



UNIVERSITY OF
TORONTO



Le génie pour l'industrie



UiO • University of Oslo

Deconstructing Blockchains: Concepts, Systems, and Insights

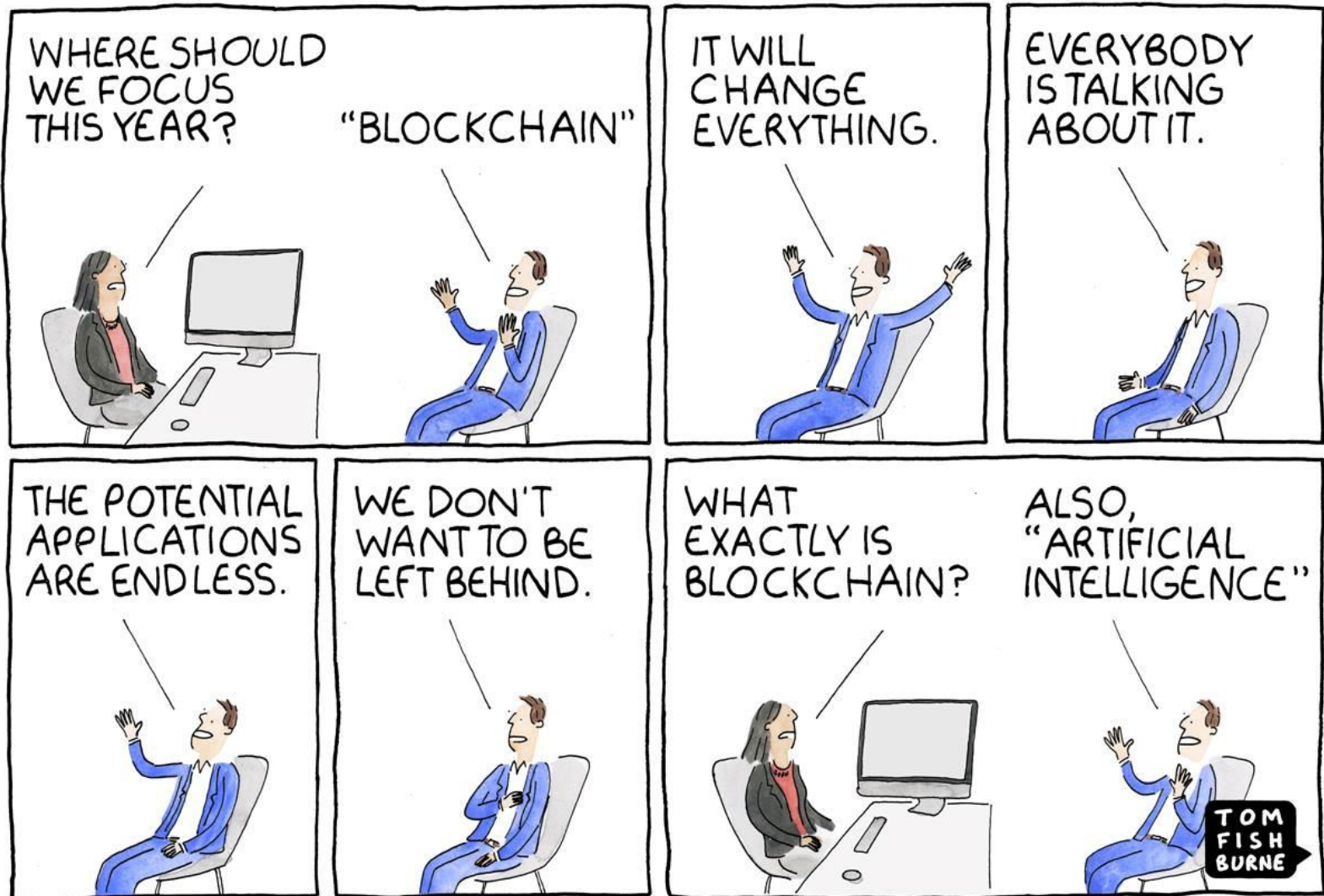
Link to our companion papers:

<http://msrg.org/papers/bcbi-tr>

<http://heim.ifi.uio.no/~romanvi/debunking-bc-myths.html>

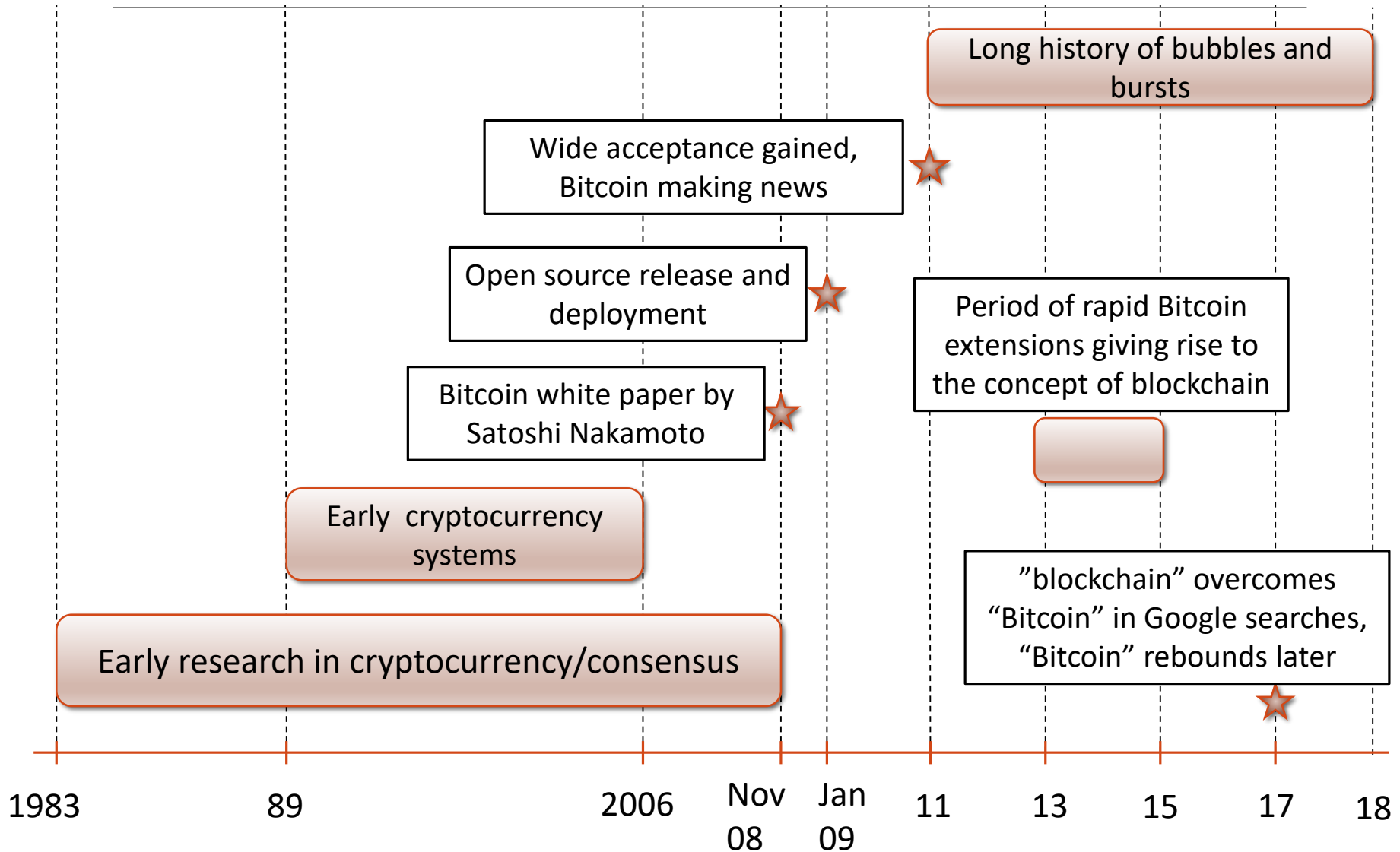
BY KAIWEN ZHANG,
ROMAN VITENBERG,
HANS-ARNO JACOBSEN

Understanding Blockchains



© marketoonist.com

Historical perspective



Status today: the Blockchain hype

Bitcoin gold rush

15 percent of top global banks rolled out full-scale commercial blockchain products in 2017

- Goldman Sachs alone investing half a billion USD

Blockchain became national storage technology in Estonia

Blockchain storage strategy and regulations in Netherlands

Microsoft declares “blockchain” as a “must win” technology for the Azure platform and business

IBM unveils new blockchain-oriented strategy; opens a new department

Dedicated labs and education programs in blockchain engineering around the globe

- A master program in blockchain engineering at the University of Delft
- A new course at the University of Oslo, TUM, Cornell, and many others

Hottest topic at many societal, industrial, and academic conferences

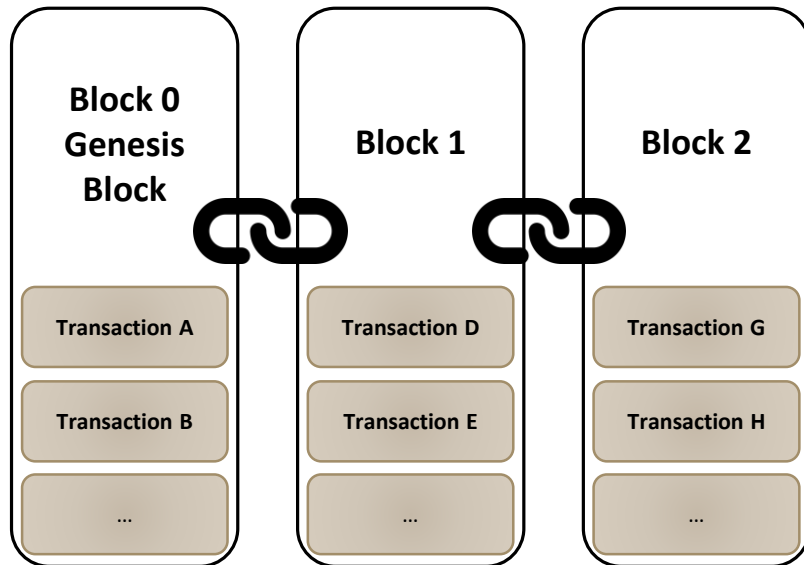
Blockchain 101



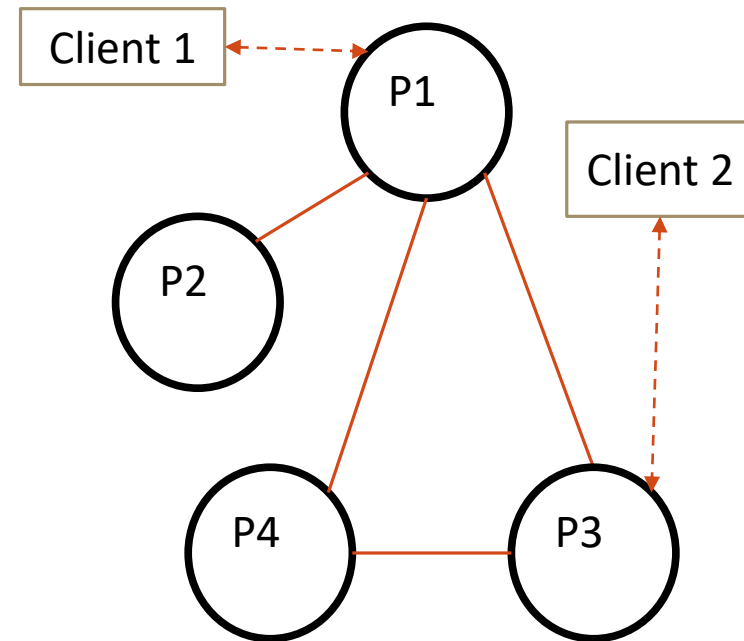
Distributed Ledger Technology (DLT)

BLOCKCHAIN

Blockchain data structure (replicated at every peer)



Peer-to-Peer network



Blockchain 101

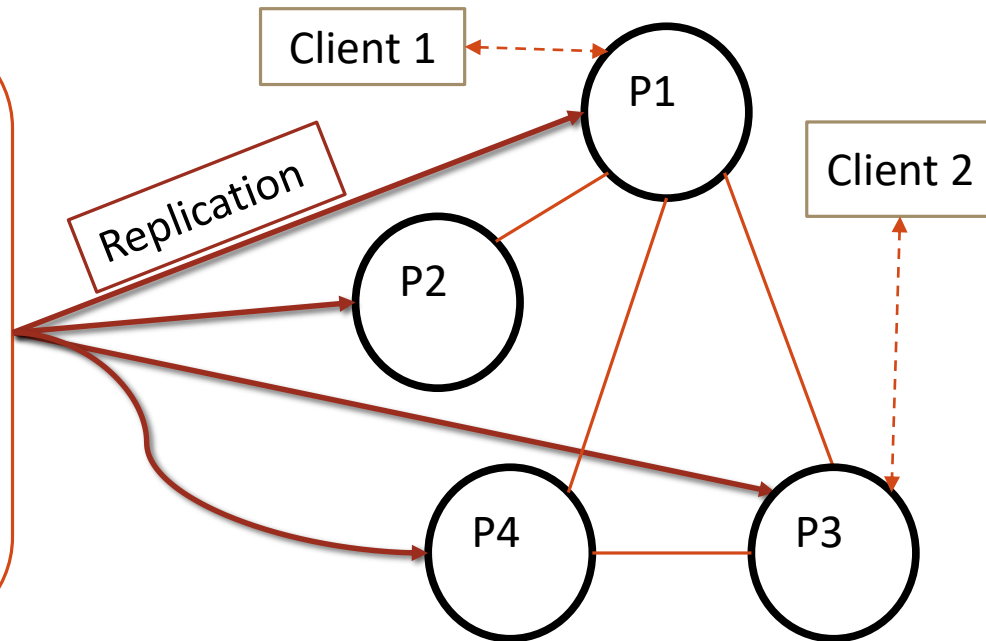
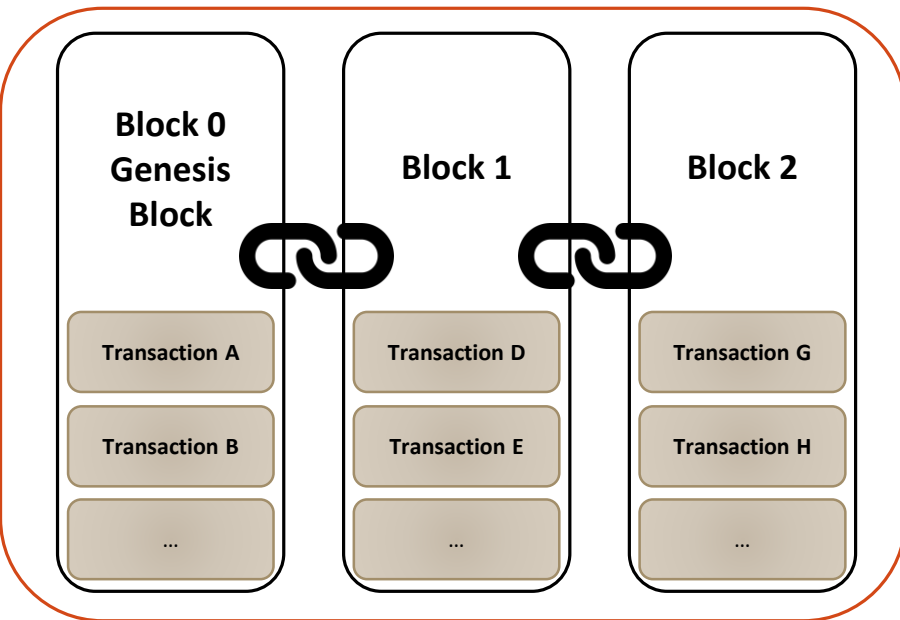


