Blockchain technology

Jiangang Wu

imahero@gmail.com

Aug. 2, Toronto

Contents

Contents

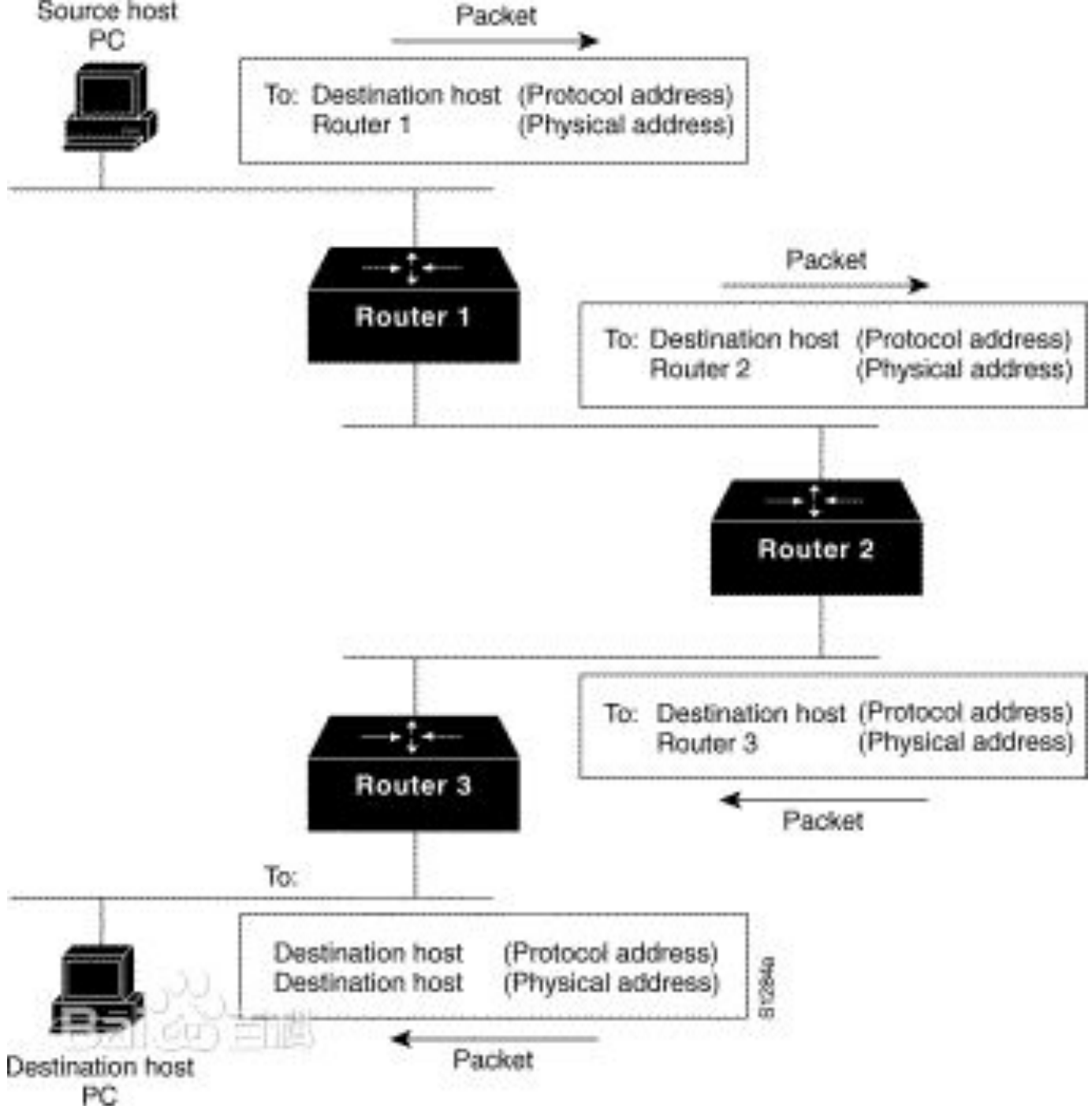
1. Topology of peer to peer network

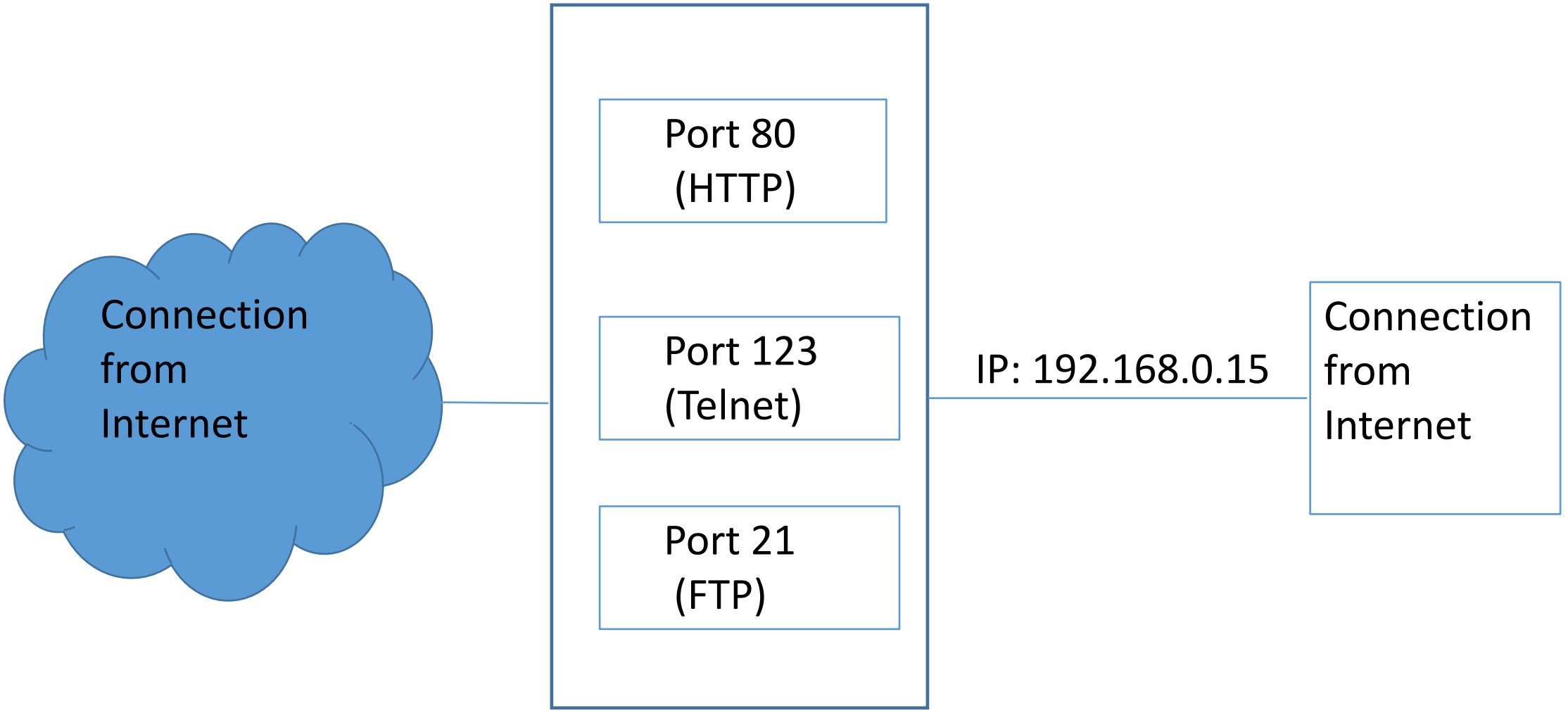
- Some basics
- Protocol and P2P network
- History of P2P network
- Topology
- Main topologies of blockchain network

1.1 Some basics--Open Systems Interconnection model (OSI model)

OSI Model			
Layer		Protocol data unit (PDU)	Function
Host layers	7. Application	Data	High-level APIs, including resource sharing, remote file access
	6. Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption
	5. Session		Managing communication sessions, i.e. continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4. Transport	Segment, Datagram	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing
Media layers	3. Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control
	2. Data link	Frame	Reliable transmission of data frames between two nodes connected by a physical layer
	1. Physical	Symbol	Transmission and reception of raw bit streams over a physical medium

Source: https://en.wikipedia.org/wiki/OSI_model


Source host
PC
Packet
To: Destination host (Protocol address)
Router 1 (Physical address)
Router 1
Packet
To: Destination host (Protocol address)
Router 2 (Physical address)
Router 2
To: Destination host (Protocol address)
Router 3 (Physical address)
Packet
Router 3
To:
Destination host (Protocol address)
Destination host (Physical address)
Packet
Destination host
PC
S12284a



Once a connection is established it is know as a "session."

Protocol: TCP VS UDP

- TCP (Transmission Control Protocol):
  - The most commonly used protocol on the Internet.
  - TCP offers error correction and flow control.
  - When the TCP protocol is used there is a "guaranteed delivery."
  - Process:
    - By using "flow control”, data needs to be re-sent, and stops the flow of data until previous packets are successfully transferred.
    - When a collision occurs, the client re-requests the packet from the server until the whole packet is complete and is identical to its original.
- Major internet applications such as the World Wide Web, email, remote administration, and file transfer rely on TCP.

Protocol: TCP VS UDP

- UDP (User Datagram Protocol):
  - Anther commonly used protocol on the Internet.
  - UDP is never used to send important data such as webpages, database information, etc.
  - UDP is commonly used for streaming audio and video.
    - Streaming media such as Windows Media audio files (.WMA) , Real Player (.RM), and others use UDP because it offers speed!
    - The reason UDP is faster than TCP is because there is no form of flow control or error correction.
    - The data sent over the Internet is affected by collisions, and errors will be present. Remember that UDP is **only** concerned with speed. This is the main reason why streaming media is not high quality.

Server-Based Networks

Pro's	Con's
Easily managed security policy	Files are not accessible if the server goes down
Files are centrally located	Hackers target the server as a central point to gain full access of a network
Files are easy to backup	Network servers can be expensive

Peer-to-Peer Networks

Pro's

No server needed, equipment is cheap

Network traffic is distributed among clients

Con's

Security Policy can be hard to control

Backup is challenging

Files are scattered across different machines

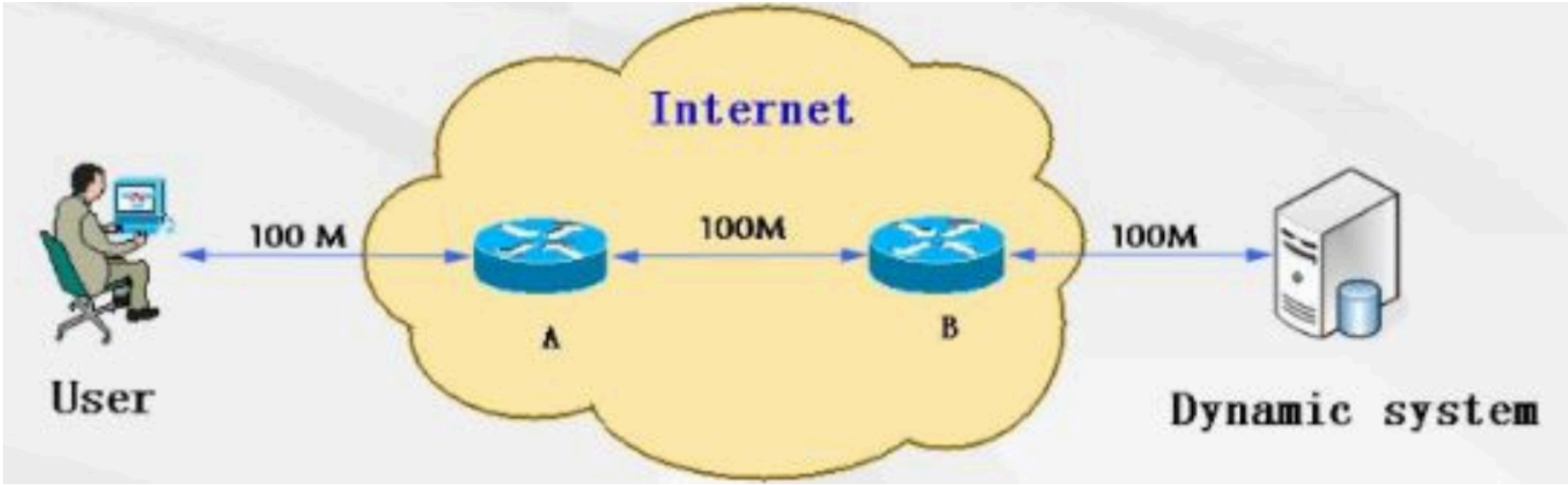
History of P2P network

- P2P networks have been typically used for file sharing applications
  - eDonkey Network (2000)
  - WinMX
  - BitTorrent
- Features
  - There is not any central hub for the network
  - Each peer is both a client and a server.
  - Files are not stored on a central server
  - Enable peers to share digitized content such as general documents, audio, video, electronic books, etc
  - Recently, more advanced applications such as real-time conferences, online gaming, and media streaming

