PWL # 9:
“A Next-Generation Smart Contract and Decentralized Application Platform”

Vitalik Buterin

Papers We Love ❤️ Brasília

Presenter: Alessandro Leite

August 02, 2018

Outline

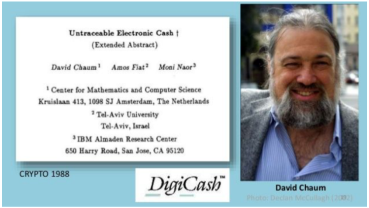
1. Introduction
2. Bitcoin
   - A Digital Assert State Transition System
   - Mining & Data Storage
   - Script Language
3. Ethereum
   - Philosophy
   - Ethereum State Machine
   - Smart Contracts
4. Applications
5. Conclusion

START

Libertarian Dream

"Privacy *is necessary for* ***an open society*** *in the* ***electronic age****. Privacy is not secrecy. A private matter is something one doesn't want the whole world to know, but a secret matter is something one doesn't want anybody to know.* ***Privacy is the power to selectively reveal oneself to the world"***
(A Cypherpunk's Manifesto, bit.ly/2LVETyZ)

Digicash (1988)



Untraceable Electronic Cash †

(Extended Abstract)

David Chaum[1] *Amos Fiat*[2] *Moni Naor*[3]

[1] Center for Mathematics and Computer Science
Kruislaan 413, 1098 SJ Amsterdam, The Netherlands

[2] Tel-Aviv University
Tel-Aviv, Israel

[3] IBM Almaden Research Center
650 Harry Road, San Jose, CA 95120

CRYPTO 1988

David Chaum

Photo: Declan McCullagh (2002)

- Based on cryptography primitives known as *Chaumian blinding*[a]
- Enabled users to sign off transactions without revealing anything about their identity
- Failed due to centralization

[a] D. Chaum, A. Fiat, and M. Naor. "Untraceable Electronic Cash". In: *Advances in Cryptology*. Springer-Verlag, 1988, pp. 319–327.

B-money (1998)

- Introduced important ideas and protocols to enable a decentralized digital ledger [a]
  - Hashcash proof of work as the way for creating money
  - Work verified by the community
  - Workers rewarded for expending compute power
  - Contracts/transactions enforced through broadcast and signing

[a] *weidai.com/bmoney.txt*

Reusable Proof-of-Work (RPoW) (2004)

- Based on Nick Szabo's *theory of collectibles* [a]
- RPoW [b] used ideas of *b-money* together with Adam Back's computationally difficult *Hashcash puzzles* to define the base for a cryptocurrency
- Failed due to the need to rely on trusted computing backend

[a] *https://nakamotoinstitute.org/shelling-out*

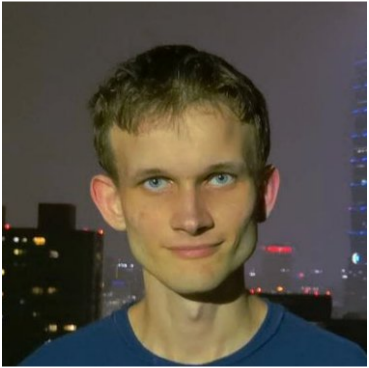
[b] *nakamotoinstitute.org/finney/rpow*

Bitcoin: A Peer-to-Peer Electronic Cash System (2008)

- An electronic payment system based on cryptographic proof instead of trust[a]
- Developed by a person or a group under the pseudonym of **Satoshi Nakamoto** in October 2008
- Provided a solution to implement distributed consensus based on Proof-of-Work (PoW): one-CPU equals one-vote
- It is in operation since early 2009
- It works without the management of any financial institution

[a] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. 2008. URL: bitcoin.org/bitcoin.pdf.

Ethereum: a next-generation smart contract and decentralized application platform (2013)



- It is a platform to implement and execute decentralized applications and smart contracts[a]
- It provides a Turing-complete programming language
- It relies on *ether*, its coin, as a "fuel" to execute the applications
- While Bitcoin **focuses** on store of value, or as an alternative for existing currency, Ethereum **focuses** on decentralized smart contracts.

[a] Vitalik Buterin. *Ethereum White Paper: a next-generation smart contract and decentralized application platform*. 2013.

