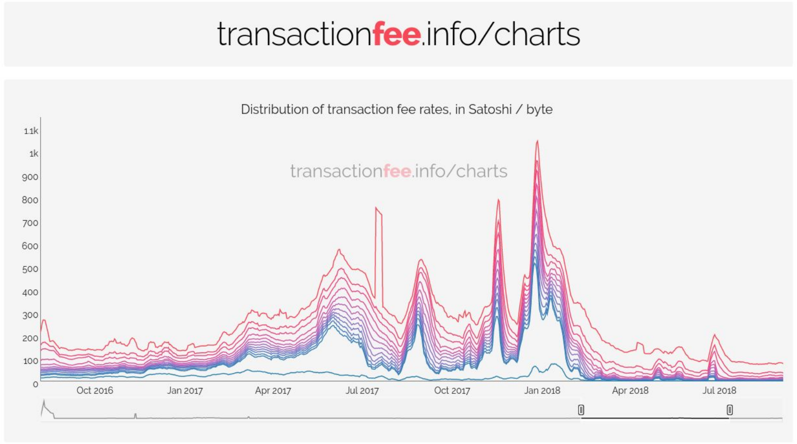
Building mempool.observer - mempool stats and visualizations

BitDevs NYC - Socratic Seminar 93 - 2019/06/13

Who am I?



B1
0C

0xb10c (Timo)

Germany

- In NYC for the first month of the Chaincode Residency
- Working on Bitcoin side projects since spring 2017
  - https://mempool.observer (v1 in 2017)
  - https://transactionfee.info/charts (2018, BitDevs SS79 and SS81)

Motivation for mempool.observer

- Educate new-ish Bitcoin users about the/my mempool
  - Why is my transaction stuck for 20 hours?
  - What can I do now?
- Inform power users about the/my mempool state
  - Is my wallets fee estimator reasonable? (i.e. current mempool)
  - When did the/my mempool clear the last time? (i.e. historical mempool)
- Personal learning and contributing

"*What's going to happen to Bitcoin?*" is the wrong question.
The right question is "*What are you going to contribute?*"

— Greg Maxwell

Live demo

mempool observer

https://mempool.observer

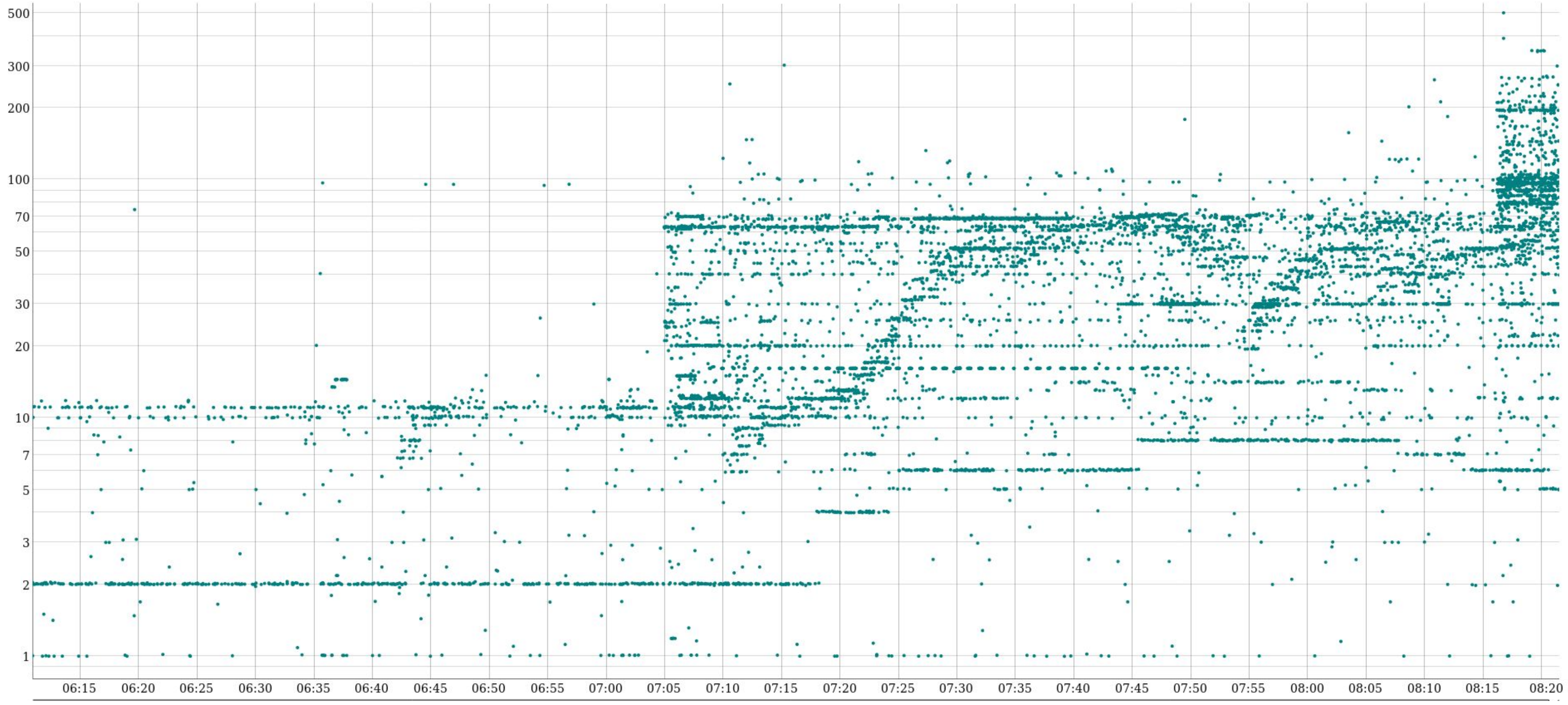
Bitcoin Core mempool interfaces and their challenges