Distributed Ledger Technology (DLT)

BLOCKCHAIN

Blockchain data structure (replicated at every peer)

Peer-to-Peer network



Blockchain 101

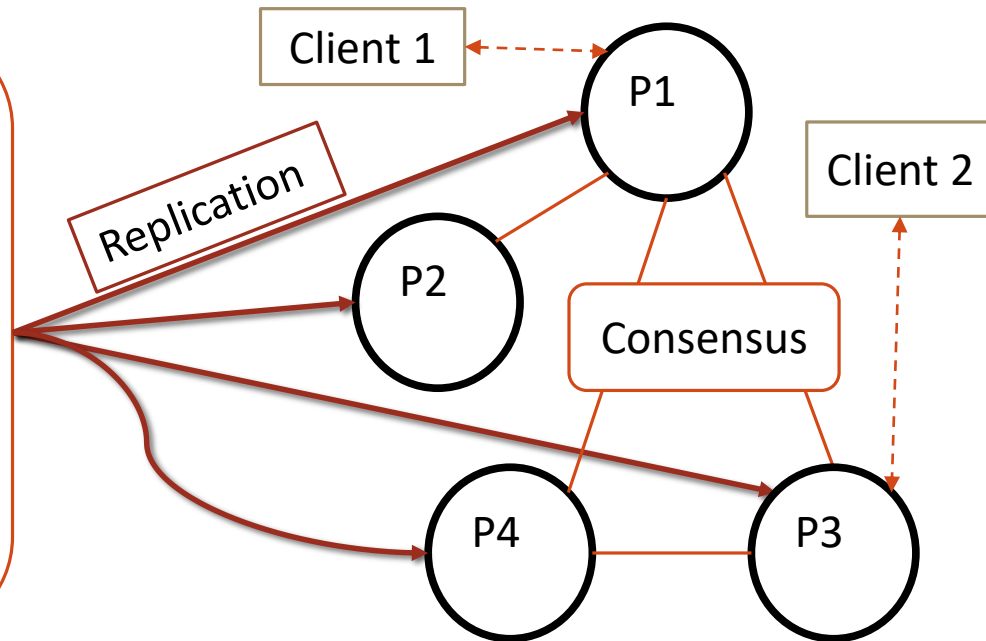
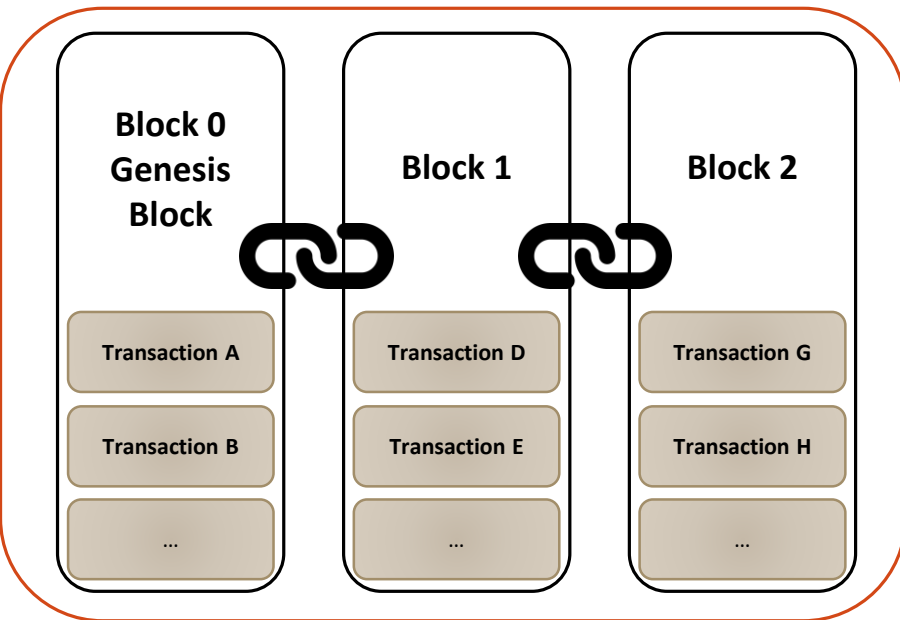


Distributed Ledger Technology (DLT)

BLOCKCHAIN

Blockchain data structure (replicated at every peer)

Peer-to-Peer network



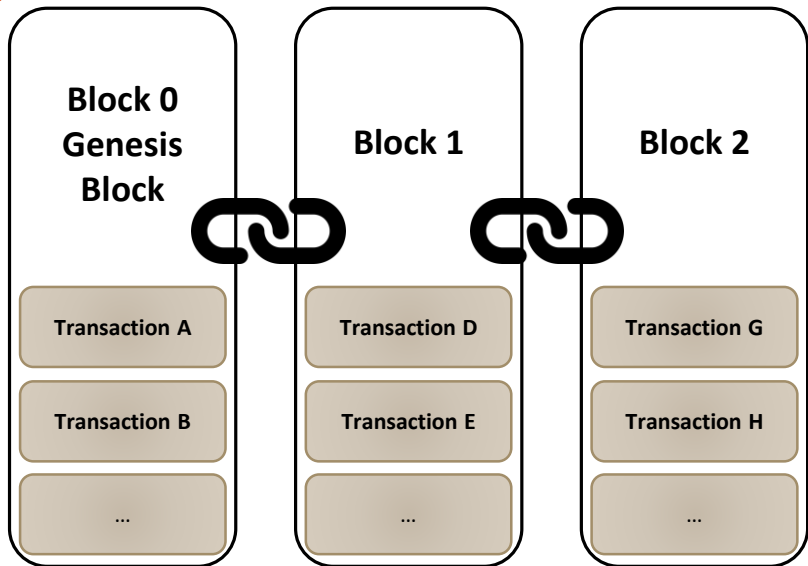
Blockchain 101



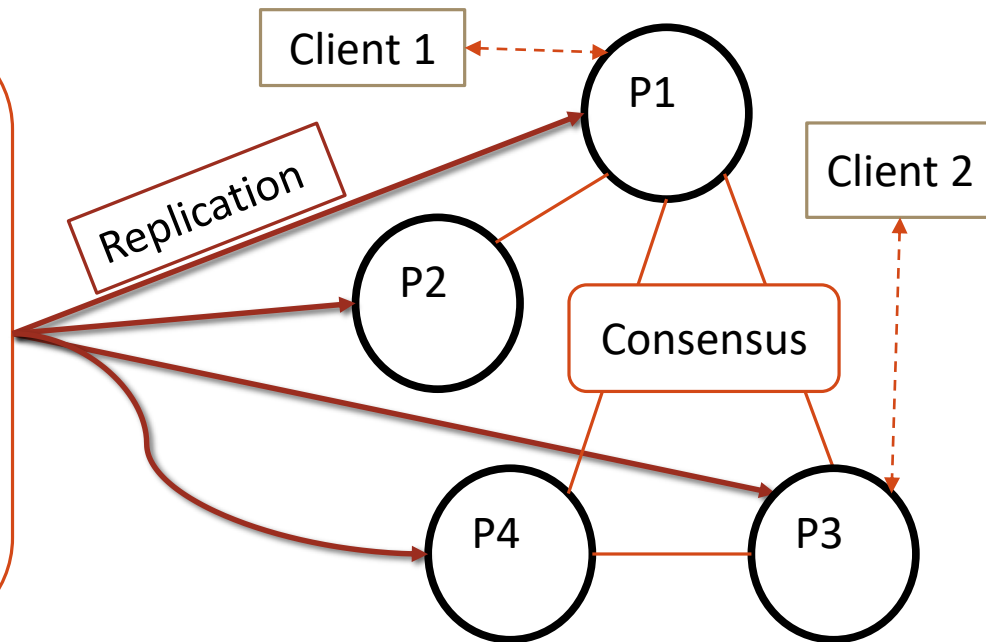
Distributed Ledger Technology (DLT)

BLOCKCHAIN

Blockchain data structure (replicated at every peer)



Peer-to-Peer network

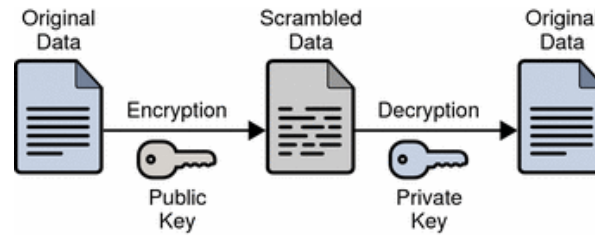


Cryptography is used to...

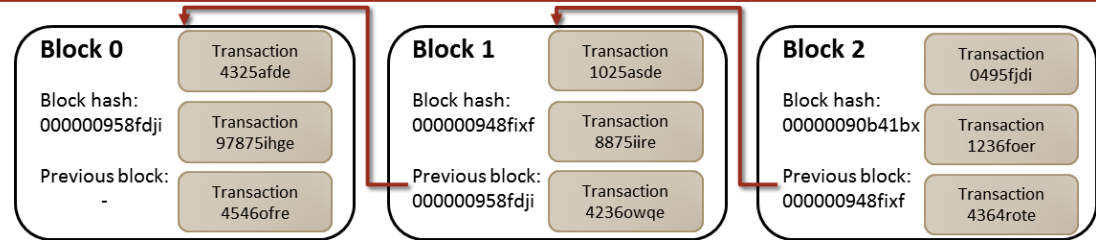
*...encrypt data, prevent modification, insert new blocks, execute transactions, and query...
the distributed ledger*

Cryptography and security in blockchains

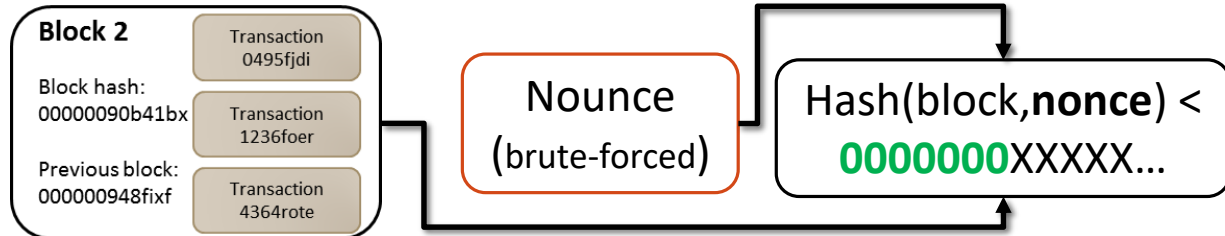
Encrypt data:
Public Key Encryption



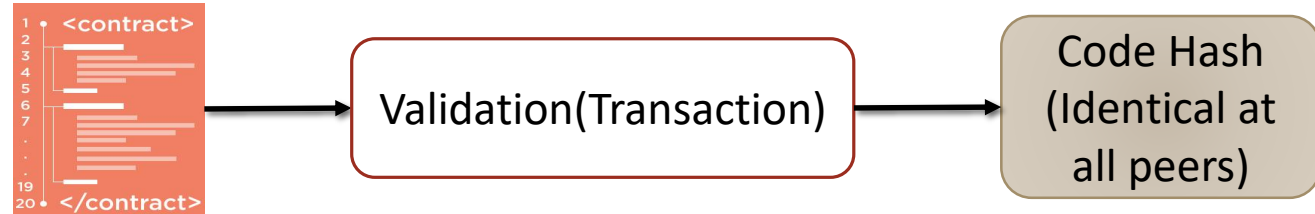
Prevent modification:
Hashed Linked List