Digital ledger as a state machine

- A ledger can be seen as **state transition system**
- A **state** represents the **ownership status** of all existing assets (e.g., cryptocurrency) and a **state transition function**
- A **state transition function** takes a state and a transaction and produces a new state

$$APPLY(S, TX) \rightarrow \{S' \text{ or ERROR}\}$$

State transition in traditional banking system

- A state represents a balance sheet (i.e., database), which can be only changed by one entity — the bank
- A transaction is an order to move \$X from A to B
- And the state transaction function subtract \$X value from A's account and increase B's balance by \$X, *iff* A's balance is at least \$X
- Example:

$$\begin{aligned} State(S) :& \{\text{Alice} : \$50, \text{Bob} : \$50\} \\ Transaction(TX) :& \{\text{Send \$20 from Alice to Bob}\} \\ \textbf{APPLY}(S, TX) =& \{\text{Alice} : \$30, \text{Bob} : \$70\} \end{aligned}$$

- But

$$\begin{aligned} Transaction(TX) :& \{\text{Send \$}\textbf{70}\text{ from Alice to Bob}\} \\ \textbf{APPLY}(S, TX) =& \textbf{ERROR} \end{aligned}$$

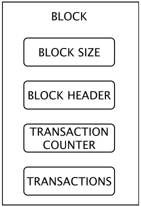
State transition in Bitcoin

- A state is the collection of all Unspent Transaction Outputs (UTXO) (i.e., coins)
- A transaction contains: (a) one or N inputs, $N \geq 1$; and (b) one or M outputs, $M \geq 1$
  - Each **transaction input** represents a reference to a UTXO and an address associated with the owner's cryptographic signature
  - Each **transaction output** contains a new UTXO to be added to the new state.

State transition in Bitcoin (cont.)

- The state transaction function APPLY$(S, TX) \to S'$ can be defined as follows:
  - For each input in TX:
    - if the provided referenced UTXO is not in S, returns an error
    - if the provided signature does not match the owner of the UTXO, returns a error
  - If the sum of all input UTXO is less than the sum of all UTXO output, returns an error
  - Return S' with all input UTXO removed and all output added

In Bitcoin, all the transactions are public but anonymous



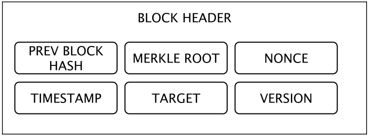
BLOCK

BLOCK SIZE

BLOCK HEADER

TRANSACTION COUNTER

TRANSACTIONS



BLOCK HEADER

PREV BLOCK HASH

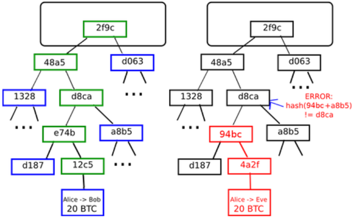
MERKLE ROOT

NONCE

TIMESTAMP

TARGET

VERSION



- Nodes collects the transactions into **blocks**
- Each block carries a **proof-of-work**
- A block does not keep any state
- All the transactions are kept in a data store known as **blockchain**
- Not all node need to have the entire Bitcoin's blockchain
- The simplified payment verification (SPV) protocol enables "light nodes" to download only the blocks associates with the transactions that are relevant to them

What is a blockchain?

BLOCKCHAIN

- Blockchain is:
  - **a transaction log** (= database)
  - **distributed** (= shared through a P2P network)
  - **secure** (= protected by cryptographic primitives)
  - **indestructible** (= or almost $\cdots$, as there are multiple copies distributed across the network)
  - **open** (= even if there is the option to store encrypted data)
  - **formed by blocks successively validated**, **timestamped**, and **chronologically organized**.

Bitcoin Scripting Language

- Stack-based programming language built specifically for Bitcoin
- Native support for cryptography
- Some limitations include:
  - **Lack of Turing-completeness**: it misses support for loops
  - **Value-blindness**: no way for a UTXO script to provide fine-grained control over the amount that can be withdraw
  - **Lack of state**: a UTXO can be either spent or unspent
  - **Blockchain-blindness**: UTXO are blind to blockchain data such as the nonce, the timestamps, and the previous block.

Ethereum platform

ethereum

- Designed to be a platform for building decentralized large-scale applications
- Focus rapid development time, security for small and rarely used applications, and the ability to enable applications to interact with each other
- Its blockchain provides a Turing-complete programming language

Ethereum design focuses on short time-to-market

- Simplicity
- Universality
- Modularity
- Agility
- Non-discrimination

Ethereum Account

- State is made up of objects called accounts
- An account contains four fields:
  - **Nonce**
  - Account's current **ether balance**
  - Account's **contract code**
  - Account's **storage**

Smart contracts are autonomous agents

- Autonomous agents living inside of an Ethereum execution environment
- Automatically execute the terms and conditions previously defined between the parties
- Work as all conditional expressions
- Have control of their own ether balance and their own data storage

That's all Folks!