

Ethereum

Saravanan Vijayakumaran
sarva@ee.iitb.ac.in

Department of Electrical Engineering
Indian Institute of Technology Bombay

August 27, 2019

Ethereum

- A blockchain platform for building decentralized applications
 - Application code and state is stored on a blockchain
 - Transactions cause code execution and update state, emit events, and write logs
 - Frontend web interfaces can respond to events and read logs
- Most popular platform for creating new tokens (ICOs)
 - Each ICO implements a ERC-20 token contract (link)
 - Investments in ICOs was about \$7 billion in 2017
 - About \$12 billion in H1 of 2018
- Other applications
 - Ethereum Name Service (<https://ens.domains/>)
 - Cryptokitties (<https://www.cryptokitties.co/>)
 - Fomo3D (<https://fomo3d.hostedwiki.co/>)
 - Decentralized exchanges (<https://idex.market>)

Ethereum History

- Proposed by then 19 y.o. Vitalik Buterin in 2013
- VB visited the Mastercoin team in Oct 2013
- Released the Ethereum white paper in Dec 2013
- Bitcointalk announcement on Jan 24th, 2014
- A presale in July-Aug 2014 collected 31,591 BTC worth 18 million USD in return for 60,102,216 ETH
- About 12 million ETH created to pay early contributors and setup non-profit foundation
- Ethereum notable releases
 - Release 1.0: Frontier on 30 July, 2015
 - Release 2.0: Homestead on 14 March, 2016
 - Release 2.1: DAO Hard Fork on 20 July, 2016
 - Release 3.0: Metropolis phase 1 (Byzantium) on 16 Oct, 2017
 - Support for zkSNARKs
 - Release 3.5: Metropolis phase 2 (Constantinople) on 28 Feb, 2019
 - Release 4.0: Serenity, TBA
 - Move from proof-of-work to proof-of-stake

<https://metamask.io/>

Bitcoin vs Ethereum

	Bitcoin	Ethereum
Specification	Bitcoin Core client	Ethereum yellow paper
Consensus	SHA256 PoW	Ethash PoW (later PoS)
Contract Language	Script	EVM bytecode
Block interval	10 minutes	14 to 15 seconds ¹
Block size limit	approx 4 MB	14 KB to 36 KB (Jan 2019 to Jul 2019) ²
Difficulty adjustment	After 2016 blocks	After every block
Currency supply	Fixed to 21 million	Variable (107 million in Aug 2019) ³
Currency units	1 BTC = 10^8 satoshi	1 ETH = 10^{18} Wei

¹<https://etherscan.io/chart/blocktime>

²<https://etherscan.io/chart/blocksize>

³<https://etherscan.io/chart/ethersupplygrowth>

Ethereum Specification

- Specified in the Ethereum yellow paper by Gavin Wood
- Implemented in Go, C++, Python, Rust
- Yellow paper models Ethereum as a transaction-based state machine
 - σ_t = State at time t , T = Transaction, Υ = Transaction-level state-transition function

$$\sigma_{t+1} = \Upsilon(\sigma_t, T)$$

- B = Block (series of transactions and other stuff), Π = Block-level state-transition function

$$\sigma_{t+1} = \Pi(\sigma_t, B)$$

$$B = (\dots, (T_0, T_1, \dots), \dots)$$

- Ω = Block finalization state-transition function

$$\Pi(\sigma, B) = \Omega(B; \Upsilon(\Upsilon(\sigma, T_0), T_1) \dots)$$

Ethereum World State

- World state consists of *accounts*
- Account types
 - **Externally owned accounts:** Controlled by private keys
 - **Contract accounts:** Controlled by contract code
- Account state
 - **nonce:** Number of transactions sent or contract-creations made
 - **balance:** Number of Wei owned by this account
 - **storageRoot:** Root hash of storage Merkle Patricia trie
 - **codeHash:** Hash of EVM code if contract account
- Mapping between account addresses and states is stored in **state database**
- Each account has a 20-byte address
 - EOA address = Right-most 20 bytes of Keccak-256 hash of public key
 - Contract address = Right-most 20 bytes of Keccak-256 hash of `RLP([senderAddress, nonce])`