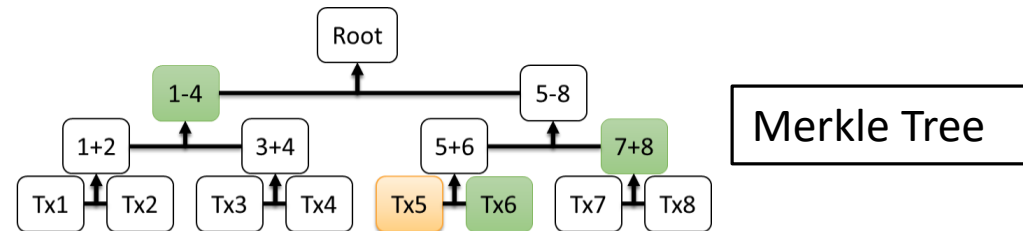
Insert new blocks:
Proof-of-Work



Execute transactions:
Smart Contracts



Query the blockchain:
Simple Payment Verification

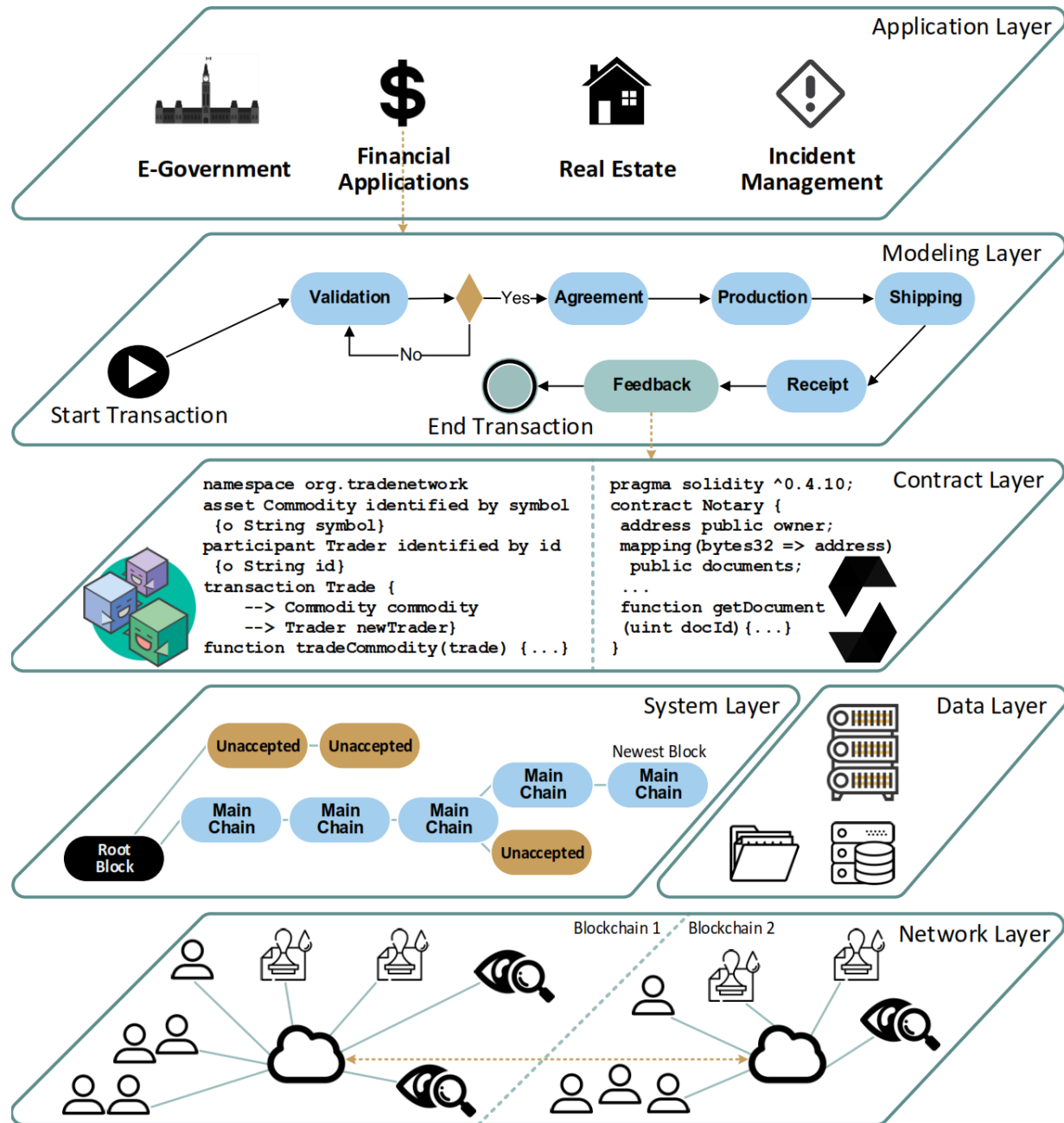


Blockchain Reference Architecture

This vision diagram encompasses all aspects related to blockchain technologies.

Upper layers capture application semantics and their implementation.

Lower layers are concerned with technical system details.



Blockchain vs. Distributed DB

Blockchains maintain a log (aka a ledger) of all transactions since the start of deployment

- e.g. in Bitcoin, there is no direct record of the current state

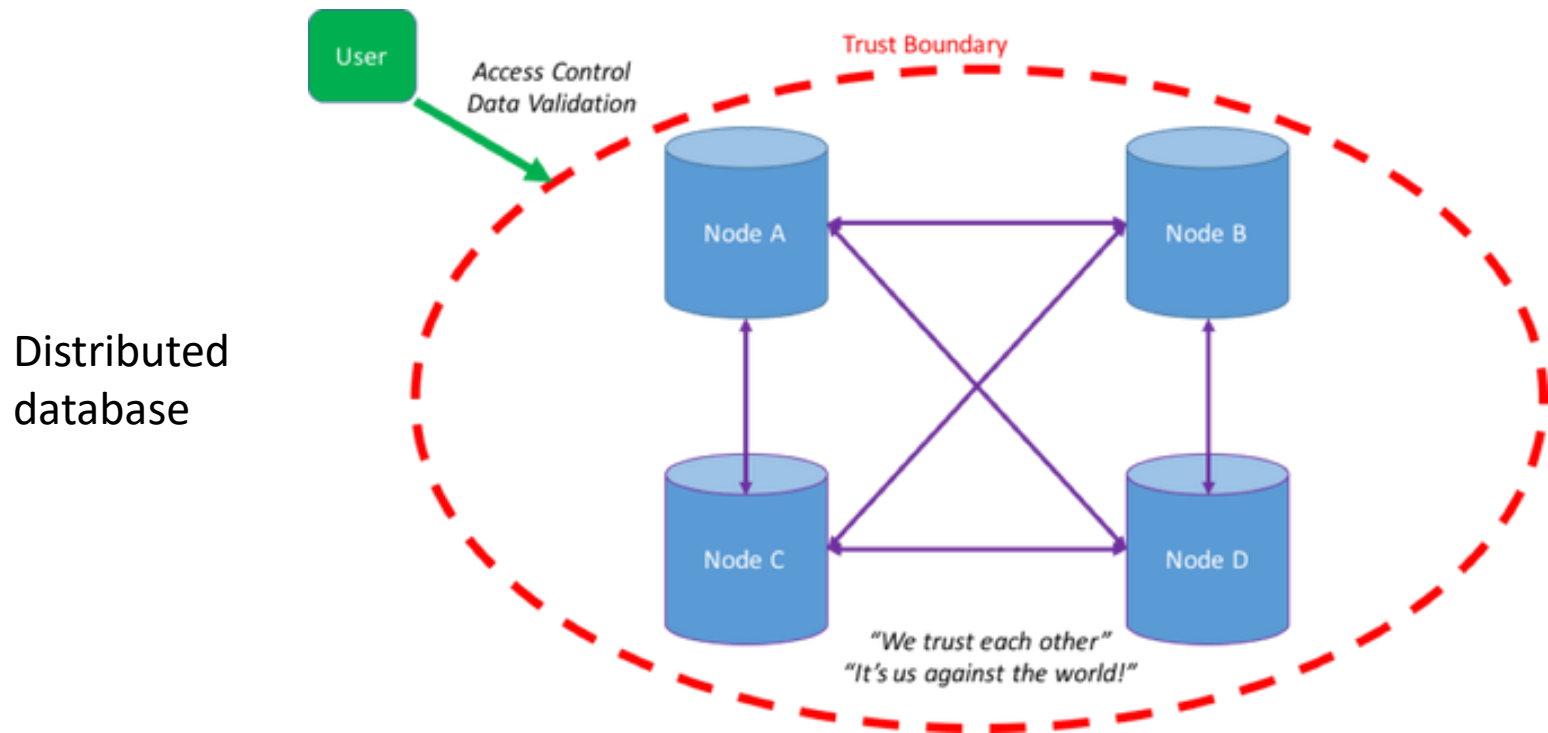
The trust model is fundamentally different

Blockchain vs. Distributed DB

Blockchains maintain a log (aka a ledger) of all transactions since the start of deployment

- e.g. in Bitcoin, there is no direct record of the current state

The trust model is fundamentally different



Distributed
database

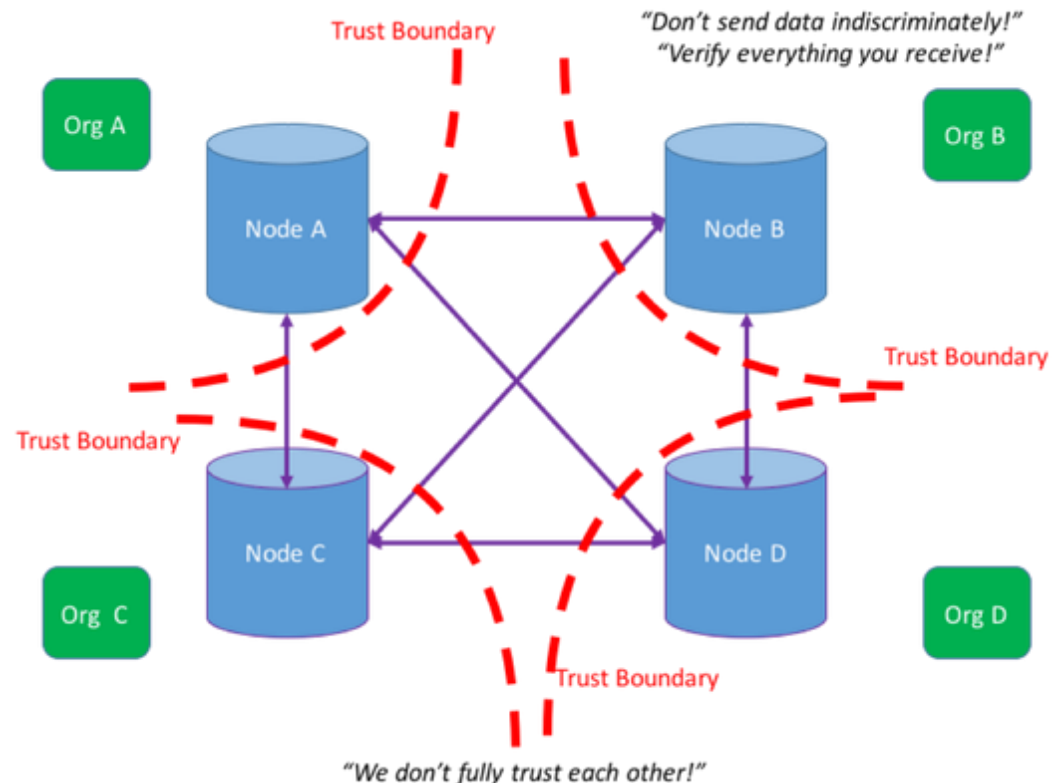
Blockchain vs. Distributed DB

Blockchains maintain a log (aka a ledger) of all transactions since the start of deployment

- e.g. in Bitcoin, there is no direct record of the current state

The trust model is fundamentally different

Blockchain /
distributed ledger



Outline

Session 1: Foundations

- Concepts: Byzantine Consensus, Mining, Proof-of-Work, Smart Contracts
- Original system: Bitcoin

Session 2: Beyond Bitcoin

- Smart contracts
- Platforms: Ethereum, Hyperledger

Session 3: Research

- System insights
- Research directions

Session 4: Hands-on tutorial on Ethereum

- Smart contract development and deployment
- Tools for deploying and managing Ethereum



Blockchain Concepts

DEFINITIONS

BITCOIN OVERVIEW

Bitcoin vs. Blockchain

Bitcoin is a specific system

- Design
- Open-source implementation
- Deployment
- There are alternative cryptocurrency systems (some of which are spawn-offs) but they are not Bitcoin

Blockchain is ambiguous: can be the data structure used in Bitcoin or a separate concept

A guiding design principle/paradigm

- Not even a standard
- Generalization of Bitcoin (In what direction?)
- Hundreds of implementations
- Ethereum alone has hundreds of proprietary deployments in addition to the main public deployment

What is a blockchain-based distributed ledger?

