Blockchain code development pitfalls

Stories from TrustChain development

Vadim Bulavintsev
*Distributed Systems*
2019
#1 unexpected blocks diagram

**Theory**

Diagram from the original TrustChain paper

Each block has exactly 2 parents and 2 children

**Practice**

Block diagram from an early version of TrustChain

Notice something wrong?
#1 unexpected blocks diagram

**Theory**

Diagram from the original TrustChain paper

```
Hash A
  ↓  ↓
  B   C
  ↓  ↓
Hash D
```

Each block has exactly 2 parents and 2 children

**Practice**

Block diagram from an early version of TrustChain

```
Blocks with more then 2 successors should never exist!
```

```
Hash A
  ↓  ↓
  B   C
  ↓  ↓
Hash D
```
#1 take-home message

- Humans think in visual images, not lines of code. Make proper use of your visual cortex.

Visualize everything from the beginning.
#2 excessive block broadcasts

**Expected**
*(and unit tested!)*

Packet time to live 1.
Broadcast to 25 peers.

**Reality**

Packet time to live 2.
Broadcast to 1110 peers.
#2 take-home message

- There were 2 errors: in the main code and in the test.
- The risk of a double error is higher in a complex system. Double-check everything.

Tests can have errors too.
Cross-check your tests.
#3 contagious crash

• One day, **all** app instances begin to crash 1 minute after connecting to the network.

• The crash spreads **like a disease** among peers.

• Who could have created this devious «message of death»? **Is this an attack?**
#3 take-home message

- A broken block was put on the TrustChain. Peers trying to re-send it **could not handle the value**.
- You guessed it: no attack, just a student playing around ;-)

Sanitize incoming values **thoroughly**.
Libtorrent developers were using this chart to detect anomalies in network packet flow for 5 years.

Same chart, after a certain single-line bug was fixed. The bug was there from the beginning.
#4 take-home message

- Never take anything for granted.

Use scientific thinking: try to falsify your basic assumptions.
Recap

1. Visualize everything from the beginning.
2. Cross-check your tests.
3. Thoroughly sanitize incoming data.
4. Always try to prove yourself wrong first.
Blockchain code development pitfalls

Stories from TrustChain development

Vadim Bulavintsev

*Distributed Systems*

2019