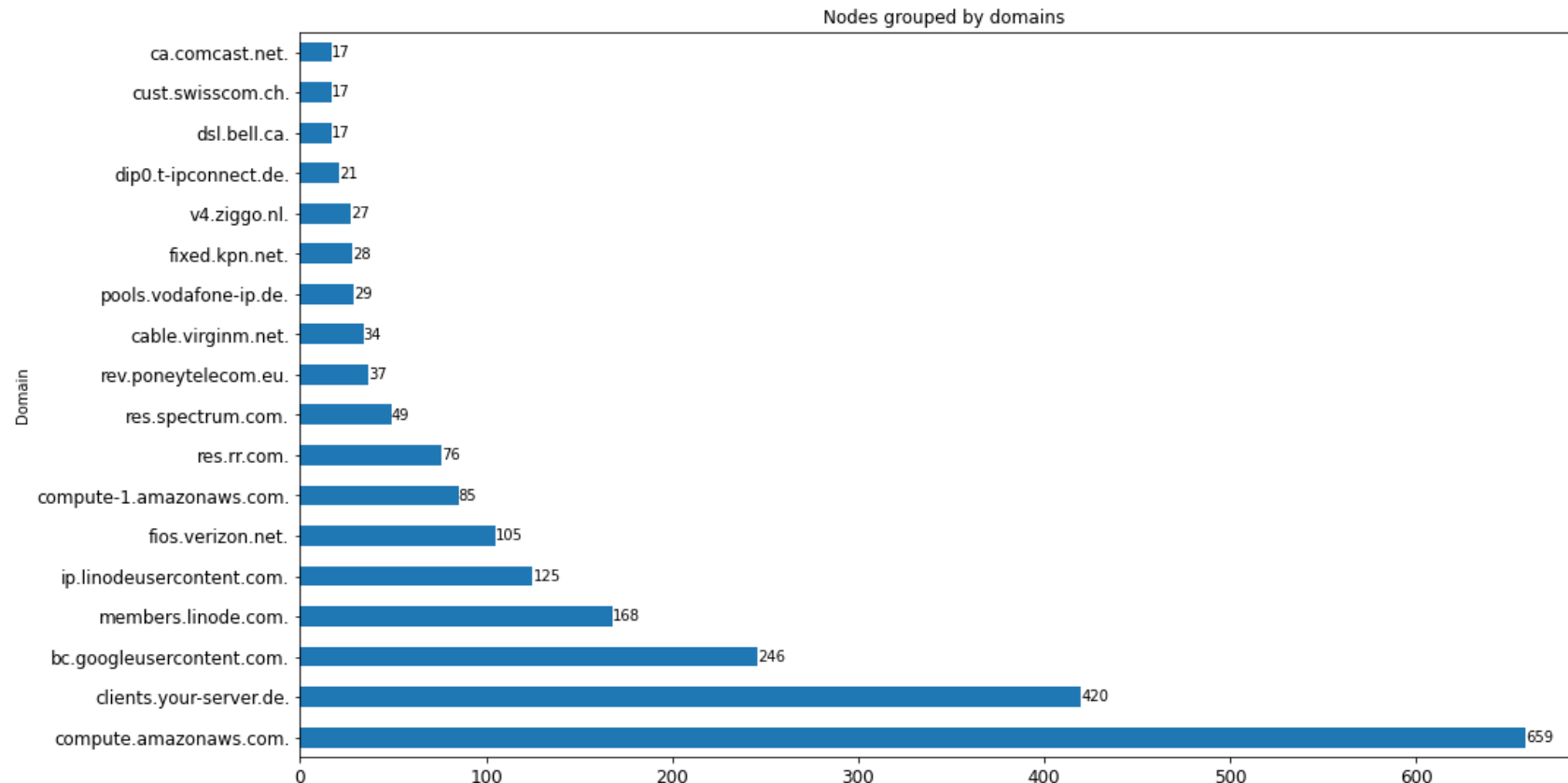
# Seminar Project

## Bitcoin nodes grouped by country



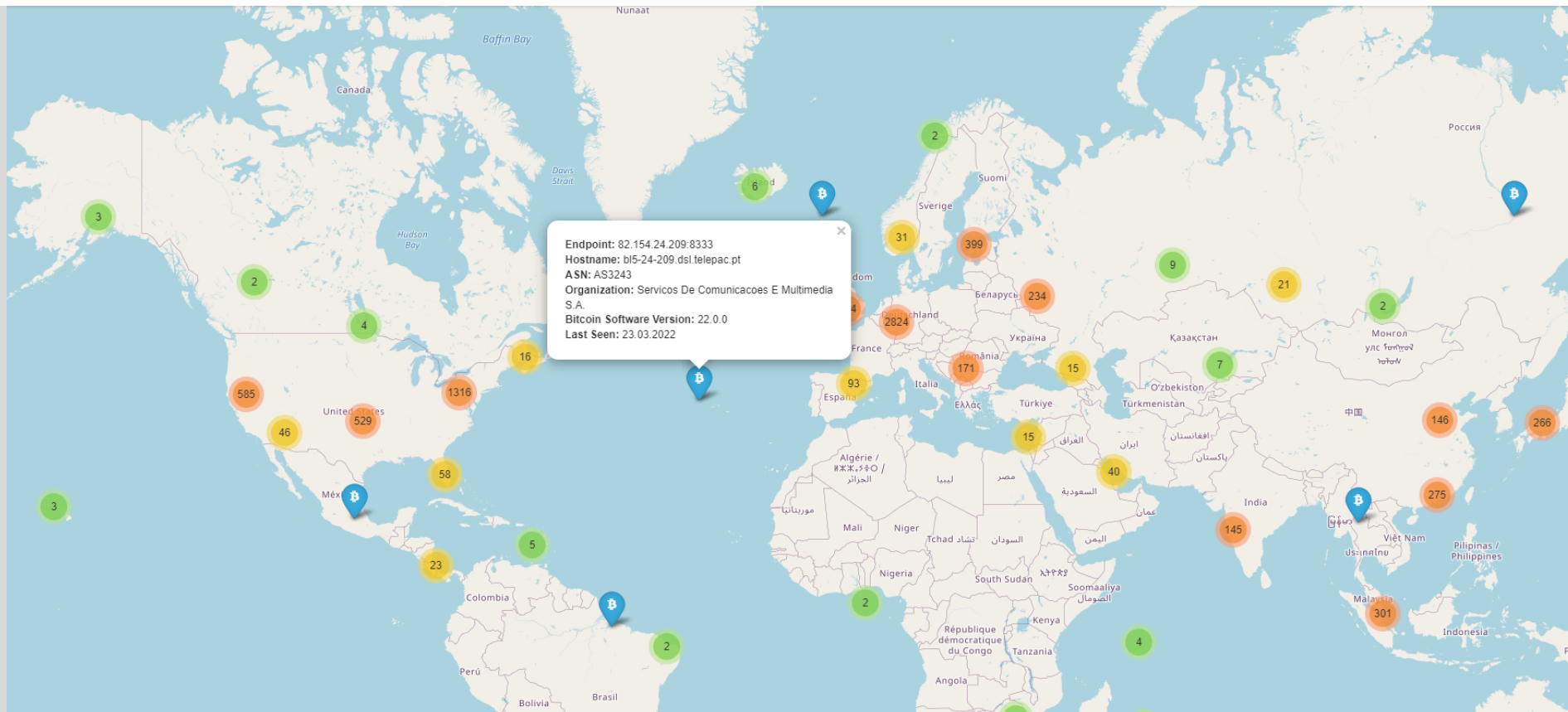
# Seminar Project

## Bitcoin nodes grouped by domain



# Seminar Project

## IPv4 & IPv6 Bitcoin nodes in mainnet



# Thank you

# For your attention

**Chris Rüttimann & Thushjandan**

Bern, 6. April 2022

***u*<sup>b</sup>**

---

<sup>b</sup>  
**UNIVERSITÄT  
BERN**

