Blockchain: Your Questions. Our Answers.

Eliezer Kanal & Gabriel Somlo
Previous models of computing

**Data Storage:** Database

**Program Execution:** Local
Blockchain

Data Storage: 
Blockchain

Program Execution: 
Blockchain
Blockchain Properties

Data on the chain cannot be removed
Identity fundamentally linked to activity
Easily auditable
Mediates untrusted party interactions
Classic Currency: Store of Value

• A $100 bill “stores” a $100 value
• My checking account “stores” a $148.23 balance
• If I pay Adam $48.23, there’s an atomic transaction:

begin atomic

Gabriel.Checking == $48.23;
Adam.Checking += $48.23;

end atomic
Cryptocoins: IOUs

- Gabriel owes Adam $48.23
- Peter owes Adam $100.00
  - Therefore, Adam “has” $148.23
    - Assuming IOUs collected instantly, on demand!

- To pay for something, Adam must:
  - Collect (some of) his IOUs
  - Issue a fresh IOU to the payee/merchant

- IOUs (a.k.a. Transactions) passed around by nodes of a distributed, P2P network
## Transactions

<table>
<thead>
<tr>
<th>Transaction</th>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>...</td>
<td>Adam's PubKey $48.23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gabriel's PrivKey Signature</td>
</tr>
<tr>
<td>Transaction</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>Adam's PubKey $100.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Peter's PrivKey Signature</td>
</tr>
</tbody>
</table>
Transactions

Transaction in
... Adam's PubKey $48.23
Gabriel's PrivKey Signature

Transaction in
... Adam's PubKey $100.00
Peter's PrivKey Signature

Transaction out

Transaction in
Merchant PubKey $125.00
Adam's PubKey $23.00
Transaction out
Adam's PrivKey Signature
Transactions

Transaction in  
Transaction out

Adam's PubKey $48.23
Gabriel's PrivKey Signature

Transaction in  
Transaction out

Merchant PubKey $125.00
Adam's PubKey $23.00

Transaction in  
Transaction out

...  
...  
...  
...  

Peter's PrivKey Signature

Transaction in  
Transaction out

...  
...  
...  
...  

Merchant PrivKey Signature

[DOUBLE CREDIT STATEMENT A] Approved for public release and unlimited distribution.
Transactions: Identity of Parties

- **Transaction 1:**
  - **In:**...
  - **PubKey:** Adam's PubKey
  - **Amount:** $48.23
  - **Signature:** Gabriel's PrivKey

- **Transaction 2:**
  - **In:**...
  - **PubKey:** Adam's PubKey
  - **Amount:** $100.00
  - **Signature:** Peter's PrivKey

- **Transaction 3:**
  - **In:**...
  - **PubKey:** Adam's PubKey
  - **Amount:** $23.00
  - **Signature:**...
Transactions: No Overspending!

\[ \sum \text{in} \geq \sum \text{out} \]
Transactions: No Overspending!

\[
\Sigma_{in} \geq \Sigma_{out}
\]

\[TF = \Sigma_{in} - \Sigma_{out} \geq 0\]  
(Transaction Fee)
Transactions: No Double-Spending!

Transaction in

Transaction out

Adam's PubKey $48.23

Gabriel's PrivKey Signature

Transaction in

Transaction out

Adam's PubKey $100.00

Peter's PrivKey Signature

Transaction in

Transaction out

Merchant PubKey $125.00

Adam's PubKey $23.00

Adam's PrivKey Signature

Transaction in

Transaction out

Gabriel's PubKey $100.00

Adam's PrivKey Signature

Merchant PrivKey Signature
Ledger

- DAG of all transactions ever issued
  - Append-only data structure
- Every peer node maintains a copy
Ledger

- Existing *(confirmed)* transactions on HDD
- New *(pending)* transactions in Memory Pool
  - Must be valid w.r.t. existing state to be confirmed
Transaction Blocks

- Confirmed transactions grouped in *blocks*
Transaction Blocks

- Confirmed transactions grouped in *blocks*
- Peers (*miners*) **compete** to create newest block
  - Containing newly validated (confirmed) transactions
Coinbase Transactions

- Compensate miners for “community service” work
  - i.e., confirming users’ pending transactions
- Reward (freshly “minted” money)
  - Also transaction fees from each confirmed transaction

\[
\sum \text{out} = \text{Reward} + TF + TF + TF + \ldots + TF
\]
Block Header
Block Header

- **Merkle Tree** root of transaction hashes
  - Uniquely identify transactions included in block
Block Header

- Timestamp of block creation
  - Monotonically increasing
Block Header

- Hash of previous block header
  - Linked list → block *chain*
Block Header

- **PoW nonce**: limit block creation rate to $1 / 10$min.
  - Give peers time to double-check block & transaction validity
  - Difficulty adjusted adaptively toward target block creation rate
PoW, a.k.a. “Difficult Math Puzzle”

\[ H(x, \text{prevHash}, \text{tStamp}, \text{txHash}) \leq 0x00...0FF...F \]

- Hash function \( H \) output unpredictable (by design)
  - No formula to solve for \( x \): Try all \( x \) until solution found!
  - Statistically, difficulty (expected \# of attempts) is \( 2^{(n-1)} \), where \( n \) is the \# of leading 0-bits at output of \( H \) func.

