What should Bitcoin Core be?

And what is it right now?

Engineering resources are scarce

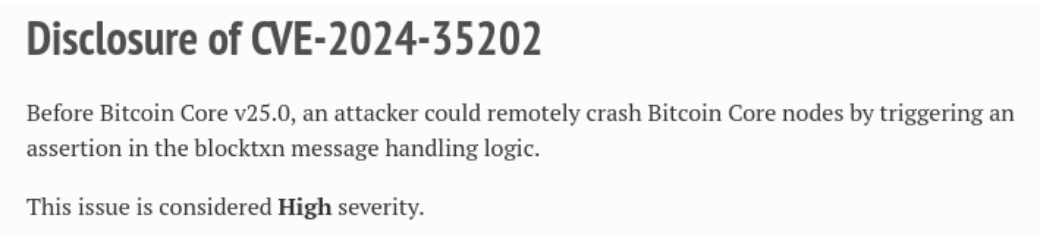
- Complex project with **several areas**
- Head count is not all that matters
- Experienced contributors **reducing involvement**

Code does not come for free

- Opportunity **cost**
  - Write
  - Review
- Maintenance **cost**
  - Update
  - Fix
  - Interaction with future developments

Code does not come for free

- **Opportunity** cost examples:
  - Important projects make little **progress** (package relay, kernel, multiprocess)
  - Important areas get **less attention**



Disclosure of CVE-2024-35202

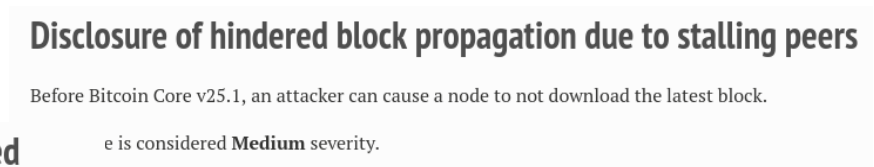
Before Bitcoin Core v25.0, an attacker could remotely crash Bitcoin Core nodes by triggering an assertion in the blocktxn message handling logic.

This issue is considered **High** severity.

Disclosure of DoS due to inv-to-send sets growing too large

Before Bitcoin Core v25.0, the per-peer `m_tx_inventory_to_send` sets could grow too large to a point where sorting these sets when constructing inventory messages would affect the node's ability to communicate with its peers. Network conditions in early May 2023 triggered this DoS and affected block and transaction propagation.

This issue is considered **Medium** severity.



Disclosure of hindered block propagation due to stalling peers

Before Bitcoin Core v25.1, an attacker can cause a node to not download the latest block.

e is considered **Medium** severity.

Disclosure of hindered block propagation due to mutated blocks

Before Bitcoin Core v25.0, a peer sending mutated blocks could clear the download state of other peers that also announced the block to us, which would hinder block propagation.

This issue is considered **Medium** severity.

Code does not come for free

- **Maintenance** cost examples:
  - Maintaining the test infrastructure (CI, OSS FUZZ, personal setups..)
  - Updating the test infrastructure (compatibility issues)
  - Updating the toolchain
  - Modernizing the codebase
  - General software decay (features may have to be adapted or fixed)
  - Future work interacting with this code directly or not (hard to predict or even measure such cost in advance)
- **Dispersed costs**, concentrated benefits

Constant resources

Increasing project size

The Trade-off

Keep going

- Diluted attention
- Reduced overall **quality**

Reduce size

- Requires **scope definition**

Scope definition

- What should Bitcoin Core be?
  - What do **users** want/need from Core?
  - How to **prioritize** these wants?
- Who are the users?
  - Bitcoin Core users
  - Bitcoin users

Scope definition

- What should Bitcoin Core be?
  - **A robust backbone for the Bitcoin network**

Additional advantages of a scope

- Sets clear **expectations**
  - Internally *and* externally
  - Give more support to NACK stuff
  - Sunk cost / cut losses situations less likely
  - New contributors
  - Better able to set goals

Fine, now what?

- Delete non-node components?
  - **Break** their users / contributors
  - Relieve **burden** on everyone else
- Keep non-node components?
  - Don’t **break** their users / contributors
  - Keep **burden** on everyone else still

Project split

Node project

Develops:

`bitcoin-node`

Releases:

`bitcoin-node`

Wallet project

Develops:

`bitcoin-wallet`

Releases:

`bitcoin-node`
`bitcoin-wallet`

`bitcoind`

GUI project

Develops:

`bitcoin-gui`

Releases:

`bitcoin-node`
`bitcoin-wallet`
`bitcoin-gui`

`bitcoin-qt`

Project split

- **Relieve burden** on the node project
- **Don’t break** users/devs of non-node components
- No free lunch, but **opportunity** for non-node components

Conclusion

- Code is cost. Opportunity **and** maintenance.
- Scope: necessary but also useful signal.
- Bitcoin Core scope = maintaining Bitcoin.
- Project split: realistic route, limit costs on node, but also opportunity for other components.
- Additional social reasons to re-focus.