- ✓ *An append-only log* storing transactions
- ✓ Comprised of *immutable* blocks of data
- ✓ *Deterministically* verifiable (using the *blockchain* data structure)
- ✓ Able to execute transactions *programmatically* (e.g., Bitcoin transactions and smart contracts)
- ✓ *Fully replicated* across a large number of peers (called miners in Bitcoin)
- ✓ *A priori decentralized*, does not rely on a third party for trust

Blockchain and the land of ambiguities

Definition 1: a system that uses the blockchain structure of Bitcoin but extends the functionality

- Extended business logic
- Different consensus protocol

Definition 2: a system that maintains a chain of blocks

- Could be a structure other than that of Bitcoin

Definition 3: a system that maintains a ledger with all transactions

- Not necessarily stored as a chain of blocks
- Aka distributed ledger systems

Definition 4: a system with distributed non-trusting parties collaborating without a trusted intermediary

Definition 5: a system that uses smart contracts

Blockchain and the land of ambiguities

Definition 1: a system that uses the blockchain structure of Bitcoin but extends the functionality

- Extended business logic
- Different consensus protocol

Definition 2: a system that maintains a chain of blocks

- Could be a structure other than that of Bitcoin

Definition 3: a system that maintains a ledger with all transactions

- Not necessarily stored as a chain of blocks
- Aka distributed ledger systems

Definition 4: a system with distributed non-trusting parties collaborating without a trusted intermediary

Definition 5: a system that uses smart contracts

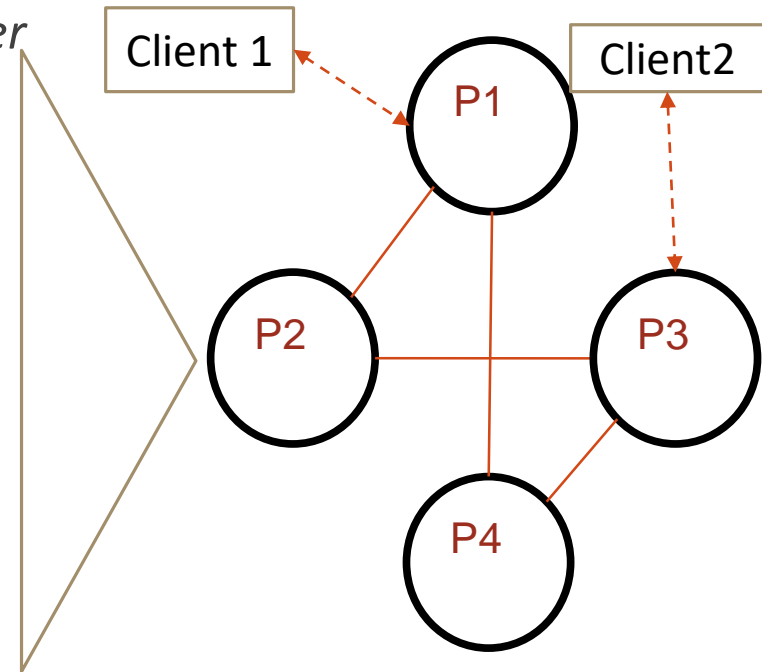
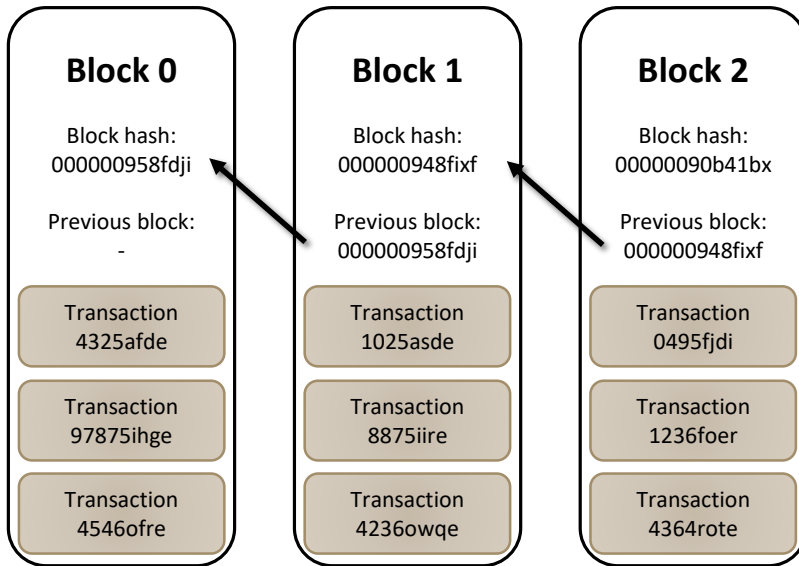
Main benefits of DLTs

Enable parties who do not fully trust each other to form and maintain consensus about the existence, status and evolution of a set of shared facts

The ecosystem of smart contracts

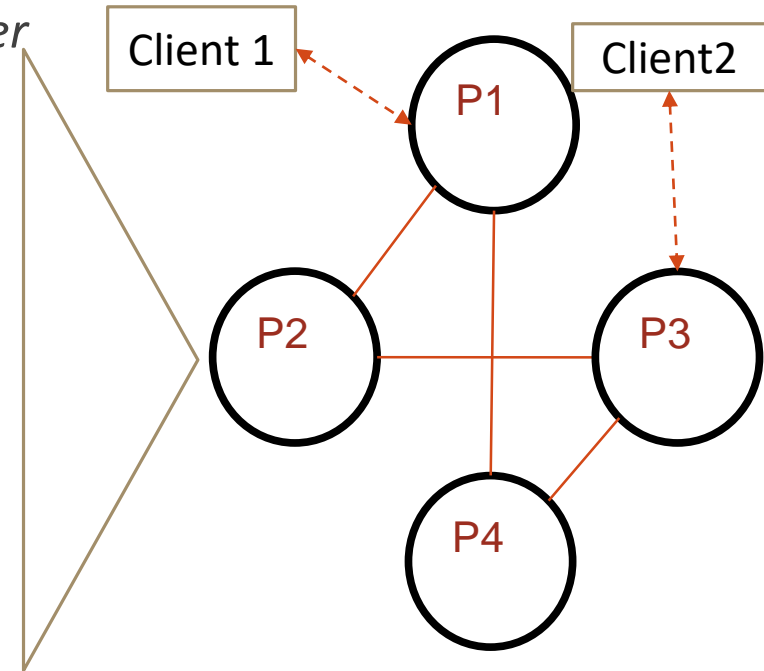
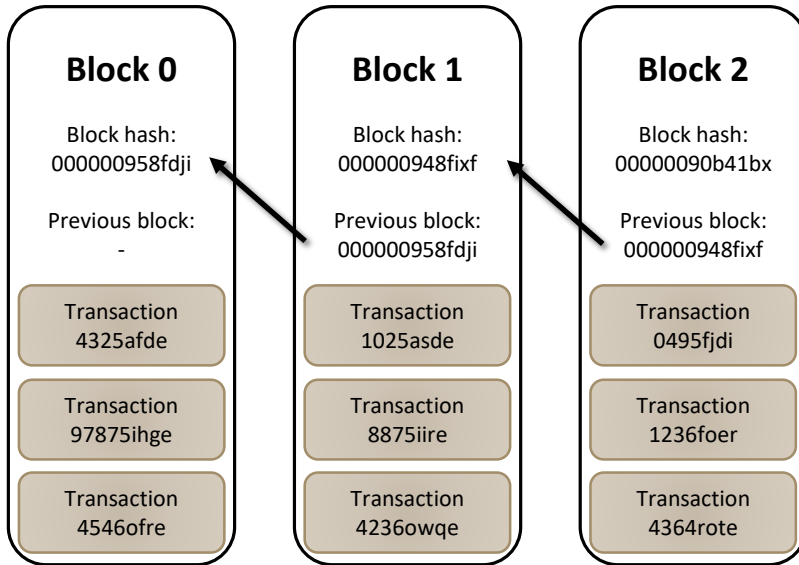
Immutability using Hashing

Blockchain data structure maintained at every peer



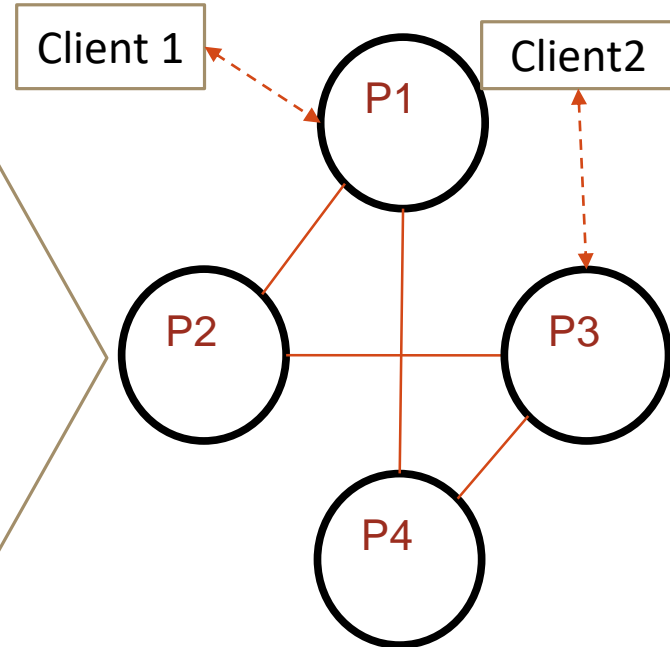
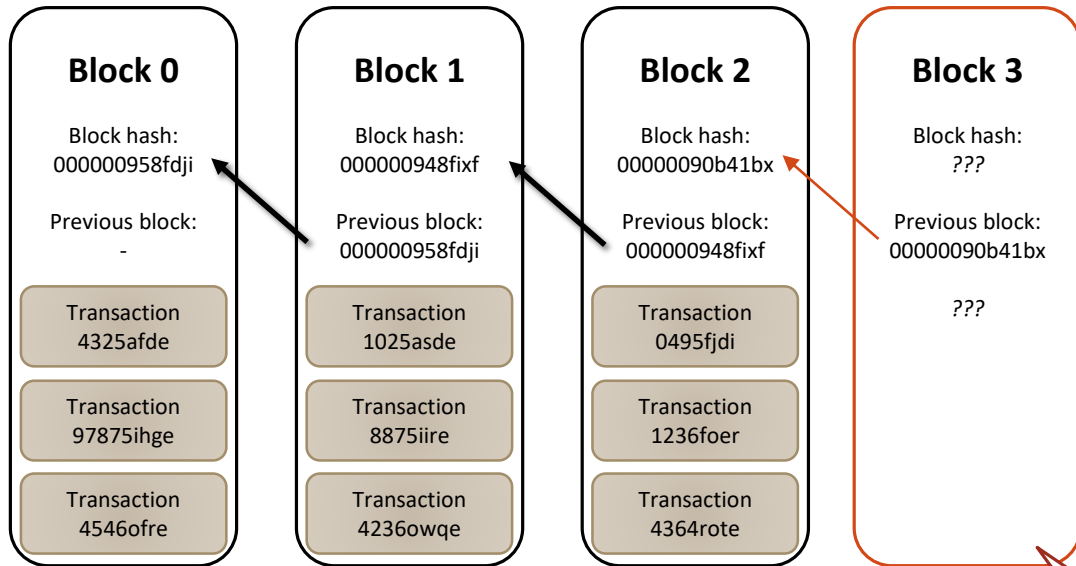
Immutability using Hashing

Blockchain data structure maintained at every peer



Immutability using Hashing

Blockchain data structure maintained at every peer

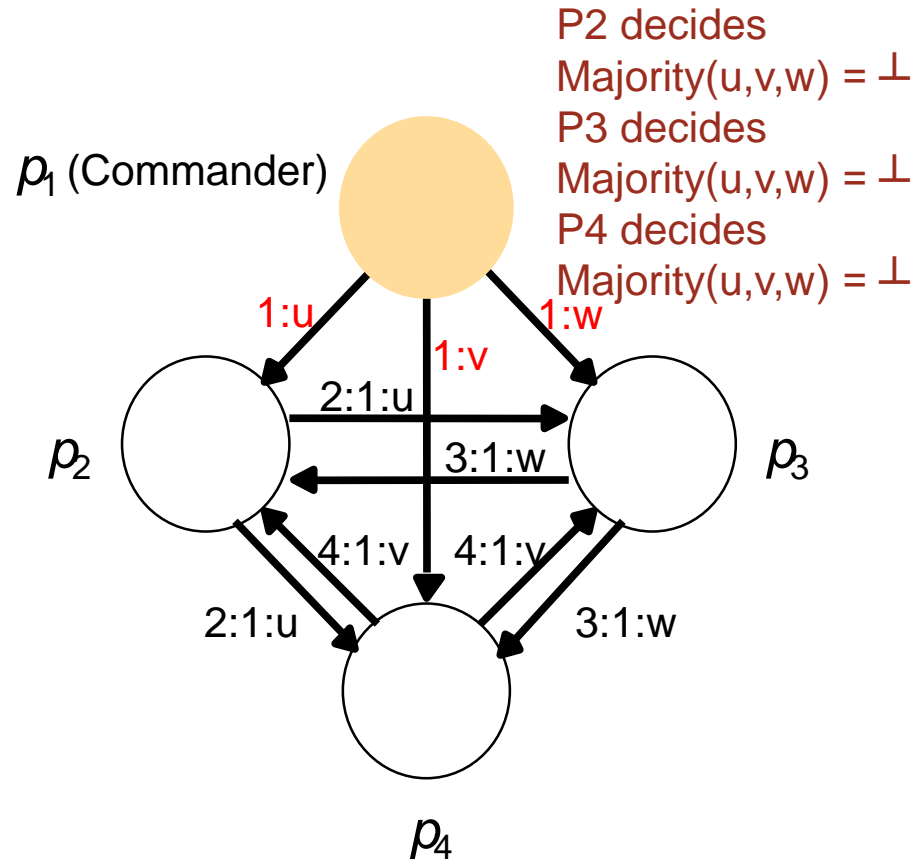
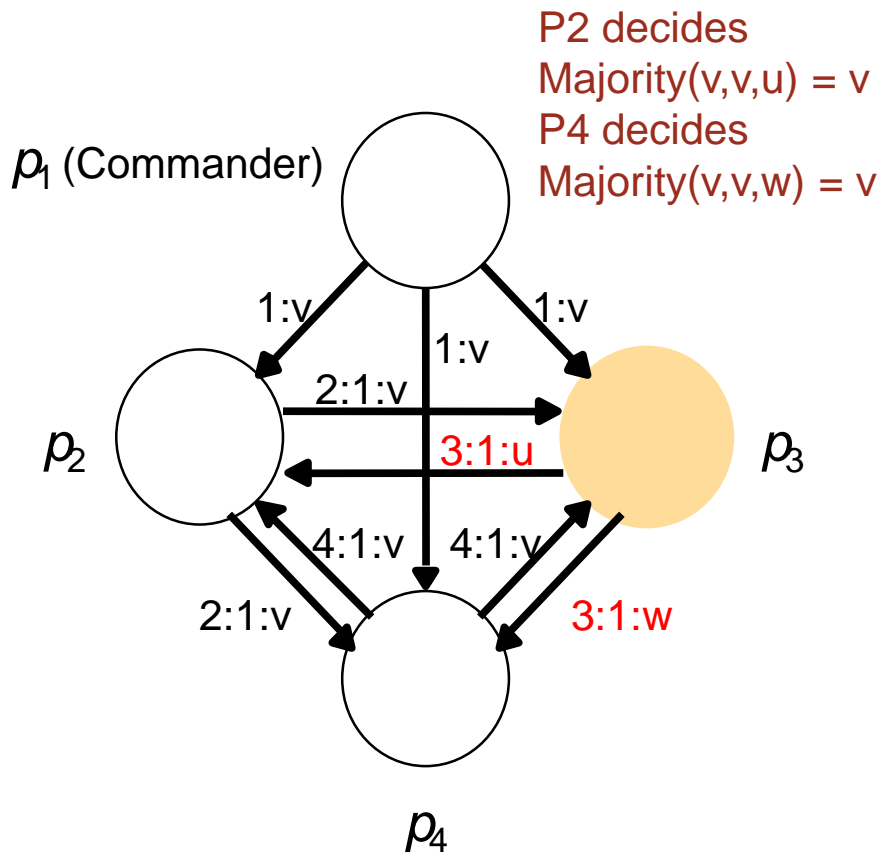


Requires a Byzantine consensus algorithm!