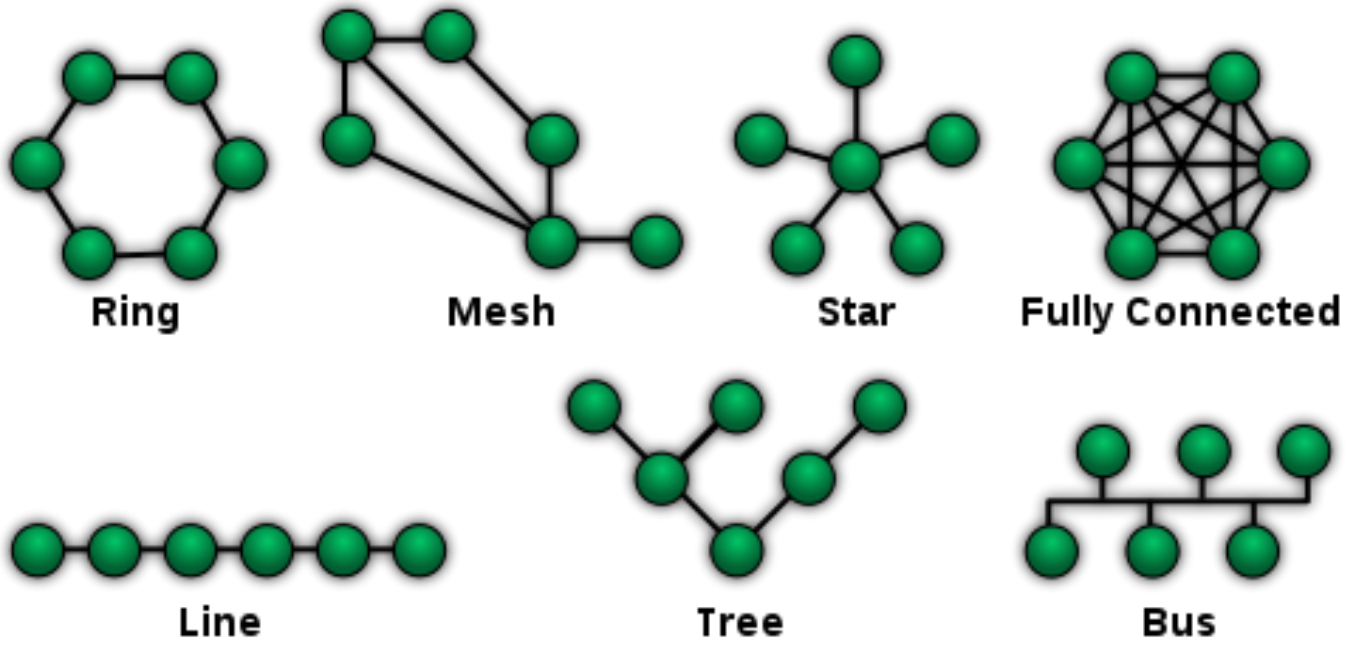
History of P2P network

- Ever popular
  - During the years of 2008-2009, show that P2P networks generated most of the traffic in all monitored regions, ranging from 43% in Northern Africa to 70% in Eastern Europe (http://www.ipoque.com/)
  - BitTorrent (Cohen, 2003) is the most popular protocol on the Internet, generating most of the traffic in 7 out of 8 regions ranging from 32% in South Africa to 57% in Eastern Europe.
- Not popular any more
  - They can take 30%-50% of the workload of network, which is too much for Internet service provider
  - Media delivery and streaming services over the Internet such as YouTube, PPLive, and Internet video broadcasting (e.g., AOL broadcast, MSNBC, CBS, etc.) have emerged.
  - They may infringe copyright
  - CDN (content delivery network) and cloud service are comercialized

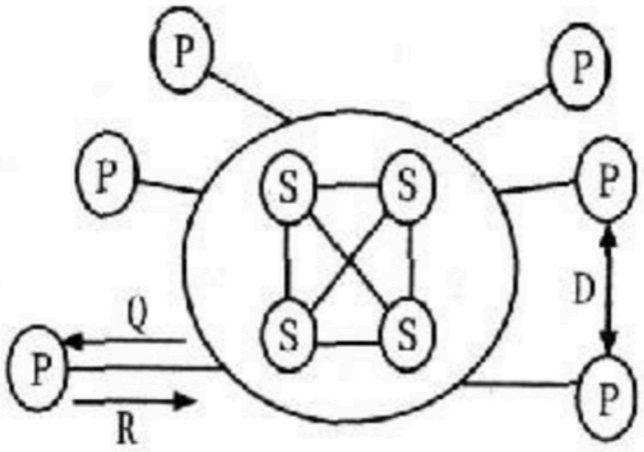
CDN



Topology



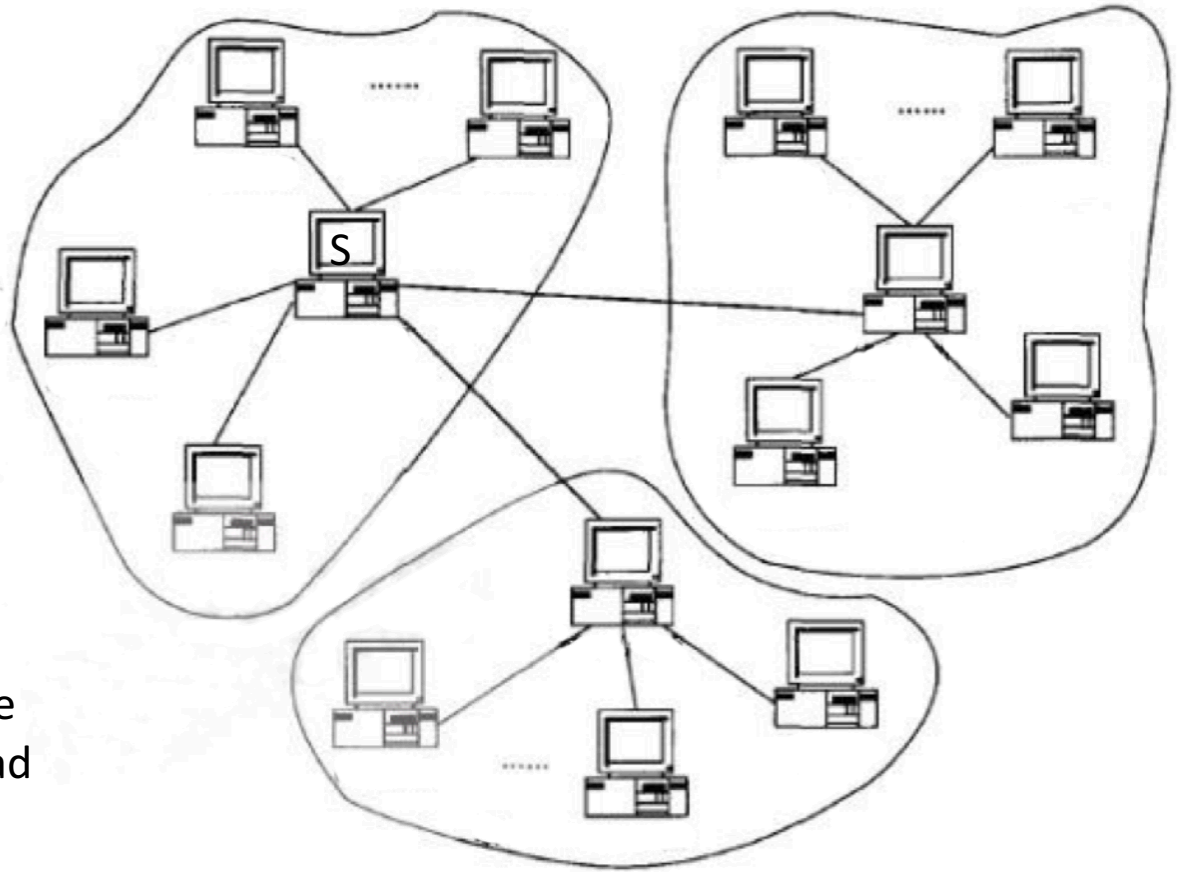
Topology of P2P network



P Peer

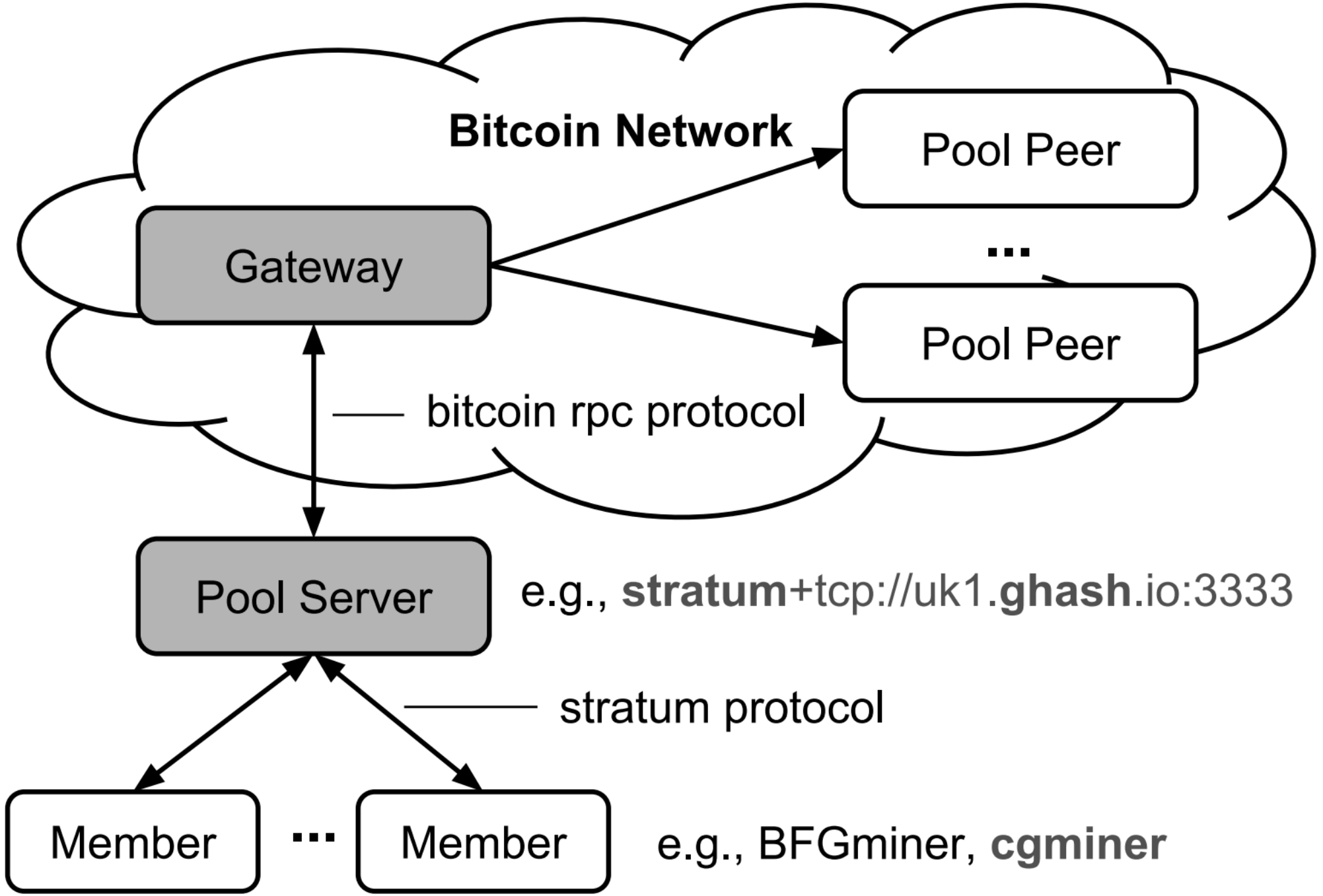
S Server

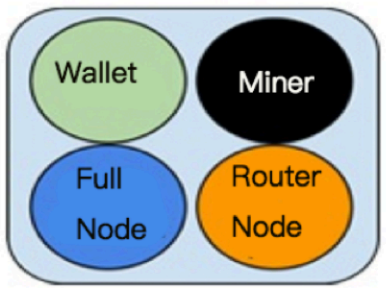
Q: Quest
R: Response
D: Download



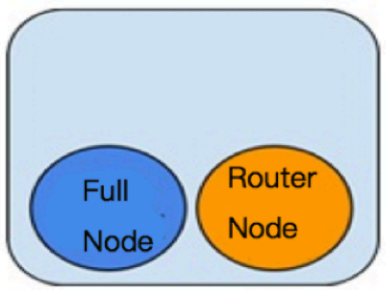
Bitcoin network topology

- Bitcoin protocol
- Pool protocol
- Different functional nodes

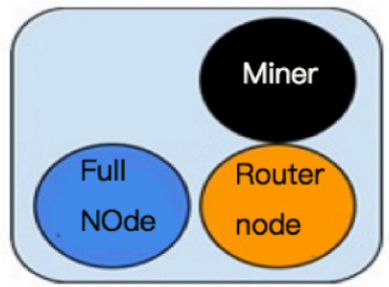

Bitcoin Network
Pool Peer
Gateway
...
Pool Peer
bitcoin rpc protocol
Pool Server
e.g., stratum+tcp://uk1.ghash.io:3333
stratum protocol
Member
...
Member
e.g., BFGminer, cgminer