- Goals:
  - Control block creation rate (every 10 minutes for BTC)
  - Prohibit changes in previously settled (confirmed) blocks
Blockchain

- Non-repudiable ledger of confirmed-transactions
  - Peers always prefer longest known blockchain (per protocol)
  - PoW makes it unfeasible to recompute, catch up to peers
Depth, Height, Confirmations

Depth, #confirmations: ... 3 2 1 0
Height: ... N-2 N-1 N N+1

nonce prevHash

prevHash tStamp

prevHash
txHash

txHash

...
Depth, Height, Confirmations

Depth, #confirmations: ...

Height:

4 3 2 1
N-2 N-1 N N+1
Stale Blocks

- Multiple miners race to create next block
Stale Blocks

- Multiple miners race to create next block
- Winner broadcasts their block to all peers
Stale Blocks

- Multiple miners race to create next block
- Winner broadcasts their block to all peers
- Losers’ work-in-progress becomes *stale*
Stale Blocks

- Multiple miners race to create next block
- Winner broadcasts their block to all peers
- Losers’ work-in-progress becomes *stale*
- Race restarts at next level of the blockchain
Stale Blocks

- Multiple miners race to create next block
- Winner broadcasts their block to all peers
- Losers’ work-in-progress becomes *stale*
- Race restarts at next level of the blockchain
Stale Blocks

- Multiple miners race to create next block
- Winner broadcasts their block to all peers
- Losers’ work-in-progress becomes stale
- Race restarts at next level of the blockchain
- Stale blocks dropped
Stale Blocks

- Multiple miners race to create next block
- Winner broadcasts their block to all peers
- Losers’ work-in-progress becomes *stale*
- Race restarts at next level of the blockchain
- Stale blocks dropped
Orphan Blocks
Orphan Blocks

- Suddenly a valid, longer chain is announced
  - Presumably, after network delay or temp. partition
Orphan Blocks

- Suddenly, a valid longer chain is announced
  - Presumably, after network delay or temp. partition
- Previous branch becomes orphaned
Orphan Blocks

- Suddenly, a valid longer chain is announced
  - Presumably, after network delay or temp. partition
- Previous branch becomes orphaned
- Miners begin working on longer, preferred branch
Orphan Blocks

• Suddenly, a valid longer chain is announced
  – Presumably, after network delay or temp. partition
• Previous branch becomes orphaned
• Miners begin working on longer, preferred branch
<table>
<thead>
<tr>
<th>Candidate</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>0</td>
</tr>
<tr>
<td>Jim</td>
<td>0</td>
</tr>
<tr>
<td>Frank</td>
<td>0</td>
</tr>
</tbody>
</table>

State: 1
### State: 1

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>0</td>
</tr>
<tr>
<td>Jim</td>
<td>0</td>
</tr>
<tr>
<td>Frank</td>
<td>0</td>
</tr>
</tbody>
</table>

Bob: 1 vote

Frank: 1 vote
State: 2

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>1</td>
</tr>
<tr>
<td>Jim</td>
<td>0</td>
</tr>
<tr>
<td>Frank</td>
<td>1</td>
</tr>
</tbody>
</table>
**State: 1**

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>0</td>
</tr>
<tr>
<td>Jim</td>
<td>0</td>
</tr>
<tr>
<td>Frank</td>
<td>0</td>
</tr>
</tbody>
</table>

**State: 2**

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>1</td>
</tr>
<tr>
<td>Jim</td>
<td>0</td>
</tr>
<tr>
<td>Frank</td>
<td>1</td>
</tr>
</tbody>
</table>
State: 1

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>0</td>
</tr>
<tr>
<td>Jim</td>
<td>0</td>
</tr>
<tr>
<td>Frank</td>
<td>0</td>
</tr>
</tbody>
</table>

State: 2

*State 1 plus...*

Bob: 1 vote

Frank: 1 vote
General purpose blockchains

Messages are… anything!

Each block is the system state at that time

\[
\text{Current State} = \text{Original state} + \text{All Changes}
\]
Use cases abound

Payment System

Health Care Records

Real Estate Records
Addendum
High Energy Use

- As of Apr. 2018, the overall Bitcoin P2P network used cca. 930 kWh per *transaction* (not block)!
  - [https://digiconomist.net/bitcoin-energy-consumption](https://digiconomist.net/bitcoin-energy-consumption)
  - Only slightly more than the *monthly* use of the *average US home* (900 kWh as of 2016)
- Increase in perceived BTC value → more competing miners → harder PoW difficulty (to maintain 10-minute block creation interval)
  - Non-linear increase in per-transaction electricity use
- Turns out, decentralization is highly expensive!
Blockchain: Executive Summary

Pros:
Authentication built-in
Easy to audit history
Easy to detect data manipulation
Very difficult to disrupt