Origin: Byzantine Generals

- Devised by Lamport, 1982
- A *distinguished process* (the **commander**) proposes initial value (e.g., “attack”, “retreat”)
- Other processes, the **lieutenants**, *communicate the commander’s value*
- *Malicious processes* can lie about the value (i.e., *are faulty*)
- *Correct processes* report the truth (i.e., *are correct*)
- Commander or lieutenants may be faulty
- **Consensus** means
 - If the commander is correct, then correct processes should agree on commander’s proposed value
 - If the commander is faulty, then all correct processes agree on a value (*any value, could be the faulty commander’s value!*)

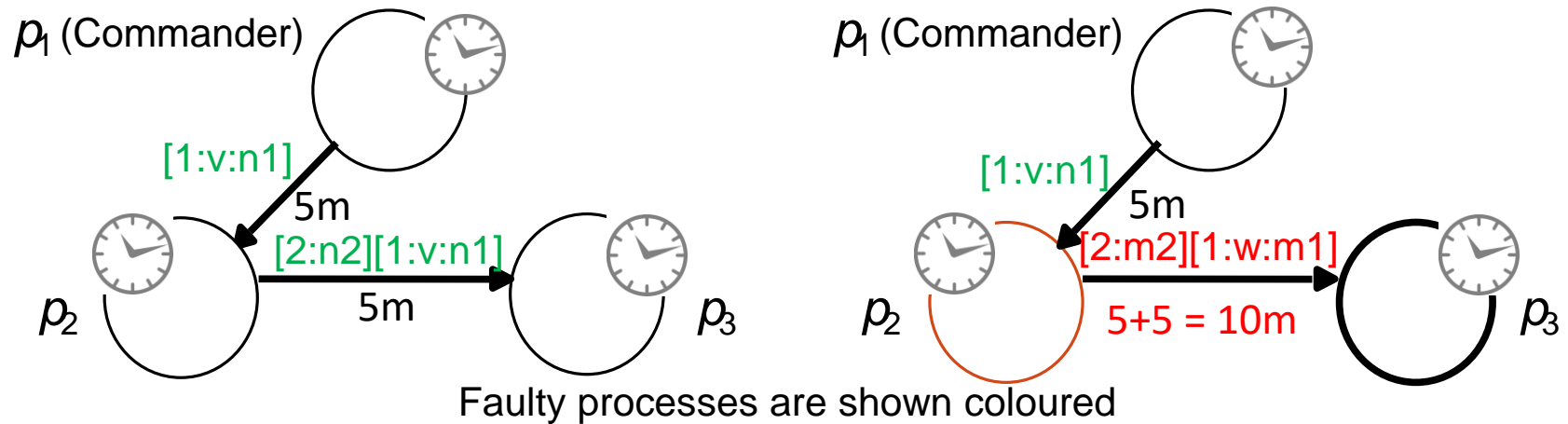
3f+1 Condition (1 failure, 4 nodes)



Source: Tanenbaum, Steen.

Faulty processes are shown coloured

With Blockchains (Proof-of-Work)



Idea #1:

Each message takes exactly 5 minutes to create by any process. ("Magic Block")

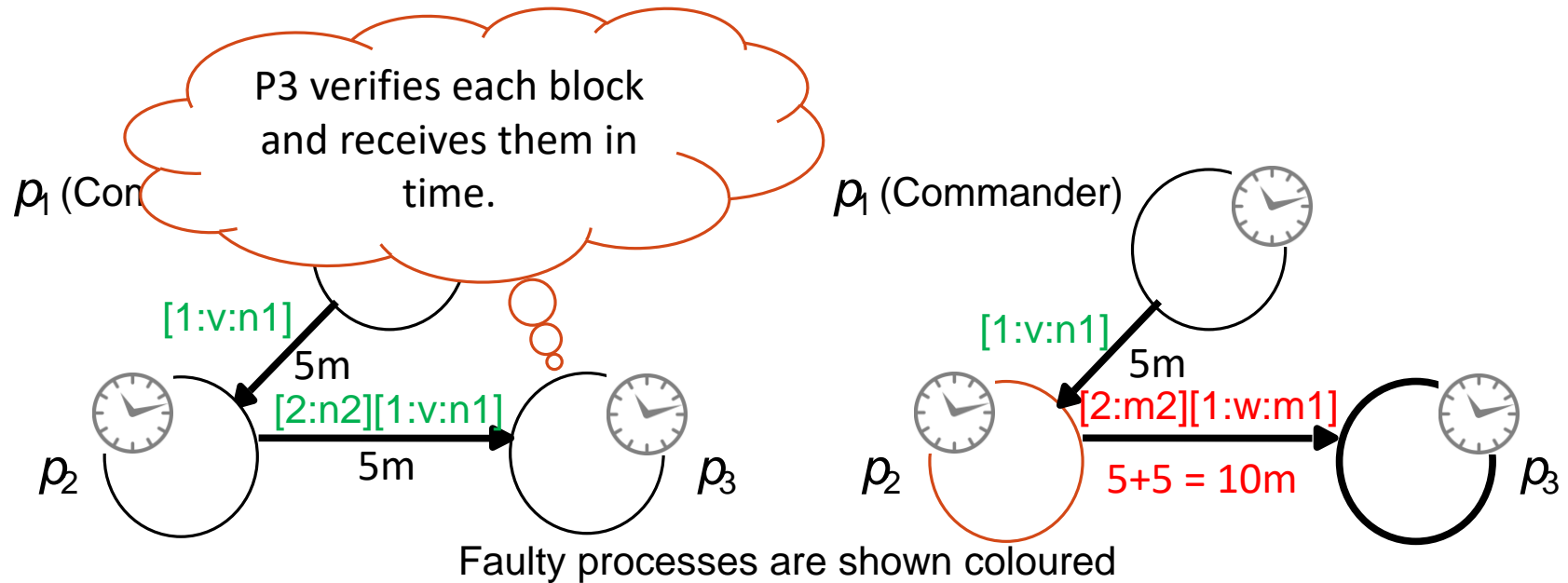


Idea #2:

Each process can accurately measure the amount of time taken by a process to create a message. ("Magic Watch")



With Blockchains (Proof-of-Work)



Idea #1:

Each message takes exactly 5 minutes to create by any process. ("Magic Block")

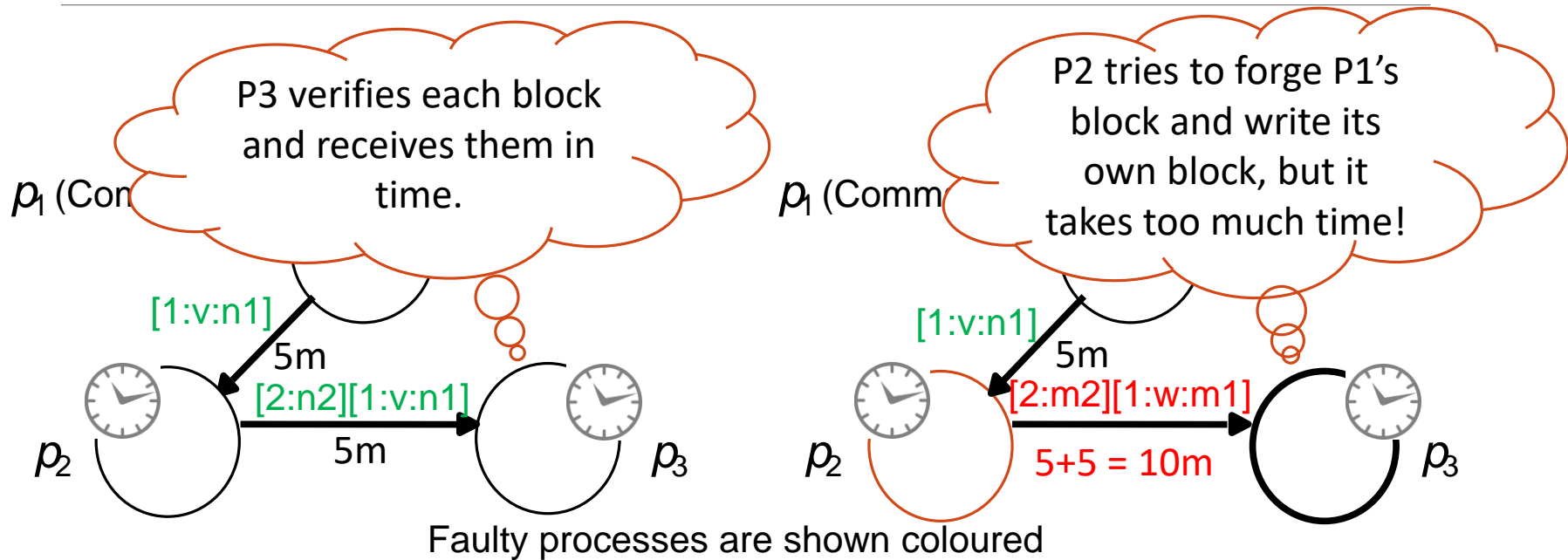


Idea #2:

Each process can accurately measure the amount of time taken by a process to create a message. ("Magic Watch")



With Blockchains (Proof-of-Work)



Idea #1:

Each message takes exactly 5 minutes to create by any process. ("Magic Block")



Idea #2:

Each process can accurately measure the amount of time taken by a process to create a message. ("Magic Watch")




Blockchain “Cryptopuzzles”

verify(nonce, data) meets some “requirements”



Use of “trapdoor functions” (hash functions)

- Cannot reverse the function to find the input
- Therefore, keep trying random values (called nonce) until you find a solution
- Like trying random combinations to a lock...
- *The more computing power you have, **the faster** you can solve the cryptopuzzle.*
- “*Magic blocks*” are blocks with cryptopuzzles, where everyone has the same power. 

Proof-of-Work Example

E.g., the challenge is:

- sha256sum("data:nonce") starts with an "0"
- Normally more complicated than that! (e.g., 18 zeroes)

➤ P1 wants to send "1:v" to P2

```
kzhang@grey:~$ echo "1:v:118" | sha256sum
9479038ca7543ece09f48e8c77fcea147d7561cac14058199afea18c2f323b8b
kzhang@grey:~$ echo "1:v:119" | sha256sum
79ae2bbac929112a349c2fe7f50210355f4a24683b2dd1ea8f059c9beeed7fd6
kzhang@grey:~$ echo "1:v:120" | sha256sum
002ce3a3b7092d960abf1795a89f70eb0f9ef960036e7d4620cbd3d26d34ffc8
```

➤ Send "1:v:120" to p2

Proof-of-Work Example

➤ P2 verifies “1:v:120” is correct (very quick!)

➤ P2 wants to send “2:1:v:120” to P3

```
kzhang@grey:~$ echo "2:1:v:120:119" | sha256sum
911ab1edf1f331ff423a45fe4c382db30a3f1cf802bb2211df53c80d5798c7ba
kzhang@grey:~$ echo "2:1:v:120:120" | sha256sum
5344a3561673b1481b9cf69493368ca408b1edef67e3f96819c5d1b36cea53ce
kzhang@grey:~$ echo "2:1:v:120:121" | sha256sum
0a908c651e9ec5374976dc8f49a3342a4a789660011551da8871a6cc123c5b57
```

➤ P2 sends “2:1:v:120:121”

➤ P3 verifies “1:v:120” AND “2:1:v:120:121” are correct

➤ If P2 wants to send “2:1:w” and fool P3, it needs to find n1 for “1:w:n1” & n2 for “2:1:w:n1:n2”

➤ If P3 has a way to *detect* that P2 is *doing too much work*, it can detect fraud.