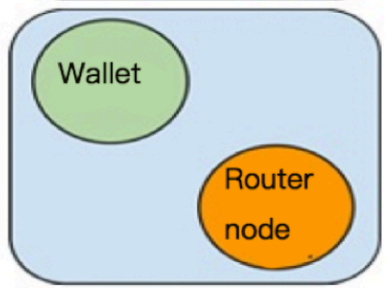
Bitcoin core



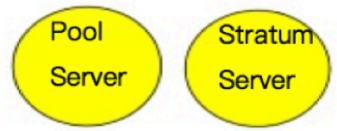
Full node



Solo miner

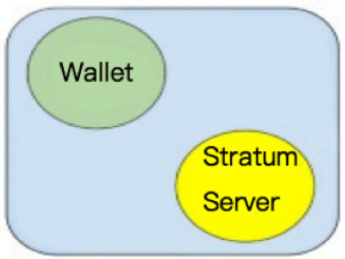


SPV wallet

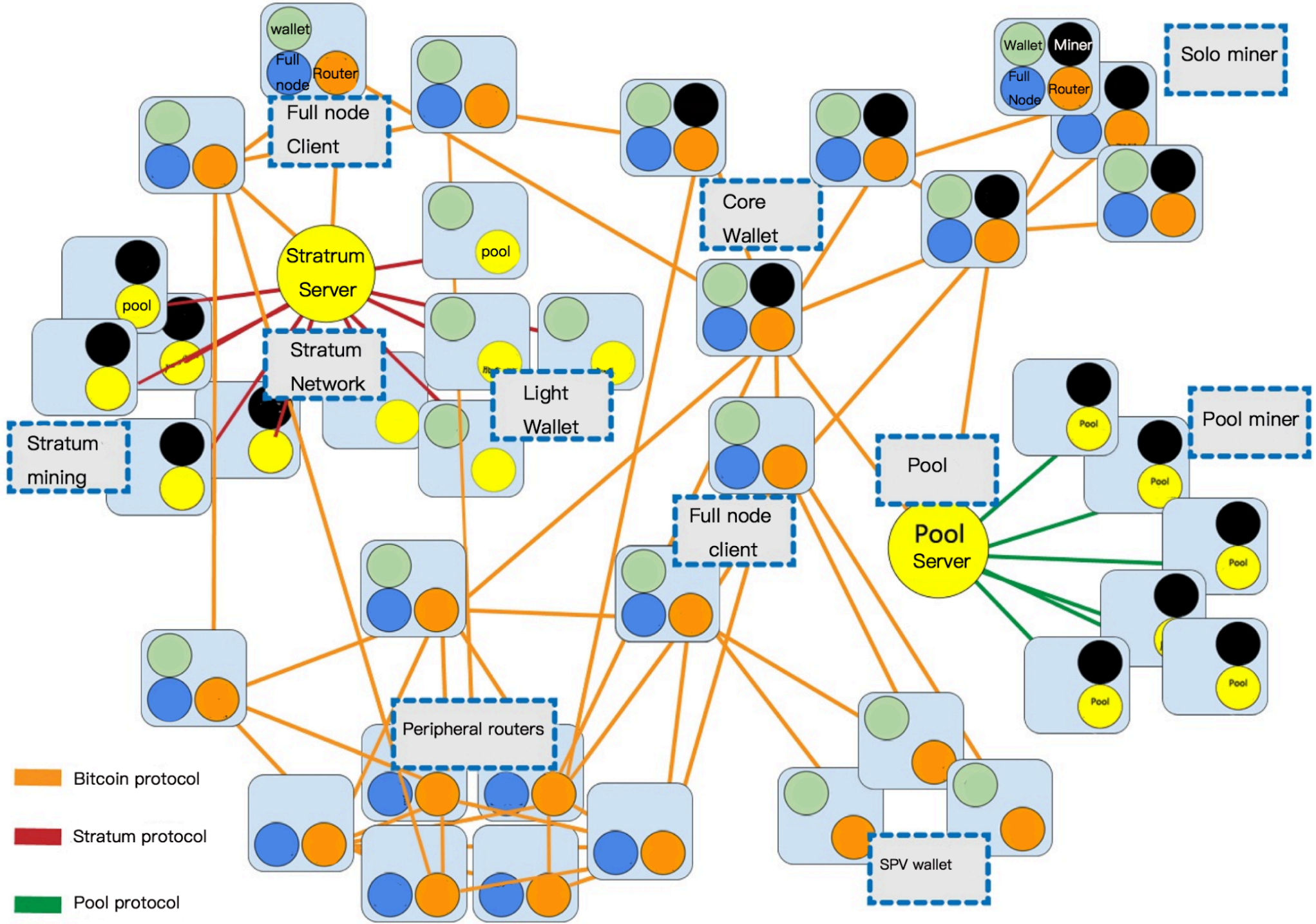


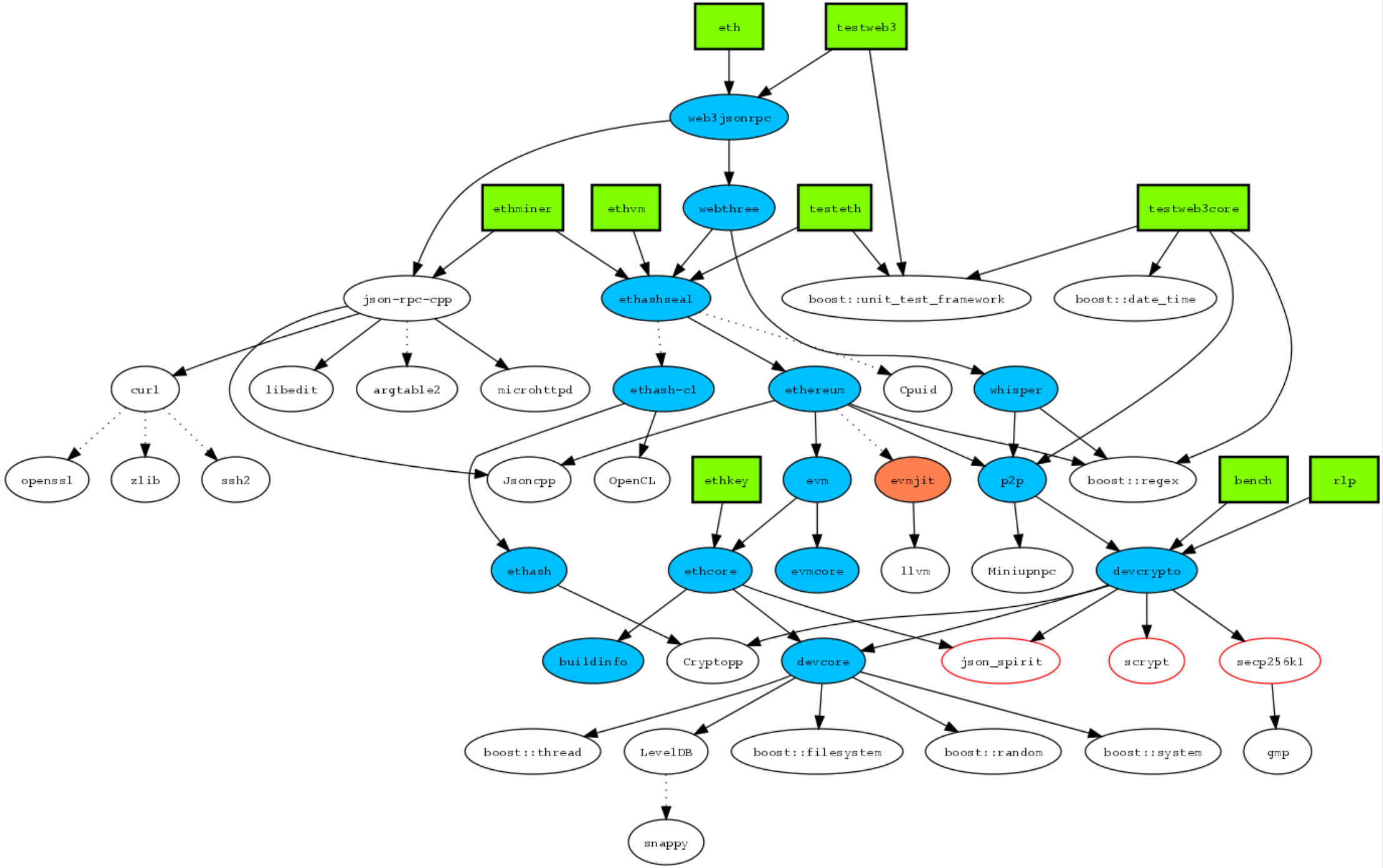
Pool protocol server

Pool miners



SPV Stratum wallet


wallet
Full node
Router
Full node Client
Wallet
Miner
Full Node
Router
Solo miner
Core Wallet
pool
Stratrum Server
pool
Stratum Network
Light Wallet
Stratum mining
Pool
Pool
Pool
Pool
Pool miner
Full node client
Pool Server
Peripheral routers
SPV wallet
Bitcoin protocol
Stratum protocol
Pool protocol



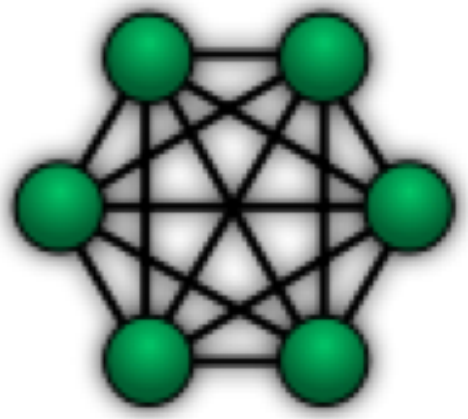
Ethereum C++ dependencies

Internet of Information	Internet of Values
	
TCP	UDP
C/S, B/S	P2P
PC, server, Internet of Things	S9, BITSTAMP, BTC ATM
ASCII, GB2312, UTF, jif, png, jpeg, mp3, wma, etc.	ERC20, ERC721
Software, Websites, Apps, etc.	Smart contracts, DApp
Operation system, Languages, IDE, VM, Docker, Hadoop, etc.	ETH, EOS, solidity, …

Contents

Trustless P2P network and collective decision

- How to make a collective decision based on P2P network?
- Leslie Lamport and Byzantine fault tolerance
  - To collect signature more than two third of the nodes
  - Too much network communication: exponential growth with number of nodes
- PBFT: 33%
- Bitcoin:
  - Economic incentives
  - PoW
  - 51%



PoW

- CPU war
- Advantage: high safety
- Weakness: not environmentally friendly, slow, but also centralized

PoS and DPoS

- PoS: proof of stake
- DPoS: delegate proof of stake
- Money war: Select a bookkeeping node by coin-days (not randomly)
- Advantages: environmental protection, high performance
- Disadvantages: centralization, widening the gap between rich and poor

PBFT and PoS+PBFT

- BFT:
  - One third tolerance
  - Need to collect two thirds of signatures
- PBFT: Practical Byzantine Fault Tolerance (http://pmg.csail.mit.edu/papers/osdi99.pdf)
  - Primary-backup
    - View: in a view, there will be a primary replica, others are replicas nodes (backups)
    - Replicas will check the behavior of primary replica, they can change the view
  - Quorum replication
    - To make sure data safety
- PoS + PBFT: choose qualified nodes to do PBFT
- Advantages: environmentally friendly, no forks
- Disadvantages: centralization, widening the gap between rich and poor

Consensus

Platform	Consensus	Status	Smart contract	Transaction
Bitcoin	PoW	UTXO	Pure function	Payment
Ethereum	PoW/PoS	Accounts	Turing complete	Smart contract
Fabric	Replaceable	Accounts	Turing complete	Smart contract
NEO/EOS	PBFT	Accounts	Turing complete	Smart contract
R3 Corda	Notary	UTXO	Pure function	Transaction flow

Platform	Execution environment	Data persistence	Block generation	Account design	KYC
Bitcoin	Script	None	10m	UTXO	None
Ethereum	EVM	Yes	15s	Balance	Noe
Fabric	Docker	Yes	Definable	Not applicable	Digital Certificate
NEO/EOS	VM	Yes	15-20s/1s	Balance	KYC of voting node
R3 Corda	JVM	None	Not applicable	UTXO	Digital Certificate

Bitcoin-consensus: proof of work

- Hash
- Merkel tree
- Elliptical encryption
- Asymmetric signature
- Transaction

Hash

- Hash functions are a class of mathematical functions that have the following three characteristics:
  - The input can be a string of any size;
  - It produces fixed-size output, such as 256 bits;
  - It can perform efficient calculations, that is, for a specific input string, the output can be derived within a reasonable time.
    - More precisely, the corresponding bit string, whose hash value is calculated to have a complexity of O( ), is a linear function.
- Three additional features:
  - Collision resistance
  - Concealment
  - The puzzle is friendly

Secure hash algorithm (SHA)

- SHA is a series of cryptographic hash functions designed by the National Security Agency (NSA) and published by the National Institute of Standards and Technology (NIST). It has undergone the development of SHA-0, SHA-1, SHA-2, and SHA-3 series.
- Bitcoin uses the SHA256 algorithm, which belongs to the SHA-2 series and was recognized as one of the safest and most advanced algorithms in 2008 when Nakamoto invented Bitcoin.

SHA256: to introduce 7 numbers as a to f

```
s[0] = 0x6a09e667ul;
s[1] = 0xbb67ae85ul;
s[2] = 0x3c6ef372ul;
s[3] = 0xa54ff53aul;
s[4] = 0x510e527ful;
s[5] = 0x9b05688cul;
s[6] = 0x1f83d9abul;
s[7] = 0x5be0cd19ul;
```

SHA256 : to use a to f to calculate w0 to w15

```
Round(a, b, c, d, e, f, g, h, 0x428a2f98, w0 = ReadBE32(chunk + 0));
Round(h, a, b, c, d, e, f, g, 0x71374491, w1 = ReadBE32(chunk + 4));
Round(g, h, a, b, c, d, e, f, 0xb5c0fbcf, w2 = ReadBE32(chunk + 8));
Round(f, g, h, a, b, c, d, e, 0xe9b5dba5, w3 = ReadBE32(chunk + 12));
Round(e, f, g, h, a, b, c, d, 0x3956c25b, w4 = ReadBE32(chunk + 16));
Round(d, e, f, g, h, a, b, c, 0x59f111f1, w5 = ReadBE32(chunk + 20));
Round(c, d, e, f, g, h, a, b, 0x923f82a4, w6 = ReadBE32(chunk + 24));
Round(b, c, d, e, f, g, h, a, 0xab1c5ed5, w7 = ReadBE32(chunk + 28));
Round(a, b, c, d, e, f, g, h, 0xd807aa98, w8 = ReadBE32(chunk + 32));
Round(h, a, b, c, d, e, f, g, 0x12835b01, w9 = ReadBE32(chunk + 36));
Round(g, h, a, b, c, d, e, f, 0x243185be, w10 = ReadBE32(chunk + 40));
Round(f, g, h, a, b, c, d, e, 0x550c7dc3, w11 = ReadBE32(chunk + 44));
Round(e, f, g, h, a, b, c, d, 0x72be5d74, w12 = ReadBE32(chunk + 48));
Round(d, e, f, g, h, a, b, c, 0x80deb1fe, w13 = ReadBE32(chunk + 52));
Round(c, d, e, f, g, h, a, b, 0x9bdc06a7, w14 = ReadBE32(chunk + 56));
Round(b, c, d, e, f, g, h, a, 0xc19bf174, w15 = ReadBE32(chunk + 60));
```