Keccak-256

- Cryptographic hash function used by Ethereum
- NIST announced competition for new hash standard in 2006
- Keccak declared winner in 2012
- In August 2015, FIPS 202 “*SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*” was approved
- Ethereum adopted Keccak-256 but NIST changed the padding scheme
- Keccak-256 and SHA3-256 give different outputs for the same message
 - `https://ethereum.stackexchange.com/questions/550/which-cryptographic-hash-function-does-ethereum-use`

Transactions

- Two types
 - Contract creation
 - Message calls
- Contract creation transactions create new contracts on the blockchain
 - Destination address is null
 - EVM code for account initialization is specified
- Message call transactions call methods in an existing contract
 - Input data to contract methods is specified
- Transaction execution modifies the state database

Ethereum Smart Contracts

Ethereum Contracts

- Contract = Collection of functions and state at a specific address
 - Account state = [nonce, balance, storageRoot, codeHash]
- Created by contract creation transactions
- Contract logic is stored in EVM bytecode
- Written in a high level language which compiles to bytecode
 - Solidity <https://solidity.readthedocs.io>
 - Vyper <https://vyper.readthedocs.io>
- Anatomy of a contract
 - State variables
 - Functions
 - Events

Currency Example

Currency Example

```
1  pragma solidity ^0.4.7;
2
3  contract Coin {
4      address public minter;
5      mapping (address => uint) public balances;
6
7      event Sent(address from, address to, uint amount);
8
9      constructor() public {
10         minter = msg.sender;
11     }
12
13     function mint(address receiver, uint amount) public {
14         if (msg.sender != minter) return;
15         balances[receiver] += amount;
16     }
17
18     function send(address receiver, uint amount) public {
19         if (balances[msg.sender] < amount) return;
20         balances[msg.sender] -= amount;
21         balances[receiver] += amount;
22         emit Sent(msg.sender, receiver, amount);
23     }
24 }
```

Currency Example Anatomy

- State variables

```
address public minter;  
mapping (address => uint) public balances;
```

- Functions

```
constructor() public {..}  
  
function mint(address receiver, uint amount) public {..  
  
function send(address receiver, uint amount) public {..}
```

- Events

```
event Sent(address from, address to, uint amount);
```

Contract Creation and Currency Allotment

- At contract creation, `minter` is initialized to creator

```
address public minter;  
  
constructor() public {  
    minter = msg.sender;  
}
```

- `minter` can call `mint` and allot currency to addresses

```
mapping (address => uint) public balances;  
  
function mint(address receiver, uint amount) public {  
    if (msg.sender != minter) return;  
    balances[receiver] += amount;  
}
```

- Public functions form the contract interface (can be called via message call)
- Private functions and variables are only visible in original contract, not in derived contracts

Currency Transfers

```
mapping (address => uint) public balances;

event Sent(address from, address to, uint amount);

function send(address receiver, uint amount) public {
    if (balances[msg.sender] < amount) return;
    balances[msg.sender] -= amount;
    balances[receiver] += amount;
    emit Sent(msg.sender, receiver, amount);
}
```

- Once allotted currency, address owners can transfer to others
- An event is emitted to enable light clients to find this log
- Remix Demo <https://remix.ethereum.org>

References

- **White paper** <https://github.com/ethereum/wiki/wiki/White-Paper>
- **Ethereum Wikipedia Article** <https://en.wikipedia.org/wiki/Ethereum>
- **A Prehistory of the Ethereum Protocol**
<https://vitalik.ca/general/2017/09/14/prehistory.html>
- **Ethereum announcement on Bitcointalk**
<https://bitcointalk.org/index.php?topic=428589.0>
- **History of Ethereum** <http://ethdocs.org/en/latest/introduction/history-of-ethereum.html>
- **The DAO Wikipedia Article**
[https://en.wikipedia.org/wiki/The_DAO_\(organization\)](https://en.wikipedia.org/wiki/The_DAO_(organization))
- **Yellow paper** <https://ethereum.github.io/yellowpaper/paper.pdf>
- **Solidity Documentation** <https://solidity.readthedocs.io>
- **Remix IDE** <https://remix.ethereum.org>
- **Metamask** <https://metamask.io/>