Bitcoin

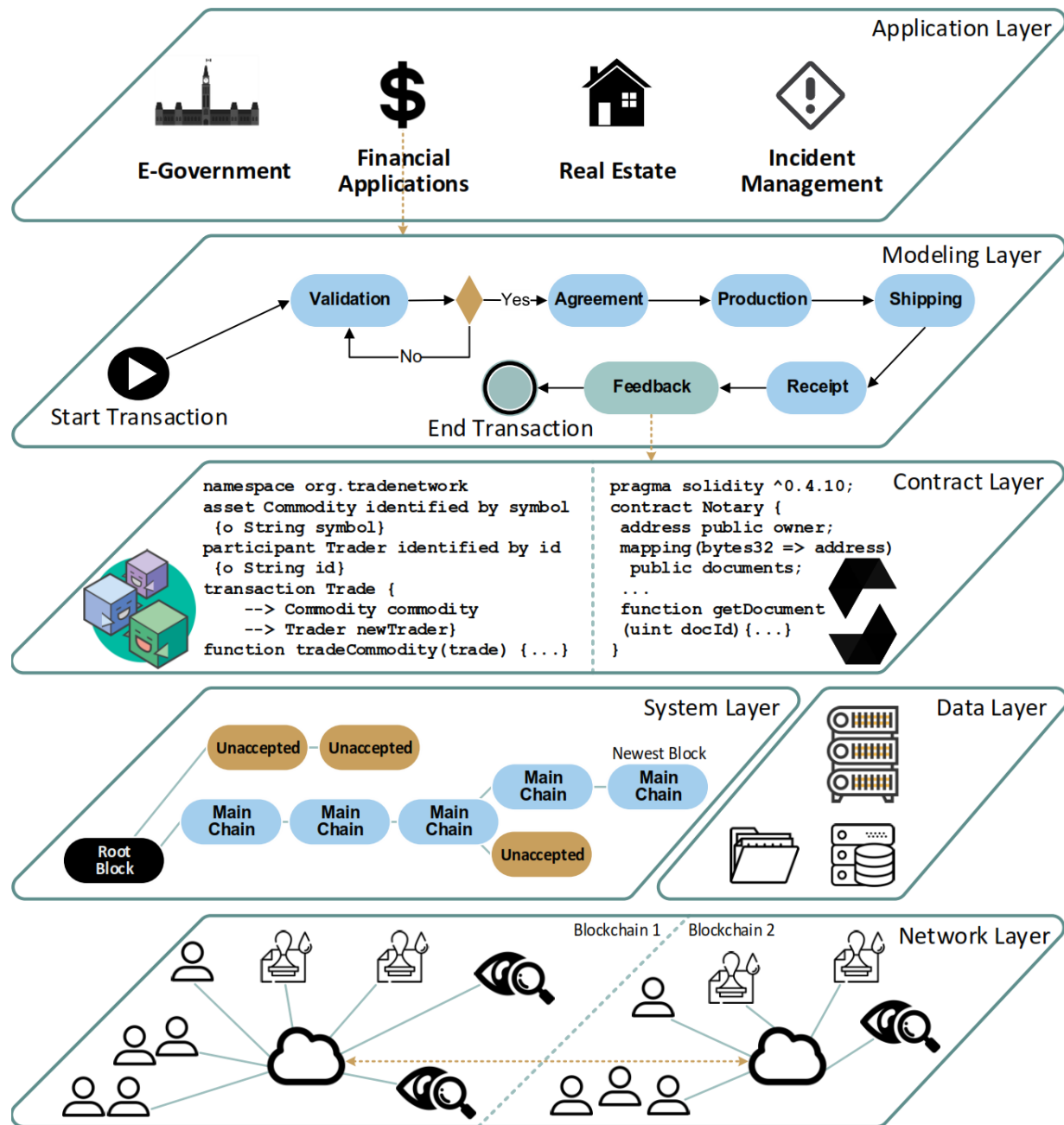
LAYER BY LAYER

Blockchain Reference Architecture

This vision diagram encompasses all aspects related to blockchain technologies.

Upper layers capture application semantics and their implementation.

Lower layers are concerned with technical system details.

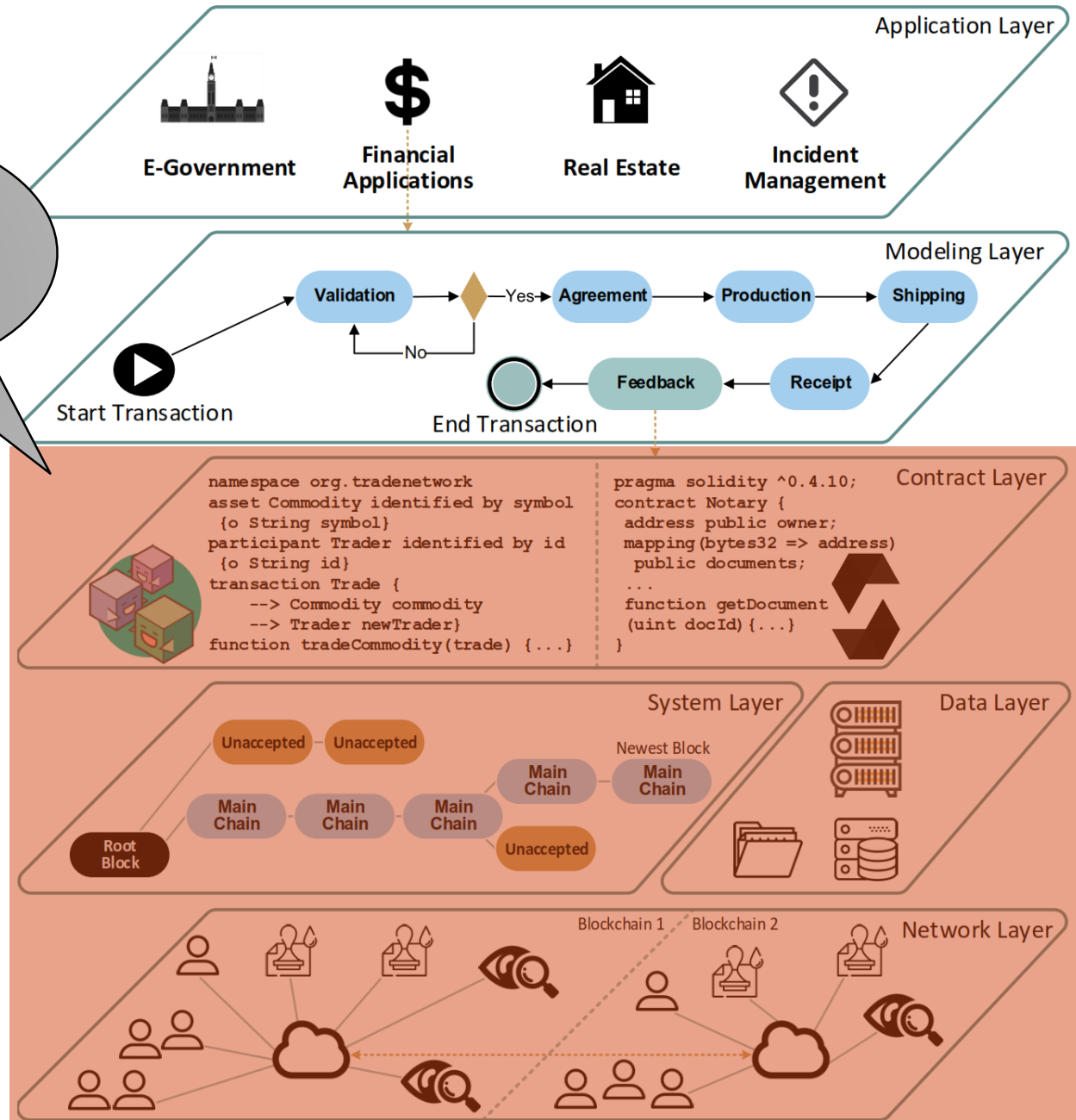
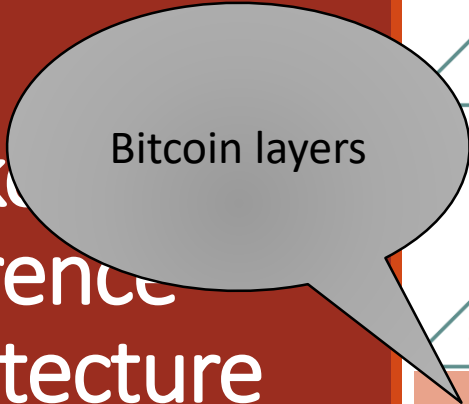


Blockchain Reference Architecture

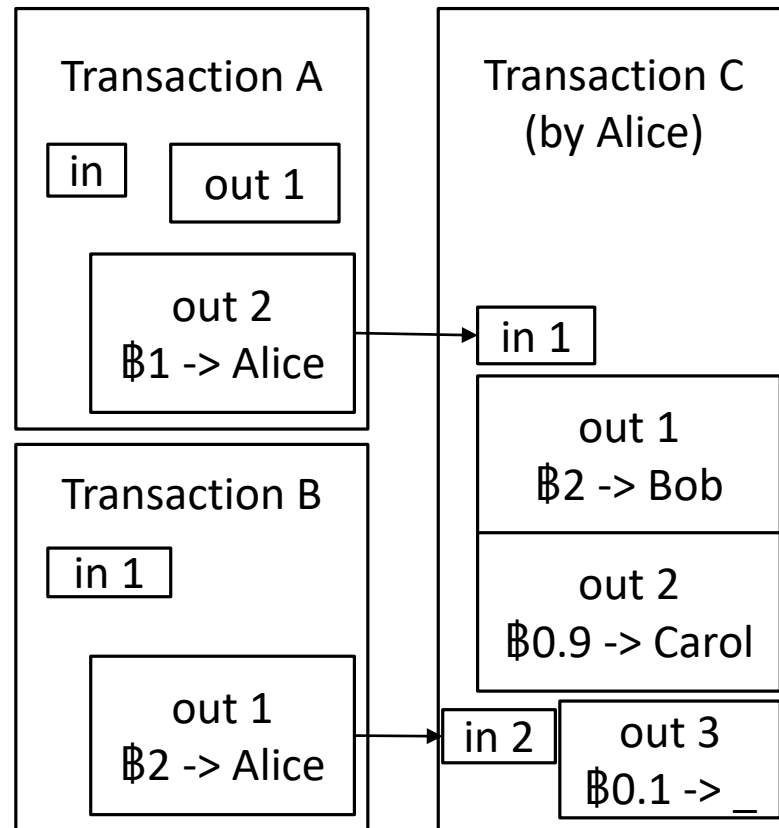
This vision diagram encompasses all aspects related to blockchain technologies.

Upper layers capture application semantics and their implementation.

Lower layers are concerned with technical system details.