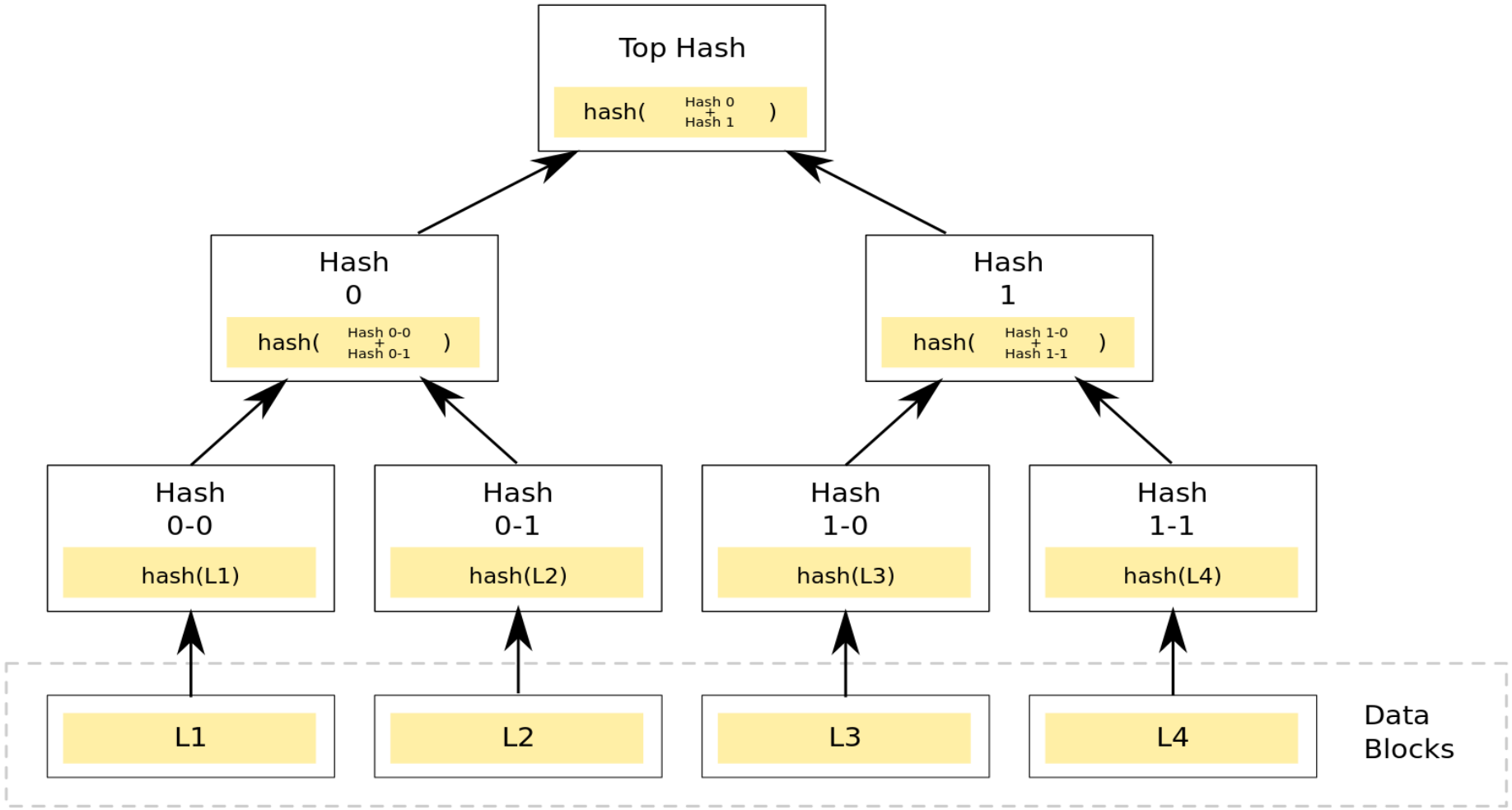
SHA256 : to introduce 6 function algorithms

- Introducing 6 function algorithms combined by bit operations and logic operations to operate from 0 to 15 will continuously replace the loops, using 48 pairs of operations. Constantly generate new ones and replace them with new ones, and finally get the latest ones.
- Round(a, b, c, d, e, f, g, h, 0xe49b69c1, w0 += sigma1(w14) + w9 + sigma0(w1));
- Round(h, a, b, c, d, e, f, g, 0xefbe4786, w1 += sigma1(w15) + w10 + sigma0(w2));
- Round(g, h, a, b, c, d, e, f, 0x0fc19dc6, w2 += sigma1(w0) + w11 + sigma0(w3));
- ……
- In each of the 64 rounds of operations in steps 4 and 5, a constant is changed for each of a total of 64 constants. These 64 constants are the fraction of the first 64 prime cube roots multiplied by 2^{32} and then rounded off.

To bring waterfall changes

- We can calculate : "The quick brown fox jumps over the lazy dog” to get the hash:
  - 'd7a8fbb307d7809469ca9abcb0082e4f8d5651e46d3cdb762d02d0bf37c9e592'。
- We can calculate"The quick brown fox jumps over the lazy dog.” to get the hash:
  - ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c'。

Merkel tree


Top Hash
hash(Hash 0 + Hash 1)
Hash 0
hash(Hash 0-0 + Hash 0-1)
Hash 1
hash(Hash 1-0 + Hash 1-1)
Hash 0-0
hash(L1)
Hash 0-1
hash(L2)
Hash 1-0
hash(L3)
Hash 1-1
hash(L4)
L1
L2
L3
L4
Data Blocks

Proof of work

SHA256(Top hash of transactions + miner’s address + nonce) should be smaller than a number D.

To adjust the D will change mining difficulty.

```
import time
import random
import hashlib
start = time.clock()
i = 0
while True:
    nonce = random.randint(1,10**32)
    h = hashlib.sha256(str(nonce).encode('utf-8')).hexdigest()
    if h<"000f0000000000000000000000000000000000000000000000000000000000":
        print(h)
        print(nonce)
        end = time.clock()
        print(end - start)
        break
```

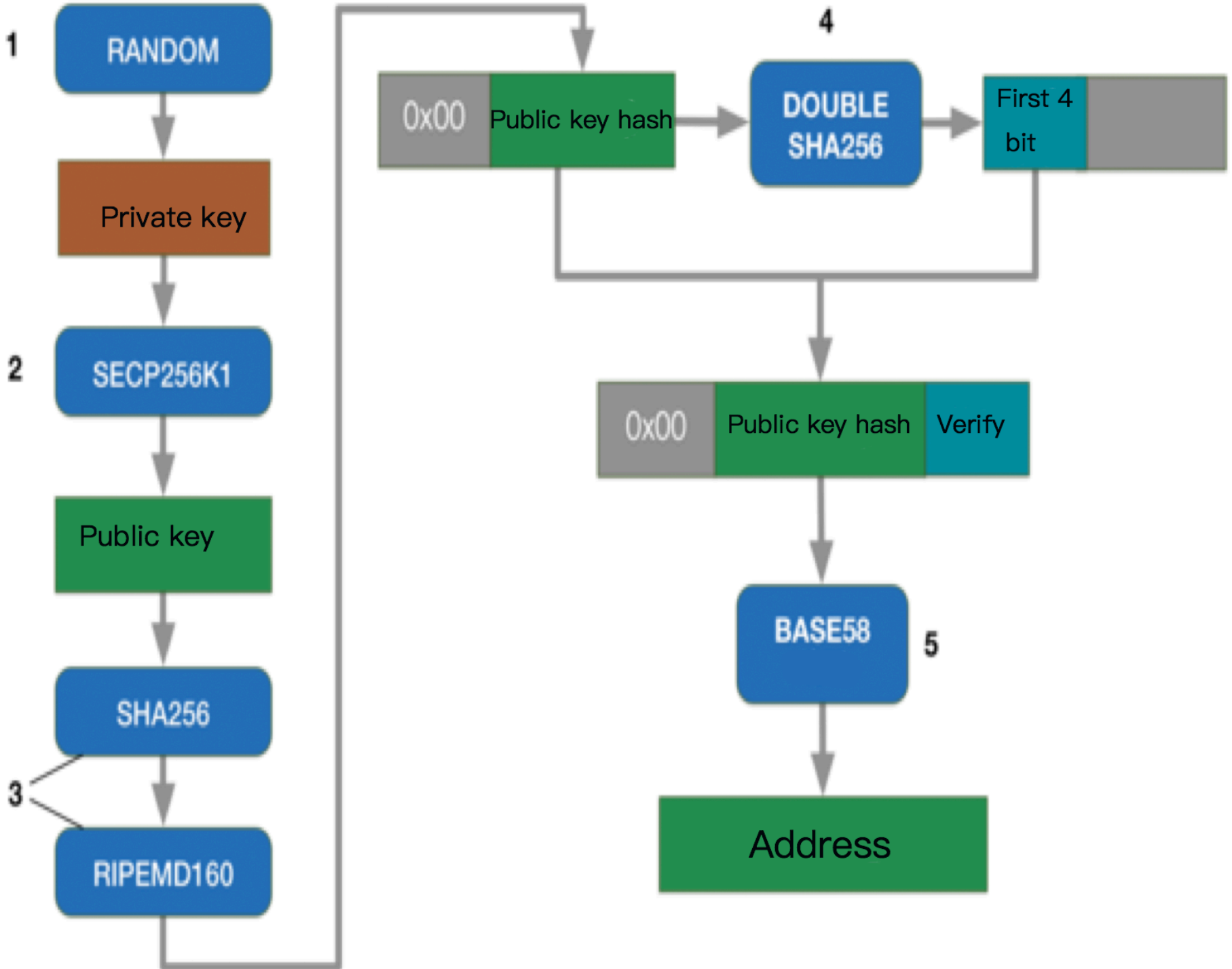
Current difficulty

- 2.4 EH/s（E is 1024P, 1P is about 1 million G）:
  - One billion common computers.
- Difficulty adjustment: every 2016 blocks
  - To make sure every block is in 10 minutes

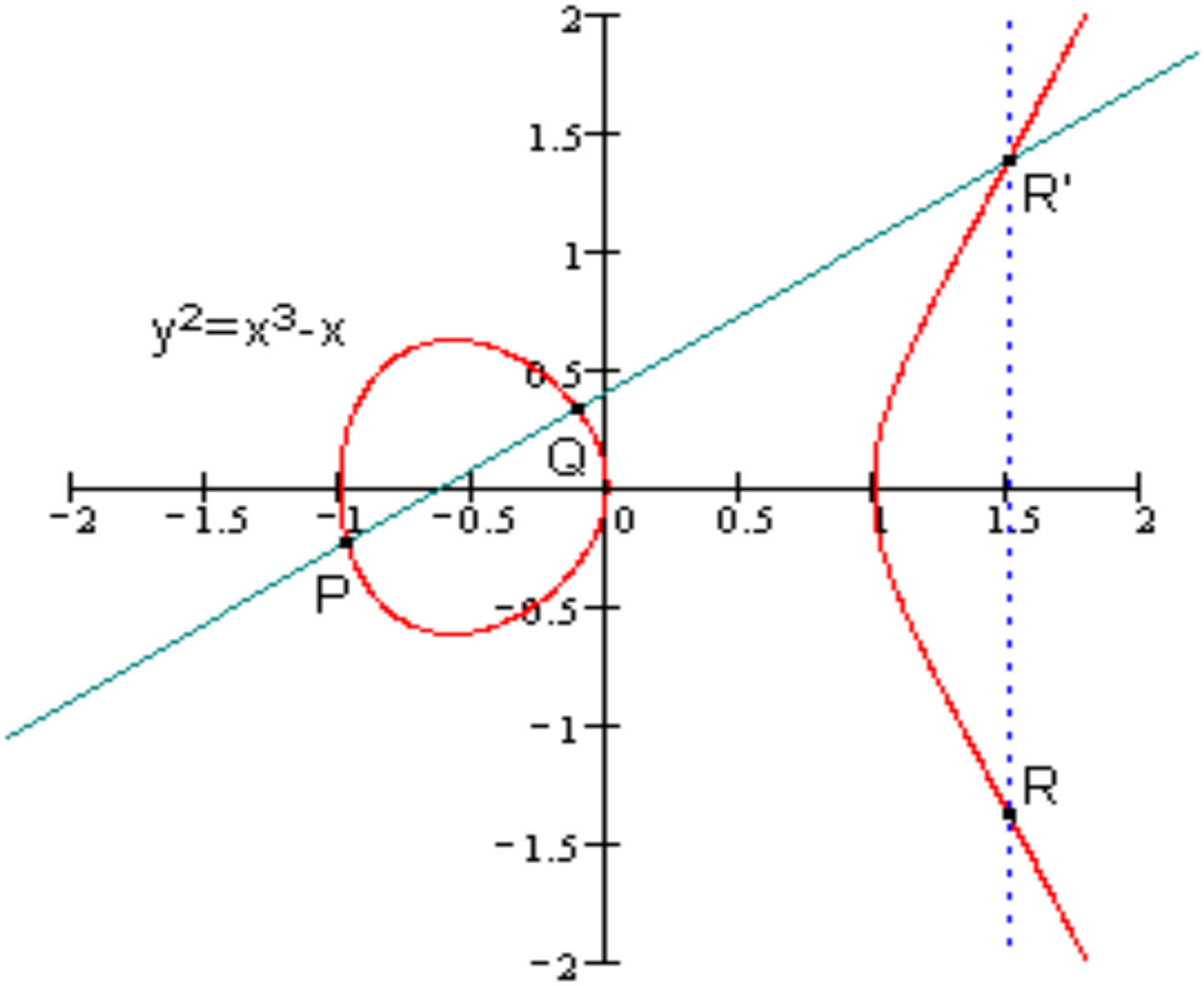
Pools

0	NETWORK	100.00 %	2.36 EH/s
1	AntPool	18.34 %	433.44 PH/s
2	F2Pool	14.50 %	342.72 PH/s
3	BW.COM	10.45 %	246.96 PH/s
4	BTCC	9.59 %	226.80 PH/s
5	SlushPool	7.25 %	171.36 PH/s
6	BitFury	6.18 %	146.16 PH/s
7	HaoBTC	5.97 %	141.12 PH/s
8	BTC.com	5.33 %	126.00 PH/s
9	GBMiners	5.12 %	120.96 PH/s
10	ViaBTC	3.84 %	90.72 PH/s
11	1Hash	3.41 %	80.64 PH/s
12	BitClub	2.77 %	65.52 PH/s