Cons:
Proof-of-work very inefficient
  • Alternatives exist!
State updates are slow
Best for simple computations
Bitcoin

Block #509169

Summary
- Number Of Transactions: 1915
- Output Total: 10,289.28130284 BTC
- Estimated Transaction Volume: 1,818.6925455 BTC
- Transaction Fees: 0.4893378 BTC
- Height: 509169 (Main Chain)
- Timestamp: 2018-02-14 15:16:59
- Received Time: 2018-02-14 15:16:59
- Relayed By: 58COIN
- Difficulty: 2,874,674,234,415.94
- Bits: 392992856
- Size: 1132.416 kB
- Weight: 3992.574 kJ
- Version: 0x20000000
- Nonce: 158986081
- Block Reward: 12.5 BTC

Transactions

No Inputs (Newly Generated Coins)

Hashes
- Hash: 00000000000000000000002c6b9c661b54b3653b2e723c5b7c2b659f4865500803
- Previous Block: 00000000000000000000001db302e03ac126ec503bbf42343c419e6b6f85f724d471
- Next Block(id): 3a2d859735c45c0d2b1e1e6907e9eb3e3f2b2a366c5c328035c05657fa03c9137e2

Unable to decode output address

2018-02-14 15:16:59

12.9893378 BTC
0.4893 BTC
5.66431602 BTC
## Ethereum

### Block 5089469

- **Hash:** 0x4b7ced1ac95607a06f8b0352468797bd038e8c1fb0fd4de2838f5712469c27
- **Difficulty:** 2,863,007,803,096,150
- **Miner:** `miningpoolhub1 (0xb293...)` (Mined in 19a)
- **Reward:** 3.13573 ETH | $2,777.29 (Block Reward: 3 ETH + Fee Reward: 0.13573 ETH + Uncle Inclusion Reward: 0 ETH)
- **Tx Fees:** 0.13573 ETH | $110.22 (4.33% of the total block reward)
- **Tx / Uncles:** 202 Transactions and 0 Uncles
- **Gas Limit:** 8,000,029
- **Gas Usage:** 83.8% (6,701,815 of 8,000,029)
- **Lowest Gas Price:** 1 GWei
- **Time:** 02/14/2018 10:31:12 AM (a minute ago)
- **Size:** 28,742 bytes
- **Extra:** t3 (Raw: 0x7433)

### Transactions

<table>
<thead>
<tr>
<th>Hash</th>
<th>Type</th>
<th>From</th>
<th>To</th>
<th>Value</th>
<th>Fee</th>
<th>Gas Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00622dc883...</td>
<td>Tx</td>
<td>0x5baeac0a9417a...</td>
<td>0x342d88c17df30...</td>
<td>0.03175 ETH</td>
<td>0.0021 ETH</td>
<td>100 GWei</td>
</tr>
<tr>
<td>0x2f71a28b88...</td>
<td>Tx</td>
<td>0x9607da642faa7...</td>
<td>0xee4d848e18c78...</td>
<td>0.01 ETH</td>
<td>0.00208 ETH</td>
<td>99 GWei</td>
</tr>
<tr>
<td>0x4a6a150361...</td>
<td>Tx</td>
<td>Bittrex (0xfbb1...)</td>
<td>0x419d0dd8d9af...</td>
<td>0 ETH</td>
<td>0.00531 ETH</td>
<td>90 GWei</td>
</tr>
<tr>
<td></td>
<td>Call</td>
<td>0x419d0dd8d9af...</td>
<td>0x267808e5246d1...</td>
<td>0 ETH</td>
<td>0.0028 ETH</td>
<td>90 GWei</td>
</tr>
<tr>
<td></td>
<td>Call</td>
<td>0x267808e5246d1...</td>
<td>0x6a516d48f93a...</td>
<td>0 ETH</td>
<td>0.00247 ETH</td>
<td>90 GWei</td>
</tr>
<tr>
<td>0x7359bb70de...</td>
<td>Tx</td>
<td>0x45a0ba49c5244...</td>
<td>0xaaa1a6e36ef20...</td>
<td>4.7598 ETH</td>
<td>0.00233 ETH</td>
<td>70 GWei</td>
</tr>
</tbody>
</table>
Existing blockchain programs are vulnerable

- Over **$40M** were stolen from TheDAO due to a bug in the implementation (June 2016)
- **$32M** were stolen due to a bug in a commonly used contract (June 2017)
- Bugs in smart contracts cannot be fixed after deployment

We want to build correct software, but current approaches have been shown to have security vulnerabilities
Obsidian: a new programming language

Goals
• Make certain vulnerabilities impossible
• Make it easier to write correct programs
• Show effectiveness and correctness

Components
1. Typestate-oriented programming
   • Shown to be helpful in documentation, but no studies of writing code
2. Resource types
   • Integration into an OO-style language is novel
Contact Information

Presenters

Elli Kanal
Technical Manager
Email:  ekanal@cert.org

Gabe Somlo
Cybersecurity Researcher
Email:  glsomlo@cert.org