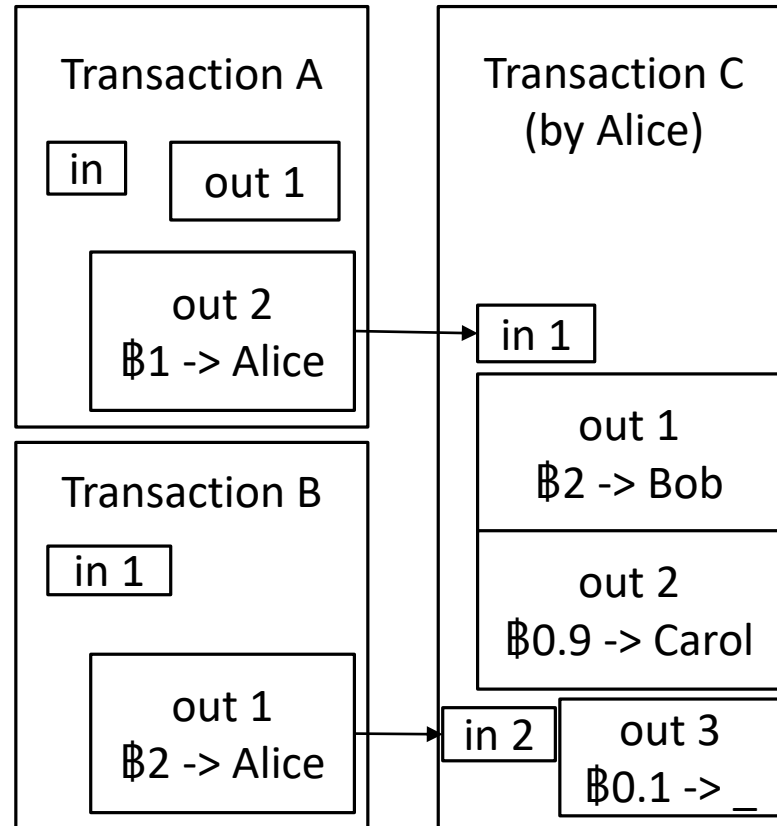


Bitcoin Transactions



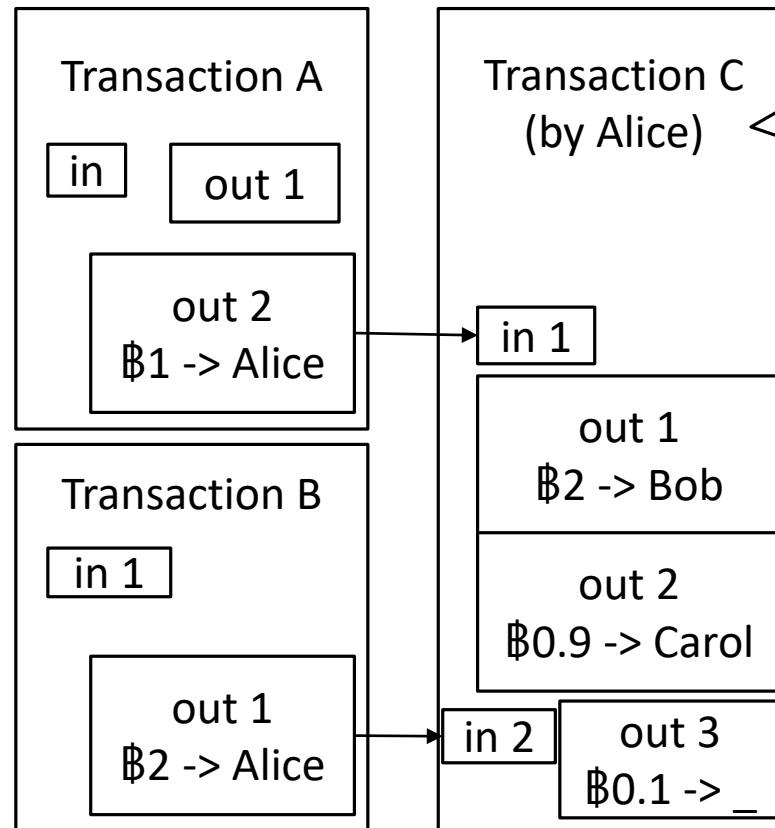
Bitcoin Transactions

Each user possesses a wallet identified by *public/private key pairs*



Bitcoin Transactions

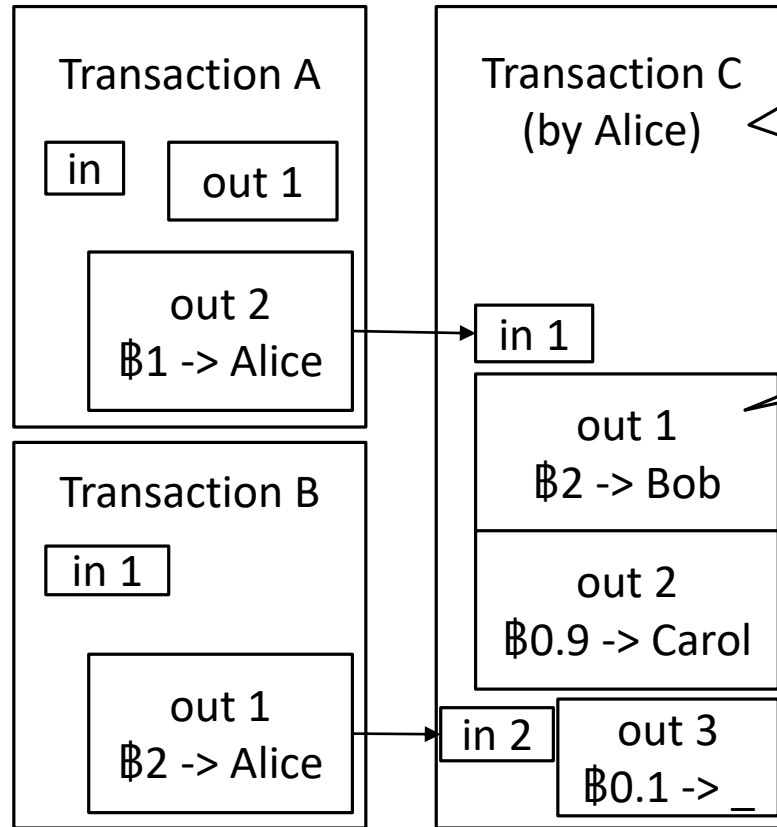
Each user possesses a wallet identified by *public/private key pairs*



User encrypts a new transaction C using the private key

Bitcoin Transactions

Each user possesses a wallet identified by *public/private key pairs*

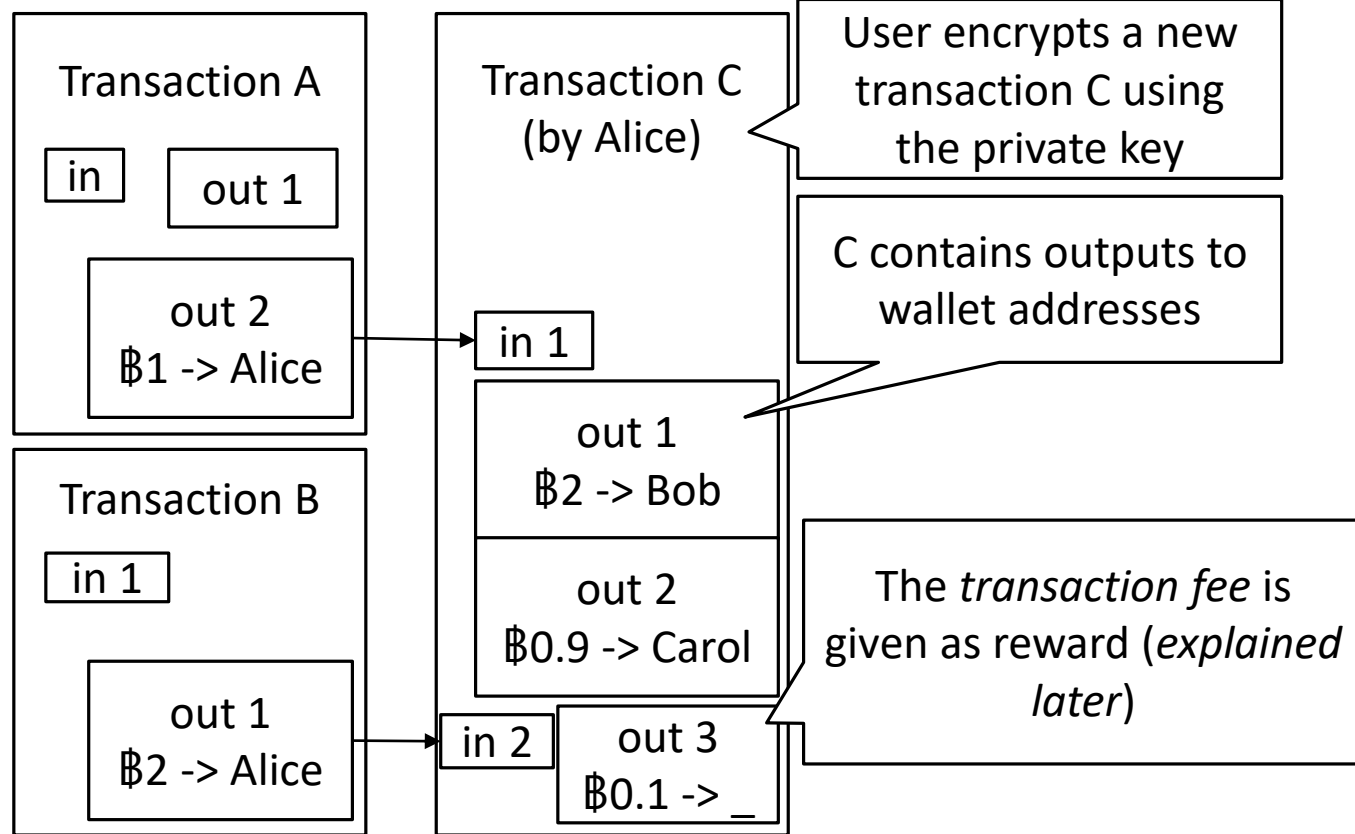


User encrypts a new transaction C using the private key

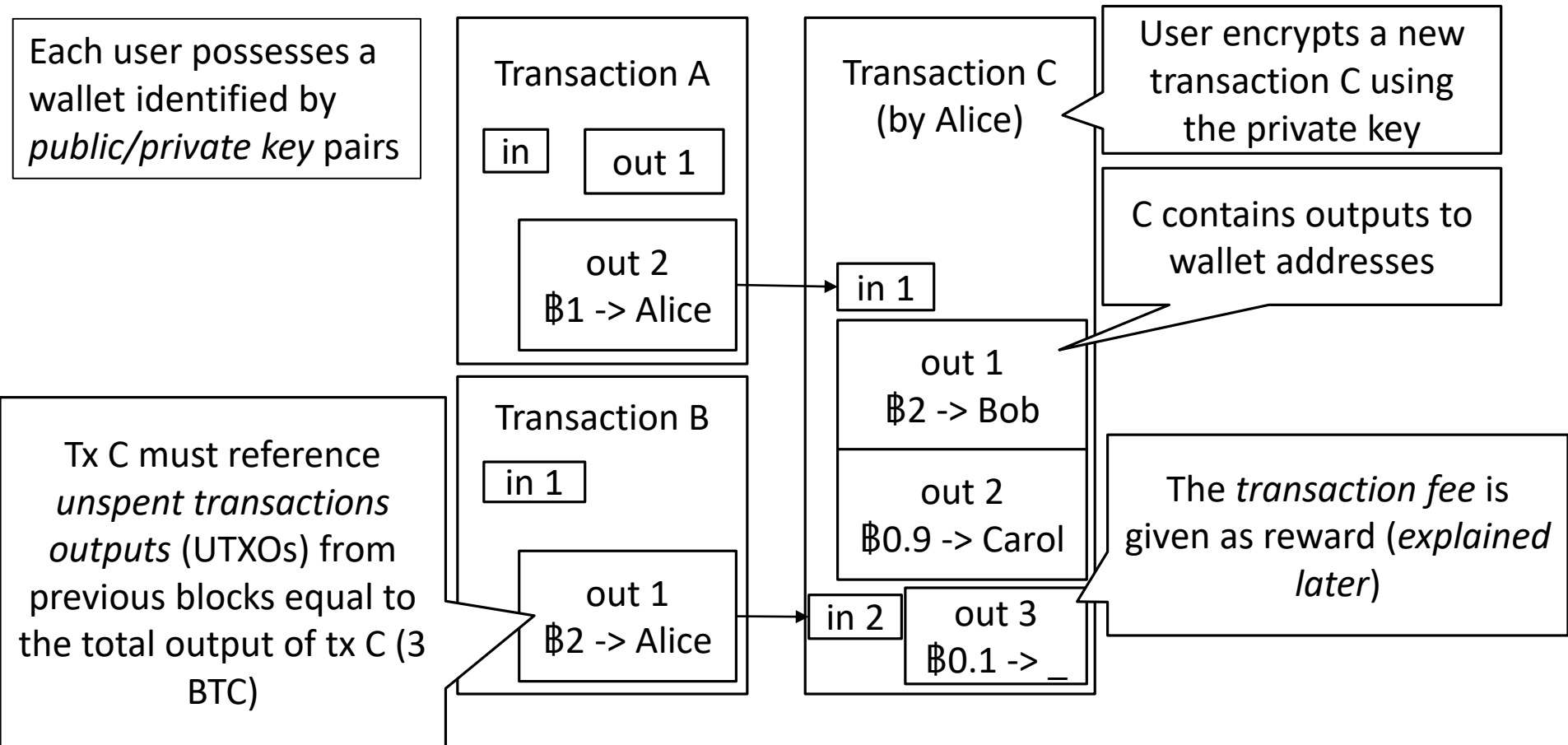
C contains outputs to wallet addresses

Bitcoin Transactions

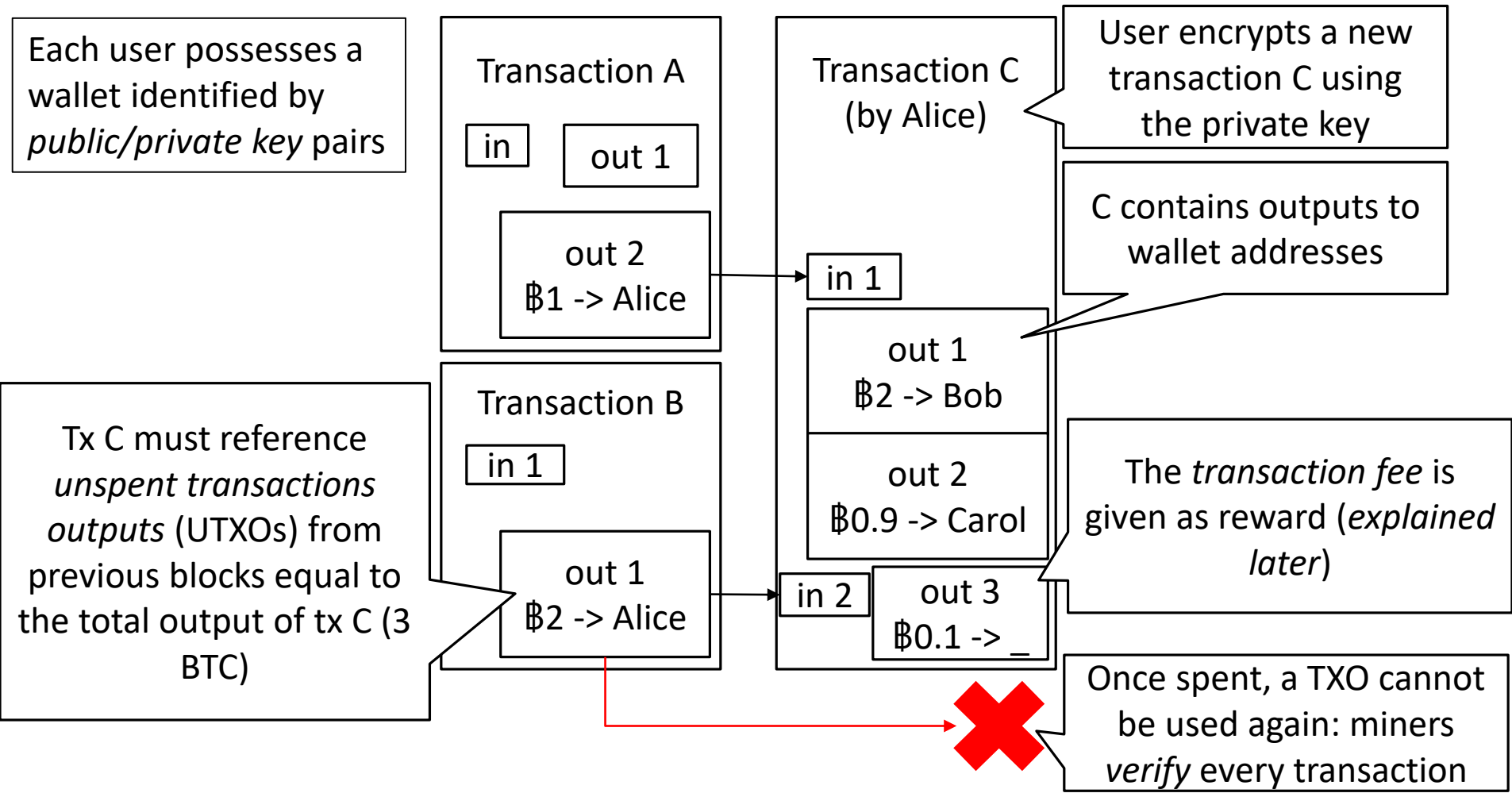
Each user possesses a wallet identified by *public/private key pairs*



Bitcoin Transactions



Bitcoin Transactions



Wallets and addresses



Users require a wallet to store money

- This includes any user, **including but not limited to** miners

Wallet is authenticated and identified by a public/private key pair

- Generated using ECDSA (Elliptic curve cryptography)
- More details here:
https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses