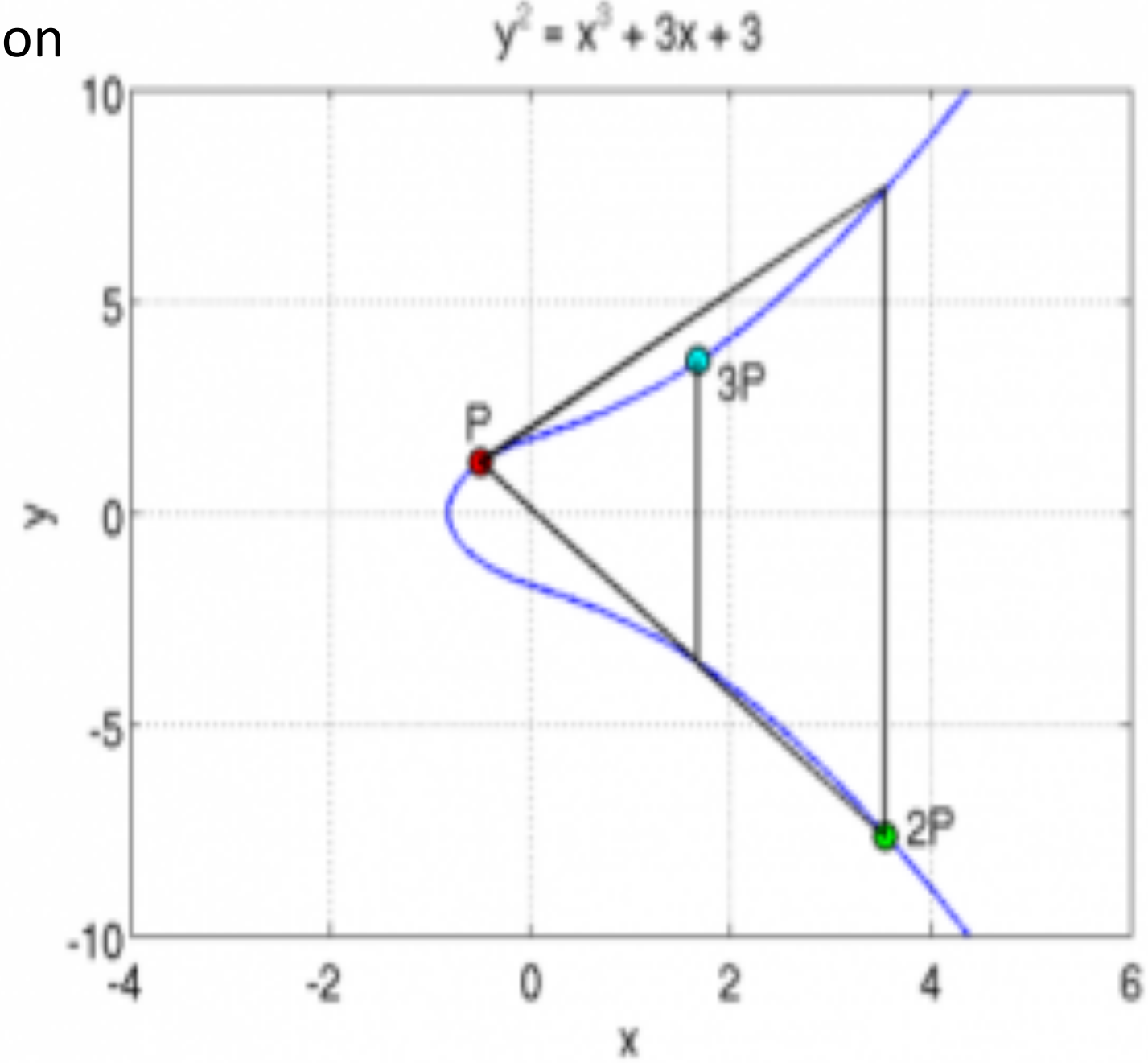
Verification



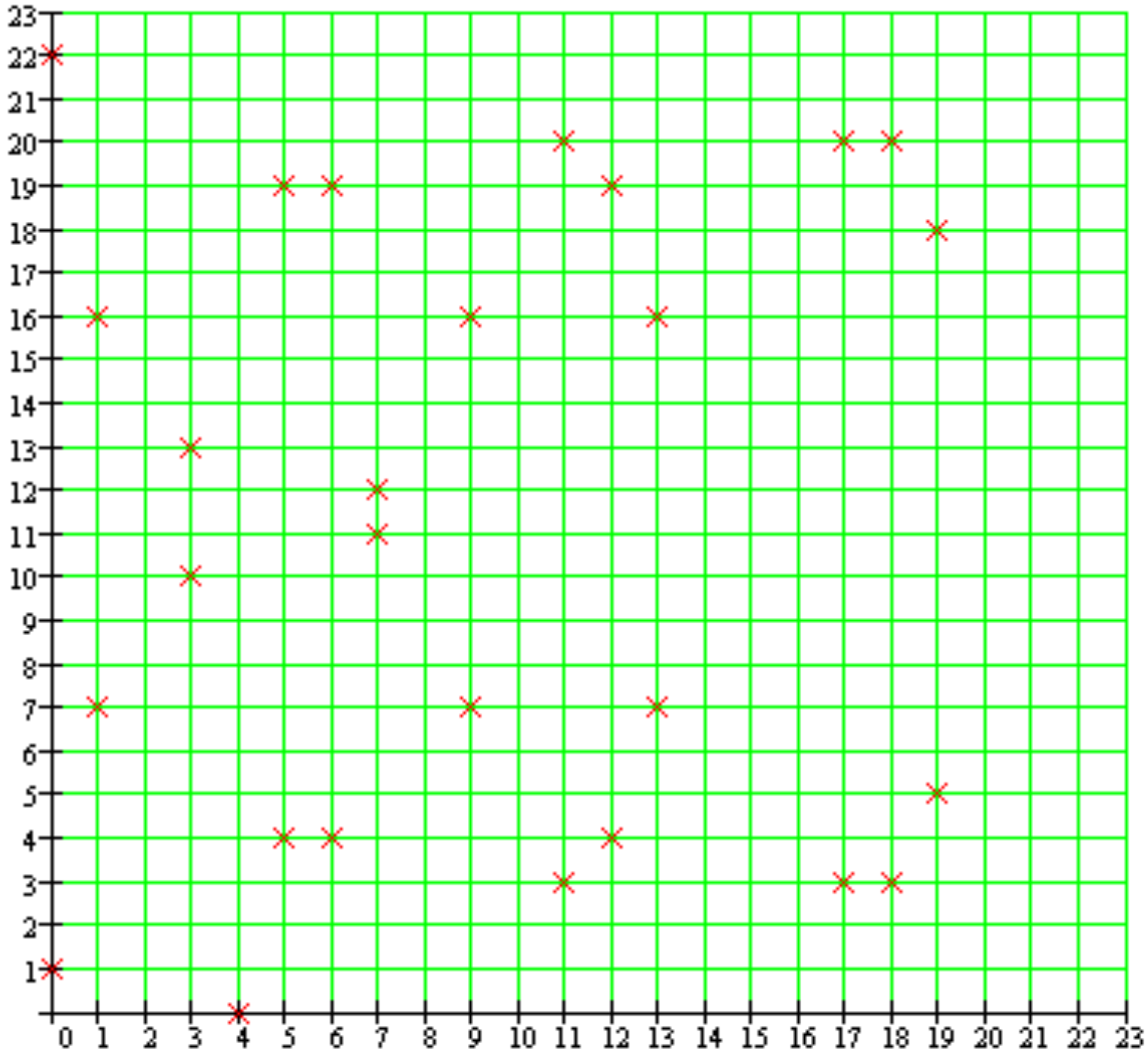
ECC (Elliptic Curves Cryptography) signature function



Discretization



Complementation



Script of transaction and stack

Script	Data	Description
PUSH DATA	PubKey sig	Push data into stack
OP_DUP	PriKey, PubKey, sig	Copy PubKey
OP_HASH160	PubKey hash, PubKey, Sig	Calculate PubKey hash
PUSH DATA	PubKey hash, PubKey hash, PubKey, Sig	Push into PubKey hash
OP_EQUALVERIFY	PubKey, sig	Verify whether they are equal
OP_CHECKSIG	0 or 1	Verify sig

Contents

A short history

- Nick Saab: Conditional Transfer
- E-cash: Centralized server to prevent double spending
- Nakamoto Satoshi:
  - UTXO
- Ethereum:
  - Smart contract
    - Account mechanism: update account status by smart contract
  - Why a cat can block the system

Smart contract of Ethereum

- Types of addresses:
  - Account address
  - Smart contract address
- Triggering methods:
  - Passive
  - To make a transaction to a smart contract address
- Oracle: cannot be triggered by off-chain data
- Interoperability: not multitple token smart contract
- ERC 20 and ERC 721:
  - https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md
  - ICO， Initial Coin Offering

What can smart contracts do?

- P2P transaction
  - To use a transaction to trigger another transaction
  - Multiple signatures
- Turing-complete but not event-driven DApp

To develop a smart contract

- To have an address: https://www.myetherwallet.com/#send-transaction
- To look up the information of a smart contract https://ropsten.etherscan.io/address/0xd6692cd2480acfe287410af6b109369fceee9eea#code
- Solidity: http://solidity.readthedocs.io/en/latest/
- ERC 2.0： https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md
- To debug a smart contract: https://ethereum.github.io/browser-solidity/#optimize=false&version=soljson-v0.4.19+commit.c4cbbb05.js
- To compile a smart contract: http://truffleframework.com/
- Some tooles: https://code.visualstudio.com/docs/?dv=osx

+ Send Ether & Tokens

To Address

0xd6692cd2480acfe287410af6b109369fceee9eea

Amount to Send

10 ROPSTEN ETH

Load Tokens

Send Entire Balance

Gas Limit

47000

+Advanced: Add Data

Generate Transaction

Raw Transaction

```
","to":"0xd6692cd2480acfe28741
0af6b109369fceee9eea","value":
"0x8ac7230489e80000","data":""
,"chainId":3}
```

Signed Transaction

```
0xf86c808509502f900082b79894d6
692cd2480acfe287410af6b109369f
ceee9eea888ac7230489e800008029
a0618905ec178435e37a7d2ee6ad26
```

Send Transaction

Account Address

0x0C9FCD363Ad9777Da801341c38F8f0669A3E11C1

Account Balance

19.99812 ROPSTEN ETH

Transaction History

ROPSTEN ETH (https://ropsten.etherscan.io)

Token Balances

Load Tokens

Remember, you can always view your Balances On https://ropsten.etherscan.io

Not seeing your token? Learn how to Add a Custom Token

Contract Overview		Misc	
ETH Balance:	0 Ether	Contract Creator	0xf64b25b181f9228... at txn 0x643636d48a5c52...
No Of Transactions:	2 txns	Token Tracker	View Tokens (1)

More Options

Transactions | Token Transfers | Contract Code

Are you The Contract Creator? **Verify And Publish**[*New] your Contract Source Code Today!

Switch Back To Bytecodes View | Find Similiar Contracts

```
PUSH1 0x60
PUSH1 0x40
MSTORE
PUSH1 0x04
CALLDATASIZE
LT
PUSH2 0x00cf
JUMPI
PUSH4 0xffffffff
PUSH29 0x0100000000000000000000000000000000000000000000000000000000
PUSH1 0x00
CALLDATALOAD
DIV
AND
PUSH4 0x2c4e722e
DUP2
EQ
PUSH2 0x01d3
JUMPI
DUP1
PUSH4 0x3197cbb6
EQ
PUSH2 0x01f8
```

Solidity

Solidity is a contract-oriented, high-level language for implementing smart contracts. It was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine (EVM).

Solidity is statically typed, supports inheritance, libraries and complex user-defined types among other features.

As you will see, it is possible to create contracts for voting, crowdfunding, blind auctions, multi-signature wallets and more.

Note

The best way to try out Solidity right now is using Remix (it can take a while to load, please be patient).

Translations

This documentation is translated into several languages by community volunteers, but the English version stands as a reference.

clesaege Move the sentence about Transfer event at creation 964104b 12 days ago

3 contributors

185 lines (101 sloc) | 5.26 KB | Raw | Blame | History

Preamble

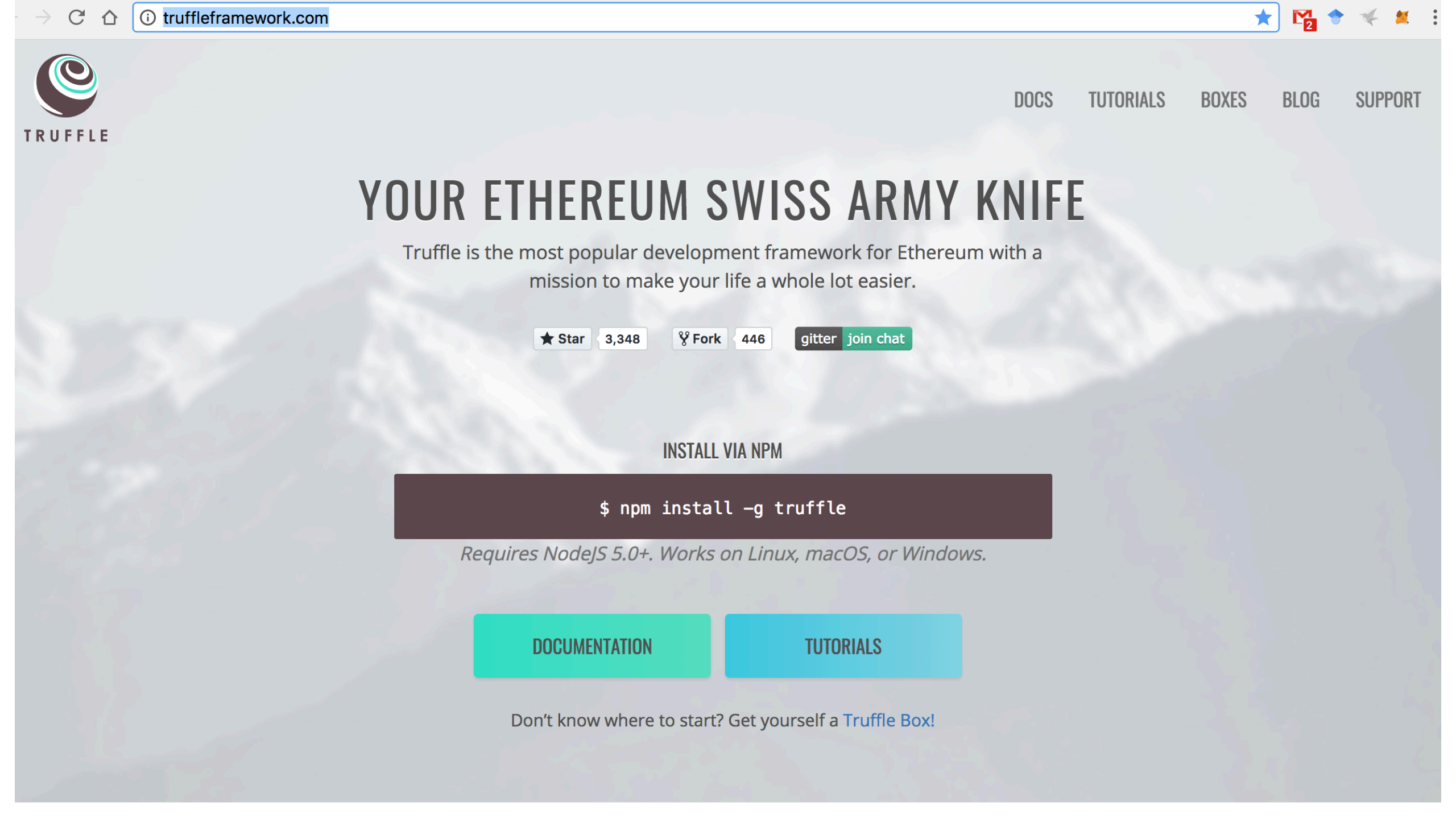
```
EIP: 20
Title: ERC-20 Token Standard
Author: Fabian Vogelsteller <fabian@ethereum.org>, Vitalik Buterin <vitalik.buterin@ethereum.org>
Type: Standard
Category: ERC
Status: Accepted
Created: 2015-11-19
```