- JSON-RPC: `getrawmempool true`
    - RPC calls can be quite slow for a large mempool: set timeout and fetch frequency accordingly
    - PR #14984 by promag: `rpc: Speedup getrawmempool when verbose=true`
        - $O(n^2) \rightarrow O(n)$ (~30% speedup for me at 57k tx)
        - Merged May 15th and tagged for v0.19.0
- REST: `/rest/mempool/contents.json`
    - Faster than JSON-RPC
    - I did some profiling on `MempoolToJSON()` but you need a full mempool for that...
        - How do you realistically fake your mempool for profiling? (ask this in Q&A)
        - Reading the HTTP response on my side somehow takes a lot of time...
- Reading mempool.dat after `savemempool`
    - By far the fastest
    - Does not include fees (that could easily be patched, but I'll try not to for now)

Current TODOs

- Notify user on transaction confirmation (while preserving privacy) **new feature**
  - Receive `rawblock` over ZMQ
  - Broadcast block txids over Websocket
  - Web notification or sound upon confirmation (i.e. `user txid == txid in block`)
- Properly handle transaction packages **bug**
- Live chart of incoming transaction **new feature**
  - Live plain tx per second chart… (boring)
  - Extended with data from `rawtx` over ZMQ→ `getmempoolentry rawtx[‘txid’]`
    - Scatterplot `entrytime` x `feerate`
    - Extend with: size, fee, tx type (e.g. SegWit, RBF, multisig...) ...

Live transactions: `entrytime` x `feerate`



Ideas for the future of mempool.observer

- Feerate converter between `sat/byte`, `BTC/kB` (Bitcoin Core), `sat/kB` and `sat/kw` (c-lightning, lnd)
- Rate feerate estimators: comparing `estimate` vs `block feerates`
- Somehow integrate Kalle Alm's `Mempool File Format`
- ...

Thank you BitDevs NYC!
Thank you Chaincode for flying me out for the Residency!

mempool.observer
github.com/0xB10C/memo

You can vote for features here and now
https://simplevote.tk/#/poll/J62y

@0xb10c https://b10c.me 0xb10c@gmail.com

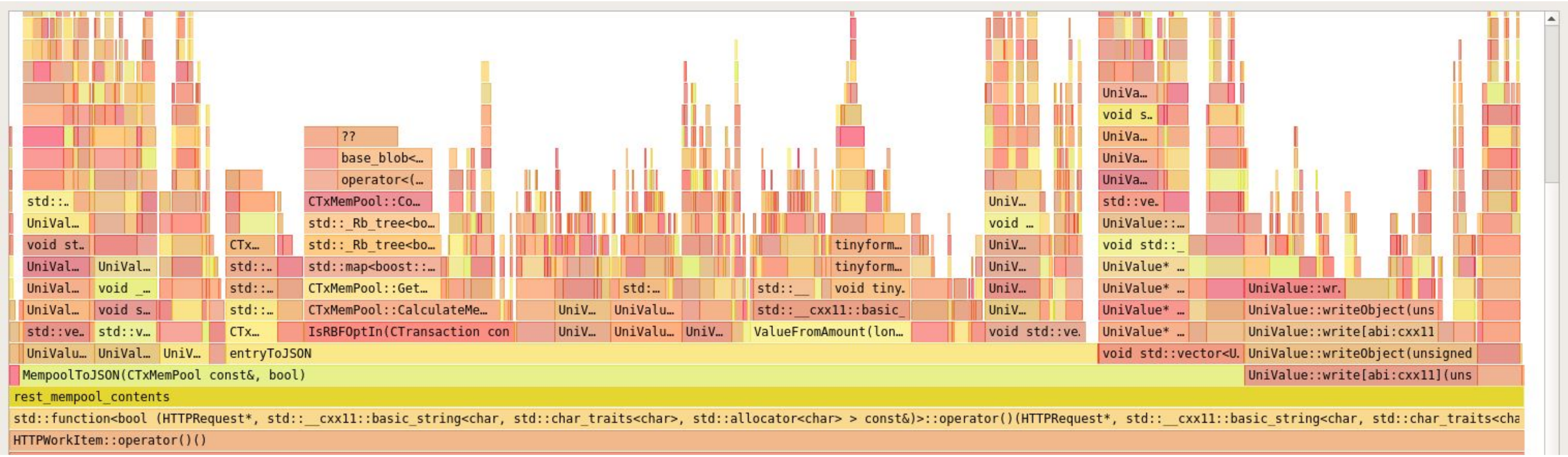
Filling up my mempool for profiling and benchmarking

```
1     nBlocks = 25
2     get all raw tx from the last <nBlocks> by iterating over each
3     invalidate block at <currentHeight> - <nBlocks> (keep that block hash)
4     broadcast the raw tx from the previous <nBlocks> (line 2)

      -- do profiling and stuff here --

5     reconsider block at <currentHeight> - <nBlocks> (with the block hash from line 3)
```

- Do this only on your local dev setup
- Fills mempool with transactions from the last `nBlocks` blocks
  - Fastest method I've tried (yet)
  - Mempool closer to reality than 100k*[vin: 1, vout: 2] on regtest
  - Transactions with e.g. locktime fail

    ⇒ Transactions spending these fail too → not a 100% realistic mempool

Results profiling REST: `/rest/mempool/contents.json`



with ~75k tx