Public blockchains

What could possibly go wrong?

Rhian Lewis @rhian_is
About me

- Test automation engineer
- Co-developer of Count My Crypto
- Founder of London Women in Bitcoin
- Teacher on the B9Lab.com Ethereum QA Engineer course
- Writer about blockchain at medium.com/@rhian_is
Themes in this talk

• Why should you be concerned about this technology

• How blockchains work

• Differences between private and public blockchains

• Examples of protocols

• How to test and testing challenges

• Examples of vulnerabilities and how to mitigate

• Tools and tips
November 6, 2017

devops199 commented 2 days ago • edited

I accidentally killed it.

https://etherscan.io/address/0x863df6bfa4469f3ead0be8f9f2aae51c91a907b4
Imagine this

- Your code is on thousands of computers over the world
- Hundreds of millions of other people’s money is locked up in their accounts
- YOU CANNOT REDEPLOY!
- THERE IS NO FIX FOR THIS!
- Everyone can see the issue that caused it
To be continued....
Why care about blockchains?
Blockchain definition: 1

- Data structure spread across many nodes
- If public, anyone can download and run the software and participate in maintaining the record
- If private, runs on a limited number of nodes to which are controlled and agreed
Blockchain definition: 2

- Immutable data structure because all transactions are bundled into blocks which are cryptographically linked together from the beginning of time
- Single source of truth
- Transparent and decentralised, with no down time
HOW THE BLOCKCHAIN WORKS

Transaction X; data of any length
Hash value #X; unique value of fixed length

Transaction A
Hash value #A

Transaction B
Hash value #B

Transaction C
Hash value #C

Transaction D
Hash value #D

MERKLE TREE

Hash value #AB
Hash value #CD

Combined hash value #ABCD
Hash of Block 49
Timestamp
Nonce

BLOCK 48
BLOCK 49
BLOCK 50

Reproduction of an original figure in “The Great Chain of Being Sure About Things” by the Economist
BLOCKCHAINS: NOT JUST FOR CRYPTOCURRENCIES
Examples of protocols

• Bitcoin - Proof of Work / C++ codebase / clients in many languages. No specific smart contract functionality

• Ethereum - Proof of Work, moving to Proof of Stake / EVM has four main client implementations / smart contracts written in Solidity (some similarities to JavaScript)

• Hyperledger - open source blockchains and tools / Hyperledger Fabric chaincode can be written in Go or JavaScript

• EOS - delegated proof of stake / smart contracts written in C++, compiled to Web Assembly
Difference between public and private chains

• Need permission to join a private chain

• Transactions are validated on public chains by members of the public who are rewarded for their efforts

• Public chains are more transparent

• Private chains have a purpose but cannot solve the trust issue

• Hybrids where private Proof of Authority chains link to a larger public chain
<table>
<thead>
<tr>
<th>Public</th>
<th>Consortium</th>
<th>Private</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anyone can join</td>
<td>Permissioned</td>
<td>Permissioned</td>
</tr>
<tr>
<td>Open source</td>
<td>Open source or proprietary</td>
<td>Open source or proprietary</td>
</tr>
<tr>
<td>Thousands or even millions of participants</td>
<td>Limited number of participants</td>
<td>Limited number of participants</td>
</tr>
<tr>
<td>Likely to have a token (currency)</td>
<td>Unlikely to have a token (currency)</td>
<td>Unlikely to have a token (currency)</td>
</tr>
<tr>
<td>Governance by consensus</td>
<td>Equal weight to participants</td>
<td>Owner can set the rules</td>
</tr>
</tbody>
</table>
Mock blockchains: 1

This is an example of a virtual Ethereum node, using a tool called Ganache
Mock blockchains: 2

You can interact with Ganache via the user interface (previous slide) or via the command line
Transaction Fees

• Not an issue for private chains

• Public chains like Bitcoin and Ethereum charge a transaction fee, which fluctuates

• Testing that the business model functions with fees at different levels is crucial

• For example, micropayments do not make sense if you pay $1 for every transaction
Testing transaction fees

Gas usage: 5752954
Vulnerability: Re-entrance

pragma solidity ^0.4.8;
contract HoneyPot {
  mapping (address => uint) public balances;
  function HoneyPot() payable {
    put();
  }
  function put() payable {
    balances[msg.sender] = msg.value;
  }
  function get() {
    if (!msg.sender.call.value(balances[msg.sender])()) {
      throw;
    }
    balances[msg.sender] = 0;
  }
  function() {
    throw;
  }
}
Vulnerability: Ownership

// constructor is given number of sigs required to do protected "onlymanyowners" transactions
// as well as the selection of addresses capable of confirming them.
function multiowned(address[] _owners, uint _required) {
    m_numOwners = _owners.length + 1;
    m_owners[1] = uint(msg.sender);
    m_ownerIndex[uint(msg.sender)] = 1;
    for (uint i = 0; i < _owners.length; ++i) {
        m_owners[2 + i] = uint(_owners[i]);
        m_ownerIndex[uint(_owners[i])] = 2 + i;
    }
    m_required = _required;
}
Vulnerability: Initialisation

- A smart contract that generates addresses for many users needs to let the blockchain know about these addresses.
- If you display the address before the blockchain transaction has been mined, there is a risk that a user might send money to it.
- **DISASTER!**
- If a user tries to send money to a non-existent address, the cash will be lost for ever.
Not just contracts

- Decentralised applications are more than just a blockchain

- Focus on smart contracts can mean other vulnerabilities are neglected

- Augur framejacking vulnerability

- Nano hack where checks performed on client side only - users could run JavaScript locally
Questions Mount Over $170 Million
BitGrail ‘Hack’
Performance and predictability

The y axis shows transaction confirmation times on the Ethereum blockchain, in minutes.
Automation and Tools

- Two useful tools for Ethereum: Truffle framework and Open Zeppelin libraries
- Truffle gives inbuilt test framework and a mock local blockchain
- Can be difficult to automate tests on public testnets because of latency and need to acquire test currencies
- Most people run tests against local nodes only
Truffle is easy to use!

```bash
Zev:~ Rhian$ mkdir truffle-demo
Zev:~ Rhian$ cd truffle-demo
Zev:truffle-demo Rhian$ truffle init
Downloading...
Unpacking...
Setting up...
Unbox successful. Sweet!

Commands:

  Compile: truffle compile
  Migrate: truffle migrate
  Test contracts: truffle test
Zev:truffle-demo Rhian$
```
Bug Bounties

• The opportunity to hone your testing skills
• The kudos of being able to add your discoveries to your resumé
• The chance to earn Ether or other tokens

• https://bounty.ethereum.org/
• https://hackenproof.com/
Thank you!

- In this talk we have learned:
  - Why blockchains are powerful
  - Why you should consider using a public blockchain
  - Why public blockchains are dangerous
  - What you can do to mitigate this by testing
  - How you can get involved