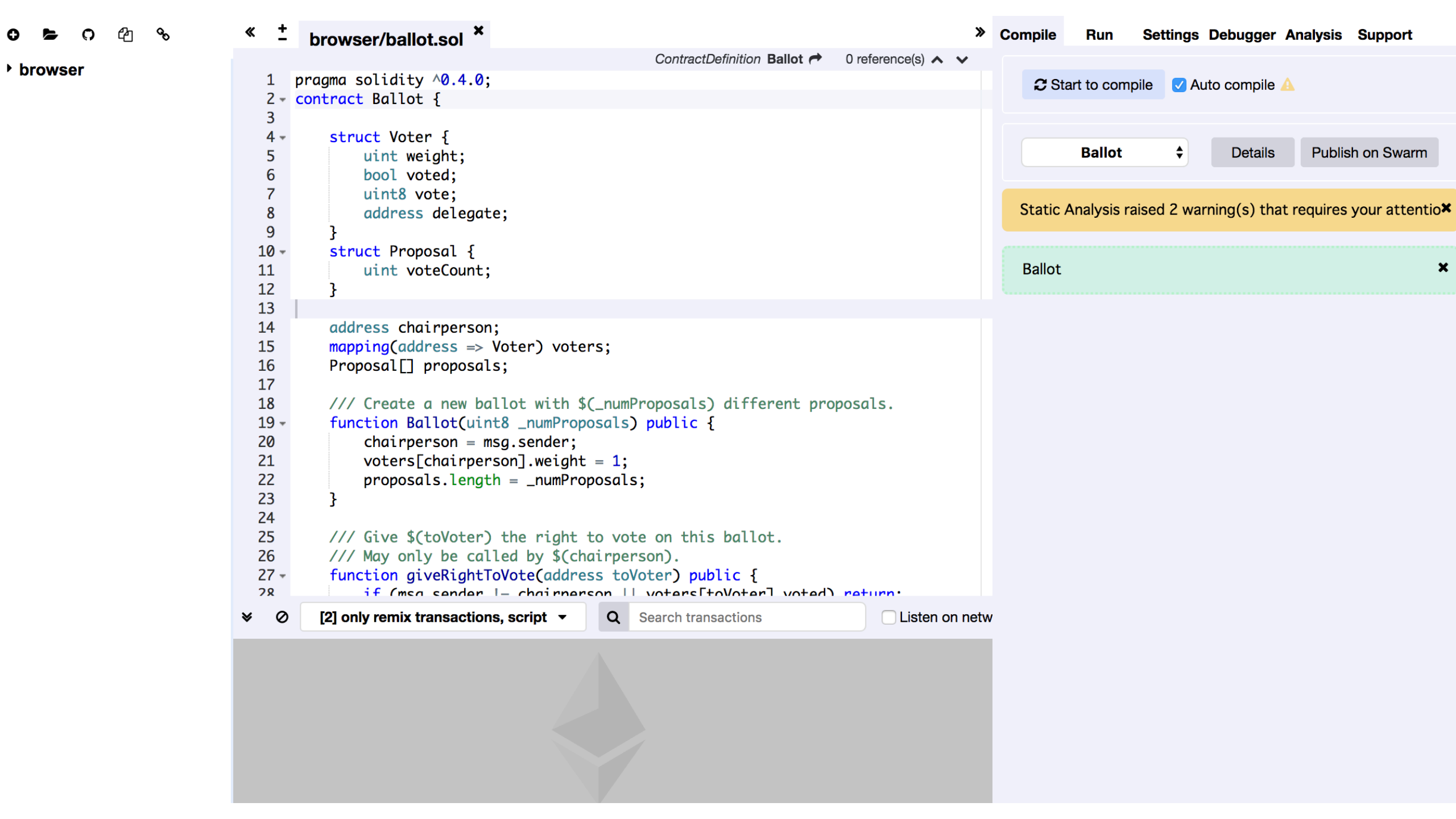
Simple Summary

A standard interface for tokens.

Abstract

The following standard allows for the implementation of a standard API for tokens within smart contracts. This standard provides basic functionality to transfer tokens, as well as allow tokens to be approved so they can be spent


truffleframework.com
TRUFFLE
DOCS
TUTORIALS
BOXES
BLOG
SUPPORT
YOUR ETHEREUM SWISS ARMY KNIFE
Truffle is the most popular development framework for Ethereum with a mission to make your life a whole lot easier.
Star 3,348
Fork 446
gitter join chat
INSTALL VIA NPM
$ npm install -g truffle
Requires NodeJS 5.0+. Works on Linux, macOS, or Windows.
DOCUMENTATION
TUTORIALS
Don't know where to start? Get yourself a Truffle Box!


browser
browser/ballot.sol
ContractDefinition Ballot
0 reference(s)
1 pragma solidity ^0.4.0;
2 contract Ballot {
3
4 struct Voter {
5 uint weight;
6 bool voted;
7 uint8 vote;
8 address delegate;
9 }
10 struct Proposal {
11 uint voteCount;
12 }
13
14 address chairperson;
15 mapping(address => Voter) voters;
16 Proposal[] proposals;
17
18 /// Create a new ballot with $(_numProposals) different proposals.
19 function Ballot(uint8 _numProposals) public {
20 chairperson = msg.sender;
21 voters[chairperson].weight = 1;
22 proposals.length = _numProposals;
23 }
24
25 /// Give $(toVoter) the right to vote on this ballot.
26 /// May only be called by $(chairperson).
27 function giveRightToVote(address toVoter) public {
[2] only remix transactions, script
Search transactions
Listen on netw
Compile
Run
Settings
Debugger
Analysis
Support
Start to compile
Auto compile
Ballot
Details
Publish on Swarm
Static Analysis raised 2 warning(s) that requires your attentio
Ballot

Essence of smart contracts

- Programs in the Internet of Information to program interaction of information
  - Information interaction: WEB
  - Mobile apps and enterprise information system: Apps、ERP
  - Many development languages and their ecosystem
- Programs in the Internet of Values to progrma interaction of values
  - Blockchain: platform
  - DApp
  - Languages: Solidity

The problems of current smart contracts

- Usability and scalability of blockchains:
  - Cannot bear big program such as Facebook, Amazon
  - No parallel computation
  - Network communication pressure : 5G may be a solution
- Operability of blockchains: Islands of blockchains
  - Values cannot be interacted in a smart contract
  - Many off-chain values have not mapped on chains
  - Cannot read off-chain data
  - Different smart contract cannot be combined: still not event-driven

Future of smart contracts

- The tools of the Internet of Information will be transplanted to the Internet of Values
  - OS: public chains and VM
  - Development ecosystem: languages and tools
  - Application ecosystem
- The Internet of Values:
  - Abstraction of trading scenarios: abstraction of financial attributes
  - The emergence of a new trading model: encryption finance
- Internet of Information + Internet of Values
  - Deep integration of information interaction and value interaction
  - New business scenario: automated production, distribution and management
  - Fully automated human collaboration: society is an automated machine

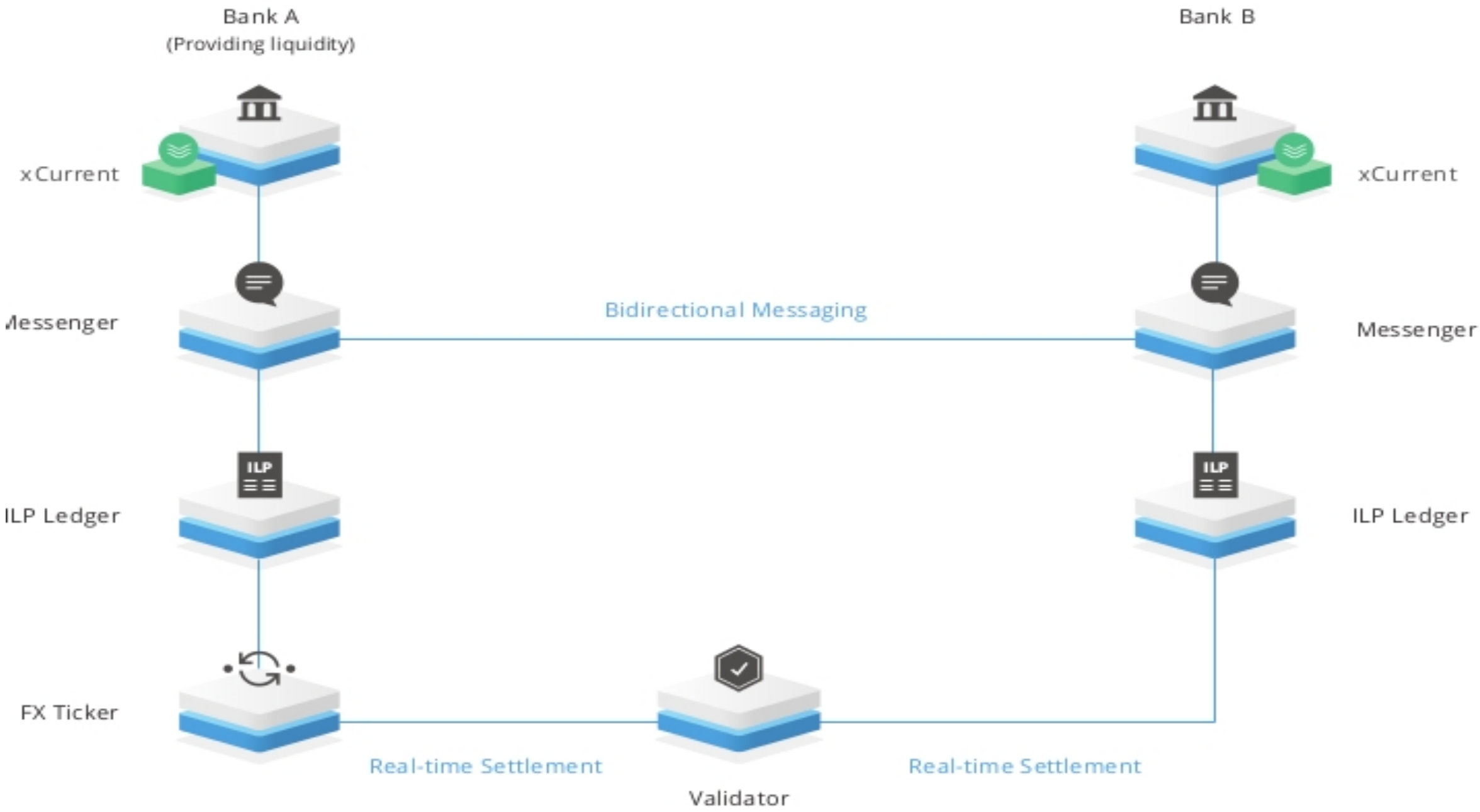
Contents

Comparison of Cross-chain Projects

	Ripple	BTC-Relay	Polkadot	Cosmos	Lightning network	WanChain	FUSION
Cross-Chain	Notaries	Side-chain	Relay	Relay	Hash-lock	Distributed private key control	Distributed private key control
Consensus	Notaries	Dependent on main chain	Asynchronous BFT	Tendermint	None	PoS	PoS+PoW
Private key mechanism	User control	User control	User control	User control	User control	Bookkeeping nodes' control	Bookkeeping nodes' control
Multiple token smart contract	None	None	None	None	None	Yes	Yes
Parallel computing	None	None	None	None	None	None	Yes
Multiple triggering mechanism	None	None	None	None	None	None	Yes
Off-chain data support	None	None	None	None	None	None	Yes
Application	Across country payment	BTC/ETH transfer	Cross-chain communication	Cross-chain communication	Atomic transfer	Banking service	Finance for the Internet of Values

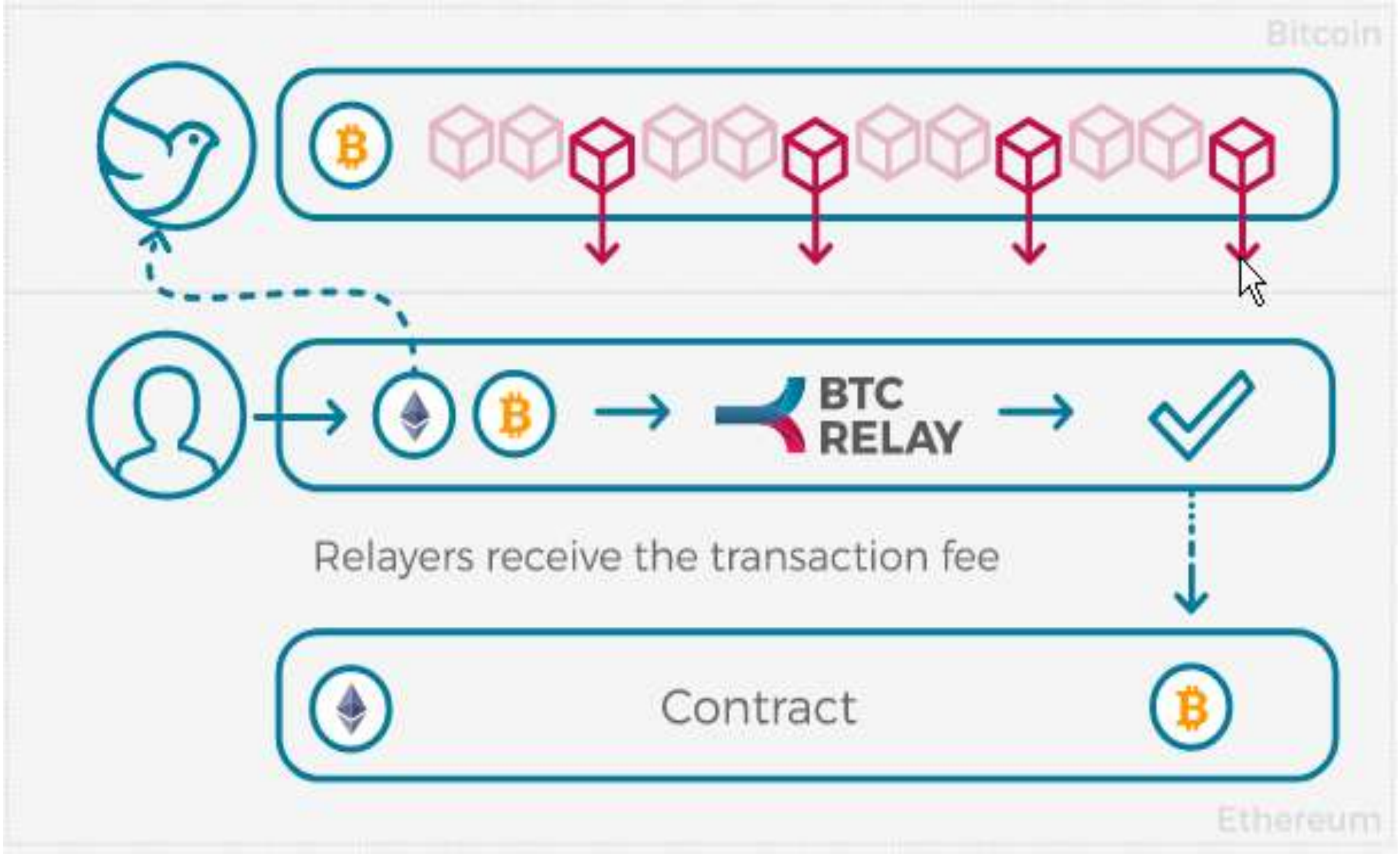
Notary schemes: Ripple Interledger Protocol

- In 2012, the Ripple Labs proposed the Interledger agreement to connect different ledgers and achieve synergies between them
- The Interledger protocol enables two different accounting systems to freely exchange currency with each other through third-party "connectors" or "validators.”
- The accounting system does not need to trust the "connector," because the protocol uses a cryptographic algorithm to create funds for the two accounting systems with a connector.
- The agreement removes the trust required by transaction participants and the connector does not lose or steal money.


Bank A
(Providing liquidity)
Bank B
xCurrent
xCurrent
Messenger
Bidirectional Messaging
Messenger
ILP
ILP
ILP Ledger
ILP Ledger
FX Ticker
Real-time Settlement
Validator
Real-time Settlement

Side-chains

- Sidechain is a new type of blockchain based on the anchoring of a token of the original chain
  - Just as the dollar is anchored to gold
- BTC Relay
  - Connects the Ethereum network with the Bitcoin network through the use of Ethereum's smart contracts to enable users
  - To validate Bitcoin transactions on Ethereum
  - It creates a small version of the Bitcoin blockchain via Ethereum smart contracts
  - However, it is very difficult for the sidechain to establish a Cross-Chained smart contract on it

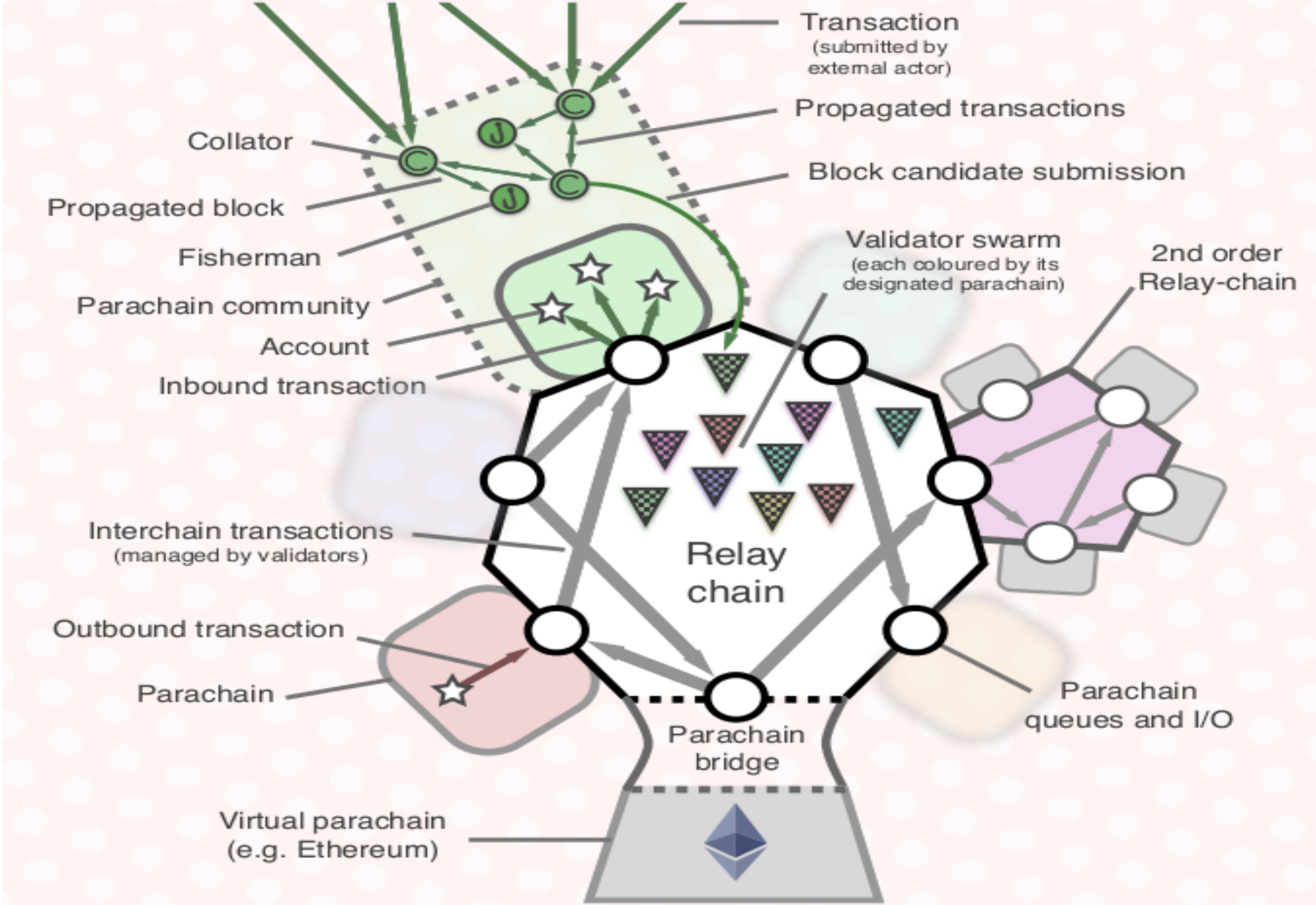

Bitcoin
BTC
RELAY
Relayers receive the transaction fee
Contract
Ethereum

Relays

- Relay chain (relay-chain) technology can be the original chain of tokens into the multi-signature control of the original chain address
- Temporarily locked
- The results of the transactions on the relay chain will be voted on by these signers to decide whether or not to take effect
- It also introduced the role of phishers in reporting and monitoring transactions

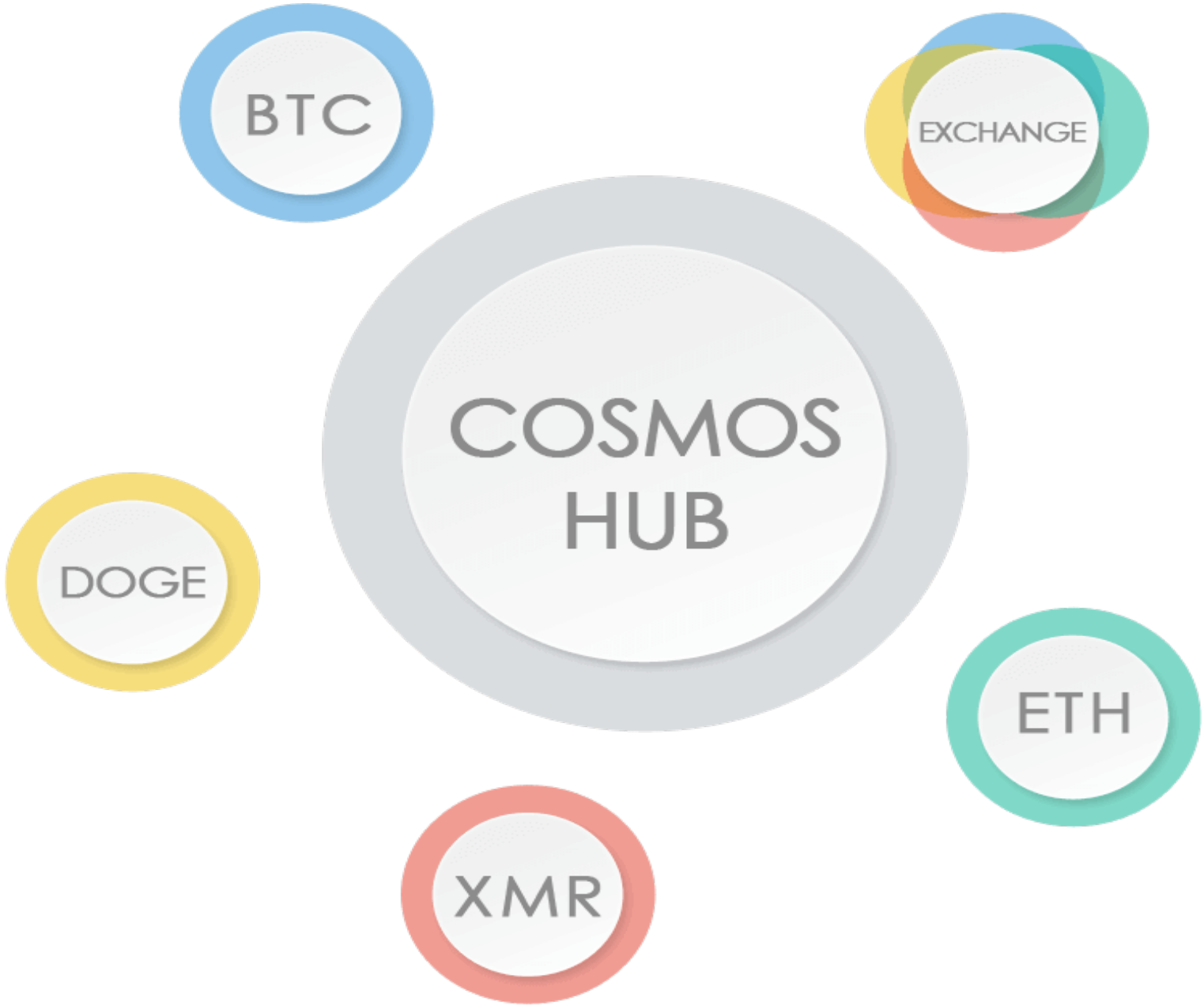
Polkadot

- Polkadot is a public chain developed by the original core developers of Ethereum.
- Polkadot plans to integrate the private chain / affiliate chain into the consensus chain of the public chain
- While preserving the original data privacy and licensing features of the private / affiliate chain
- Polkadot view, the other blockchain are parallel chain
- Polkadot is currently based in Ethereum, interconnecting with private chains and upgrading to other public-chain networks


Transaction
(submitted by external actor)
Propagated transactions
Block candidate submission
Collator
Propagated block
Fisherman
Parachain community
Account
Inbound transaction
Validator swarm
(each coloured by its designated parachain)
2nd order Relay-chain
Interchain transactions
(managed by validators)
Relay chain
Outbound transaction
Parachain
Parachain queues and I/O
Parachain bridge
Virtual parachain
(e.g. Ethereum)

COSMOS

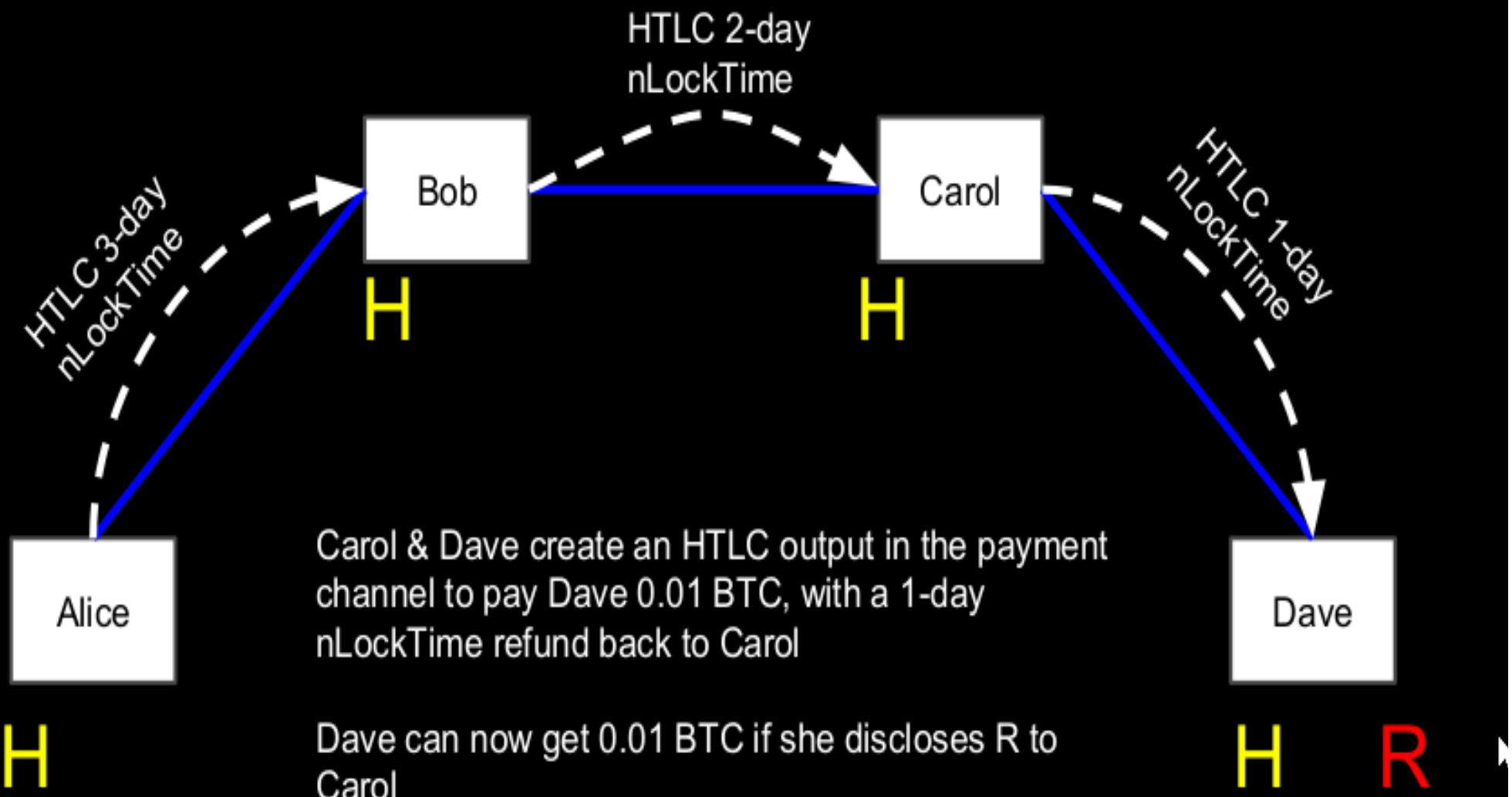
- Cosmos is a heterogeneous network supported by the Tendermint team that supports Cross-Chain interactions.
- It uses the Tendermint consensus algorithm, which is a similar practical Byzantine consensus engine with high performance and consistency
- Under its strict responsibility guarantee, it can prevent malicious actors from improperly operating.
- The first space on Cosmos is called Cosmos Hub.
  - The Cosmos Hub Center is a multi-asset-value proof cryptocurrency network
  - It enables network changes and updates through a simple management mechanism and can be expanded by connecting to other spaces.
- Cosmos network center and each space can communicate through the inter-block chain (IBC) protocol.


BTC
EXCHANGE
COSMOS
HUB
DOGE
ETH
XMR

Hash-Locking

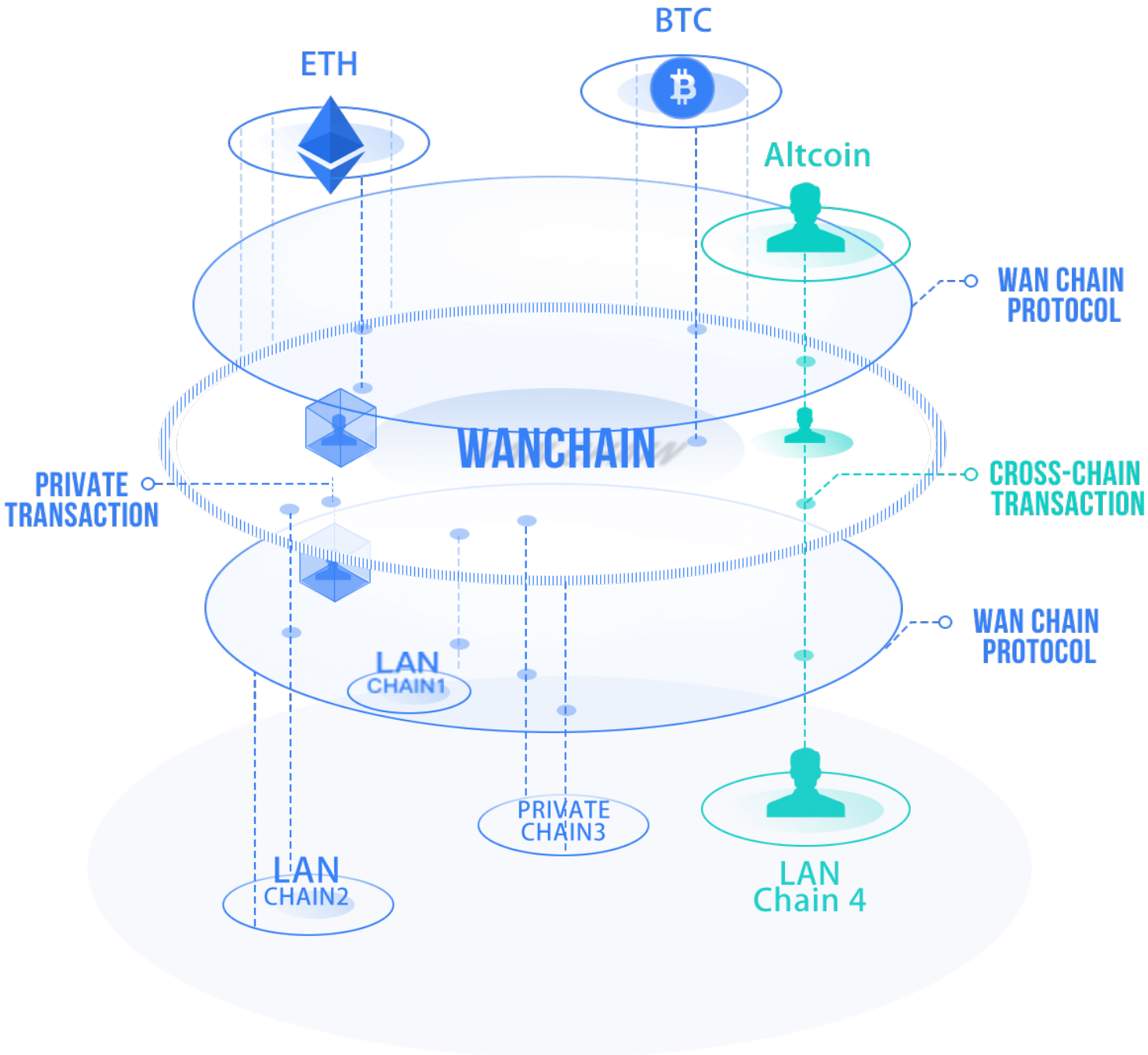
- Multiple payment channels
- The HTLC hash locking technique
  - Alice and Bob can conclude that the protocol will lock Alice's 0.1 BTC before time T
    - T is expressed in terms of some blockchain height in the future
  - Bob can get this 0.1 BTC if Bob is able to present Alice with an appropriate R (called a secret) so that the hash of R equals the pre-agreed value of H (R)
  - if Bob still does not get past time T To provide a correct R, this 0.1 BTC will automatically thaw and return Alice.
- The lightning network does not attempt to address the issue of a single payment
- No multiple token smart contracts

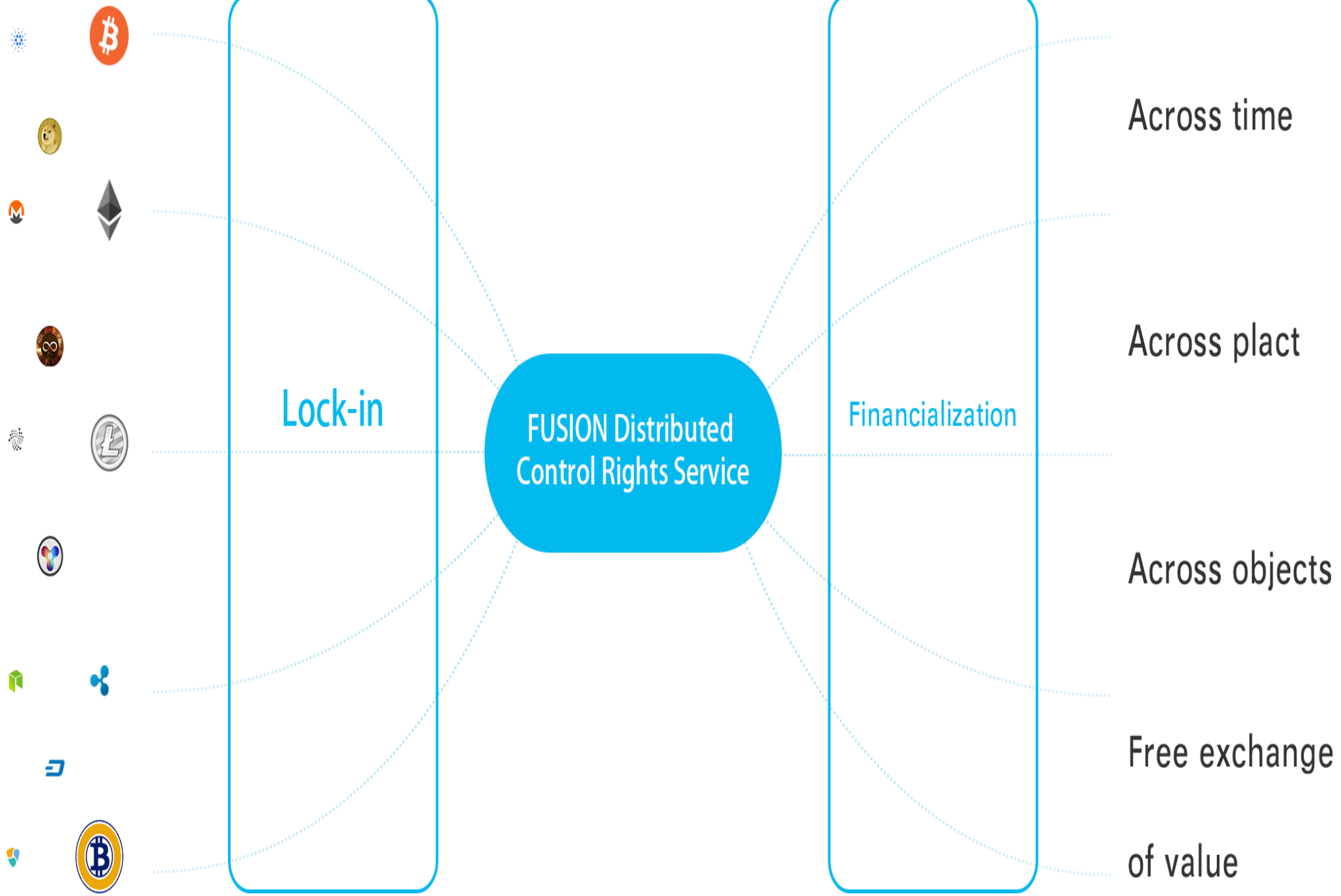
Bitcoin Lightning Network



Distributed private key control techniques

- Distributed control of private keys
- Through the distributed way to complete the connection and value exchange of different blockchain ledger
- Sidechains of all other chains
- Have multiple token smart contract
- Risks are high


BTC
ETH
Altcoin
WAN CHAIN
PROTOCOL
WANCHAIN
PRIVATE
TRANSACTION
CROSS-CHAIN
TRANSACTION
WAN CHAIN
PROTOCOL
LAN
CHAIN1
PRIVATE
CHAIN3
LAN
CHAIN2
LAN
Chain 4


Lock-in
FUSION Distributed
Control Rights Service
Financialization
Across time
Across plact
Across objects
Free exchange
of value

The future of cross-chain technology

- Interoperability will be the infrastructure of every chain
- Multiple signature address to control address will use mobiles to do proof of siganiture
- Tokens locked in a distributedly controlled address will mainly regarded as financial assets
- Cryptofinance will thrive on cross-chain platforms

Contents

From Turing-complete to function-complete

- Turing-complete programs focus on a program
  - Have ifs and whiles
- Applications focus on scenarios
  - Have resources of calculation, storage and communication
- Data storage is a huge bottleneck of blockchains
- Communication is not easy for DApps
- Cross-chain functions are much needed
- Scalability: TPS
- Flexibility of consensus, such as range of nodes of consensus

Data storage

- File storage is just first step
  - IPFS
- High-performance data such as being easy to write and read will be more important
- Safety and privacy
- Data market: transaction of data
- Behavior data and KYC
- Projects: Endereum, filecoin, IPFS Figtoo, MaidSafe, Enigma, Lambda Project, Penda, Storj, sia, decent, steemit, Tieron, LAFS, Eth Swarm, BigchainDA + IPDB, IOTA, NeoFS

Future of data storage

- File storage is just first step
  - IPFS
- High-performance data such as being easy to write and read will be more important
- Safety and privacy
- Data market: transaction of data
- Behavior data and KYC

Communication

- Address is everything
  - Chat
  - Audio
  - Video
  - Call center
  - File share
  - Storage
  - DApps
  - KYC

Contents

Scalability

- Sharding:
  - Grouping
  - Local data and public data
- Layers:
  - Computing
  - Bookkeeping
- Subchains:
  - Storage
  - Routing

Contents

Defects of current blockchains

- Scalability
  - TPS is not high enough
  - Main reason: Internet speed
  - Secondary reason: consensus technology
- Interoperability
- Usability
- Flexibility

Consensus

- Types
  - PoW: Bitcon
  - PoS: Ethereum
  - PBFT: NEO
  - DPoS
  - DPoS+PBFT: EOS
  - DAG
- Problems
  - TPS
  - Inflexibility

Contents

The future of blockchain technology--generations of blockchains

- Blockchain 1.0: Programmable currency – Bitcoin
- Blockchain 2.0:
  - Turing-complete
  - Distributed Program
  - Everybody can issue currency
- Blockchain 3.0:?
  - Function-complete
  - Everybody can contact and contract with others on distributed network

Function-complete

- Communication: embedded router services
- Interoperability: embedded cross-chain technology
- Storage: high performance reading and writing
- Unlimited scalability
- Flexibility

Main future technology

- Consensus: variety and flexibility
  - Flexibility:
    - To pay for letting more people know
    - The greater the consensus reach, the more people need to participate in verification and data synchronization.
    - To assign parameters of underlying blockchain
- Chain net: multiple chains cooperation
  - Side chains
  - subchains
- Layers
- Groups
- Cooperation chains
- Cross-chains: from relay to custody
- Anti-quantum attack
- Anti-AI attack

Smart contract

- Turing-complete to function-complete
- Cross-chain
- Cross-datasource
- Unlimited scalability
- Ability to assign the parameters of blockchain platform

Thanks for your contribution to the human civilization by spending time to come here to learn!

- Q&A