Redeeming transactions:

- Each TXO address is a hash of the public key of the receiver, who signs proof with the private key
- Transactions do not have a “from” address, so it is impossible to prove you are the sender
- Each address is designed to be single use: wallet programs will automatically generate new addresses

Losing your private key:

- Loss of private key means the wallet and its funds are permanently locked, as it is no longer possible to sign proofs redeeming existing TXOs.
- This money is essentially lost, thereby reducing the total amount of currency in Bitcoin
- Trusting an online service to store your key is also risky, since there is no way to prove that you are the rightful owner if the key is stolen or misused
- The most reliable solution is to store your private keys on tamper-proof hardware wallets

Communication in Bitcoin

Broadcast to all the network

Two primary uses

- Users broadcast their transactions
- Miners broadcasts updates to the blockchain (new blocks)

Implemented via gossiping protocol in a P2P network

- Not terribly efficient but has not been a bottleneck so far

Works because financial transactions are very short and their rate in Bitcoin is far below that of credit cards

Needs to be fairly reliable for the system to work but 100 percent reliability in message delivery is not required

- Users and miners need to detect message loss and retransmit messages if needed

Message propagation should be reasonably fast

- Slower network quantifiably increases the risk of attacks

Transaction Flow

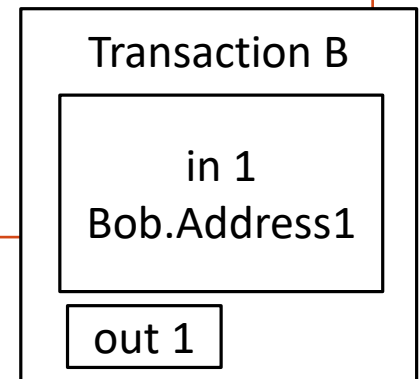
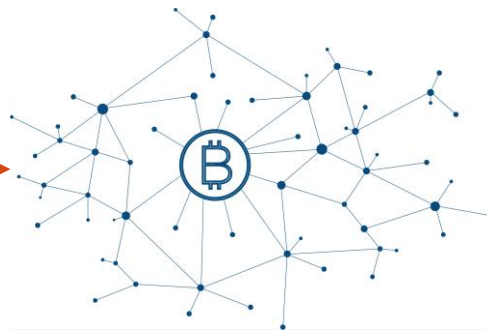
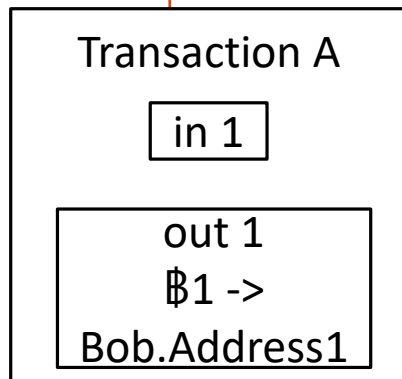


Alice
(Sender)



Bob
(Receiver)

1. Bob generates and send a public key address.
2. Alice creates a transaction using this address.
3. Alice sends the new transaction to the network.
4. The transaction is broadcast using gossiping.
5. The transaction is included in a block.
6. Bob can verify the transaction is in the blockchain.
7. Bob can now sign new transactions which redeem this address.

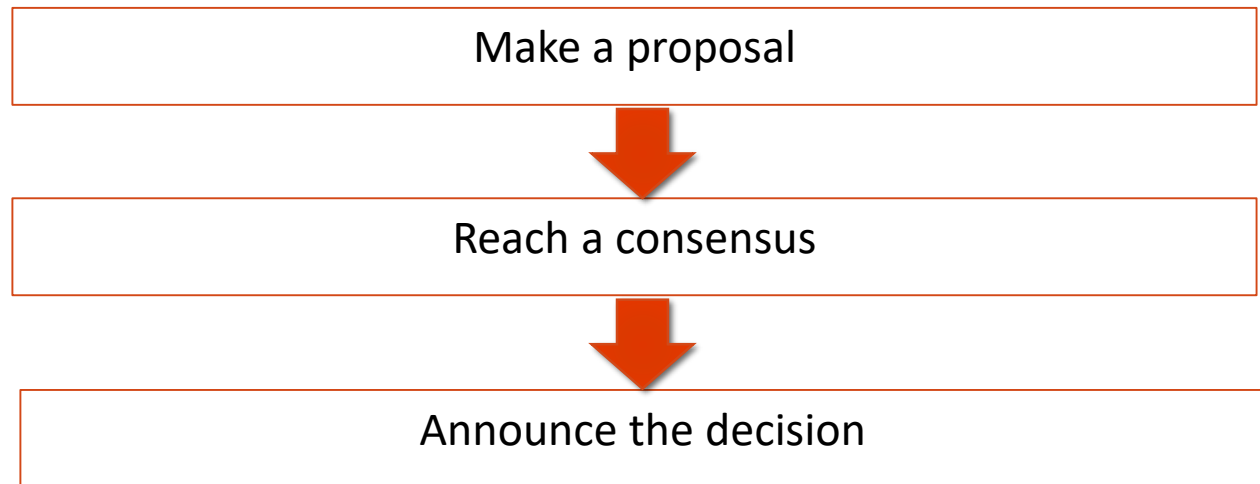


Consensus in Bitcoin

The network needs to agree on

- Which recently broadcast transactions go into the blockchain
- In what order

The general anatomy of consensus:

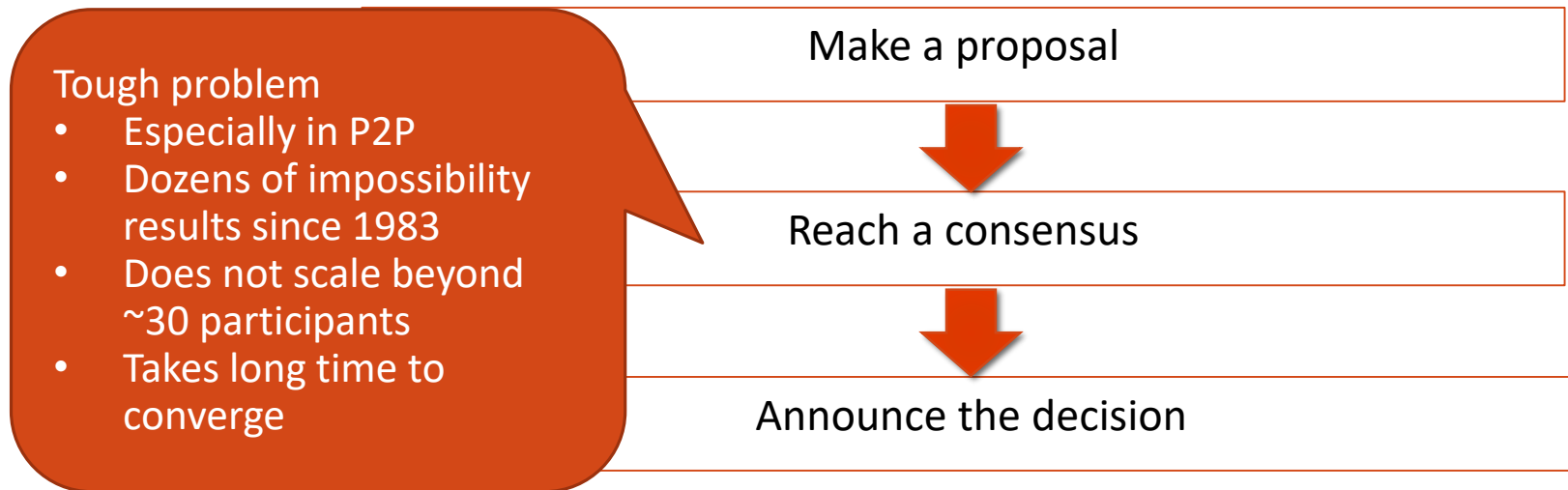


Consensus in Bitcoin

The network needs to agree on

- Which recently broadcast transactions go into the blockchain
- In what order

The general anatomy of consensus:



Challenge 1: who proposes and when?

The network cannot sustain each and every user or peer making a proposal whenever she wishes

Made worse by the proliferation of identities (Sybil attack)

Need to moderate the number of proposers and rate of concurrent proposals

- While keeping them sufficiently high

Several principal solutions

- Proof-of-work: need to do heavy computation and show the proof of it
- Proof-of-stake: need to possess a sufficient amount of coins

Important optimization: propose new transactions in batches

- A block in Bitcoin is structured as a tree of proposed transactions
- With nice cryptographic properties; called a Merkle tree

Cryptopuzzles in Bitcoin

The proposer has to find **nonce**, such that $hash(\mathit{nonce} \mid H \mid Tr_1 \mid \dots \mid Tr_n) < \mathit{target}$

Effectively has to scan the entire **nonce** space

target is a fraction of the hash space

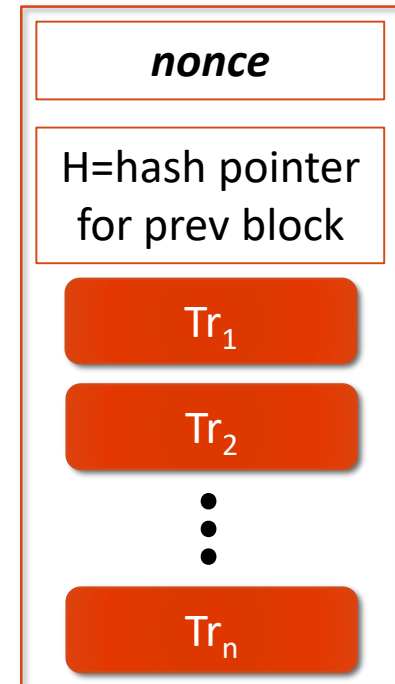
- Every node recomputes **target** every 2016 blocks
- Such that the average time for the whole network to solve a cryptopuzzle is 10 min

For proposer p ,

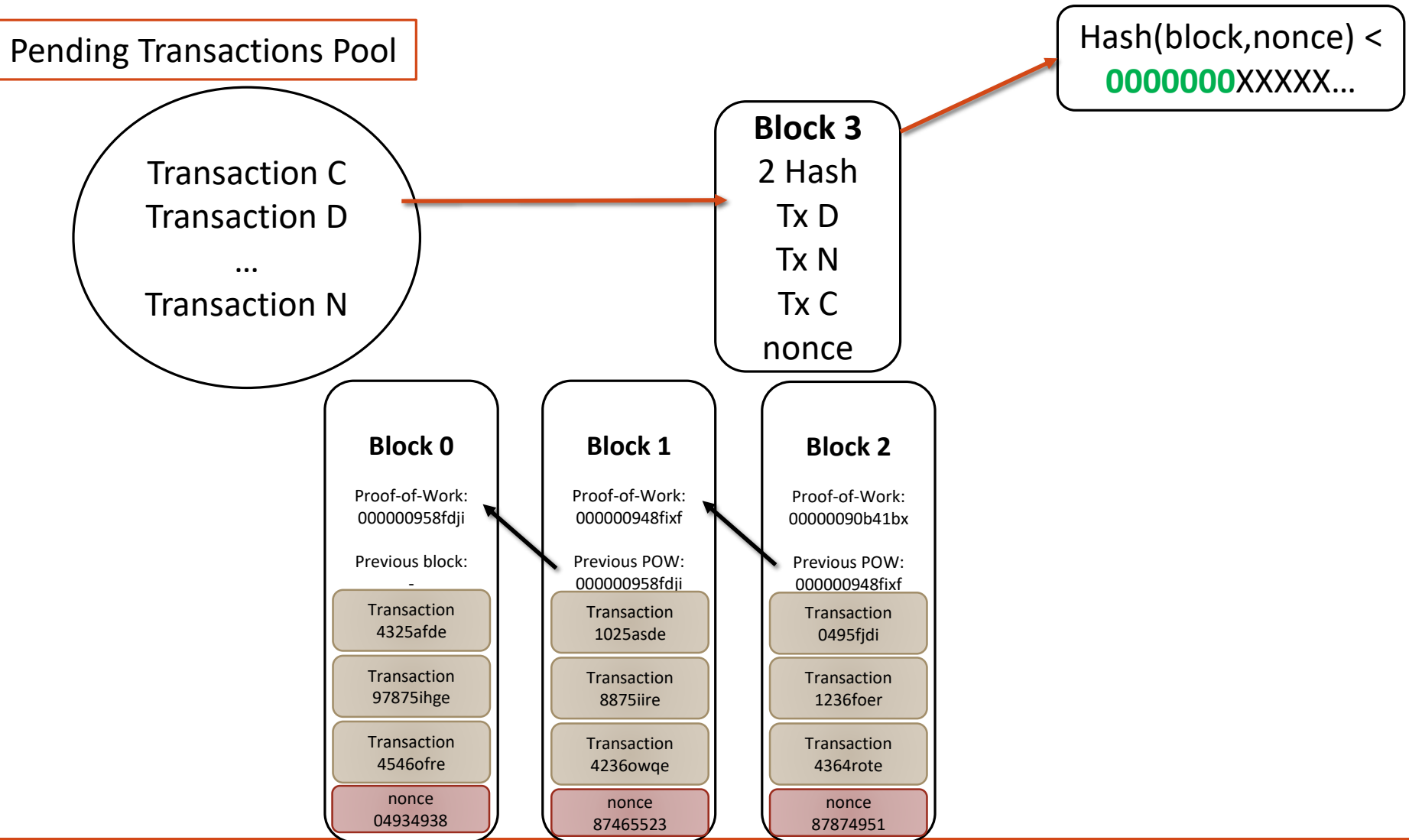
$$\mathit{mean\ time\ to\ next\ block} = \frac{10\ \mathit{minutes}}{\mathit{fraction\ of\ } p\ \mathit{'s\ computing\ power}}$$

The solution is fast to verify

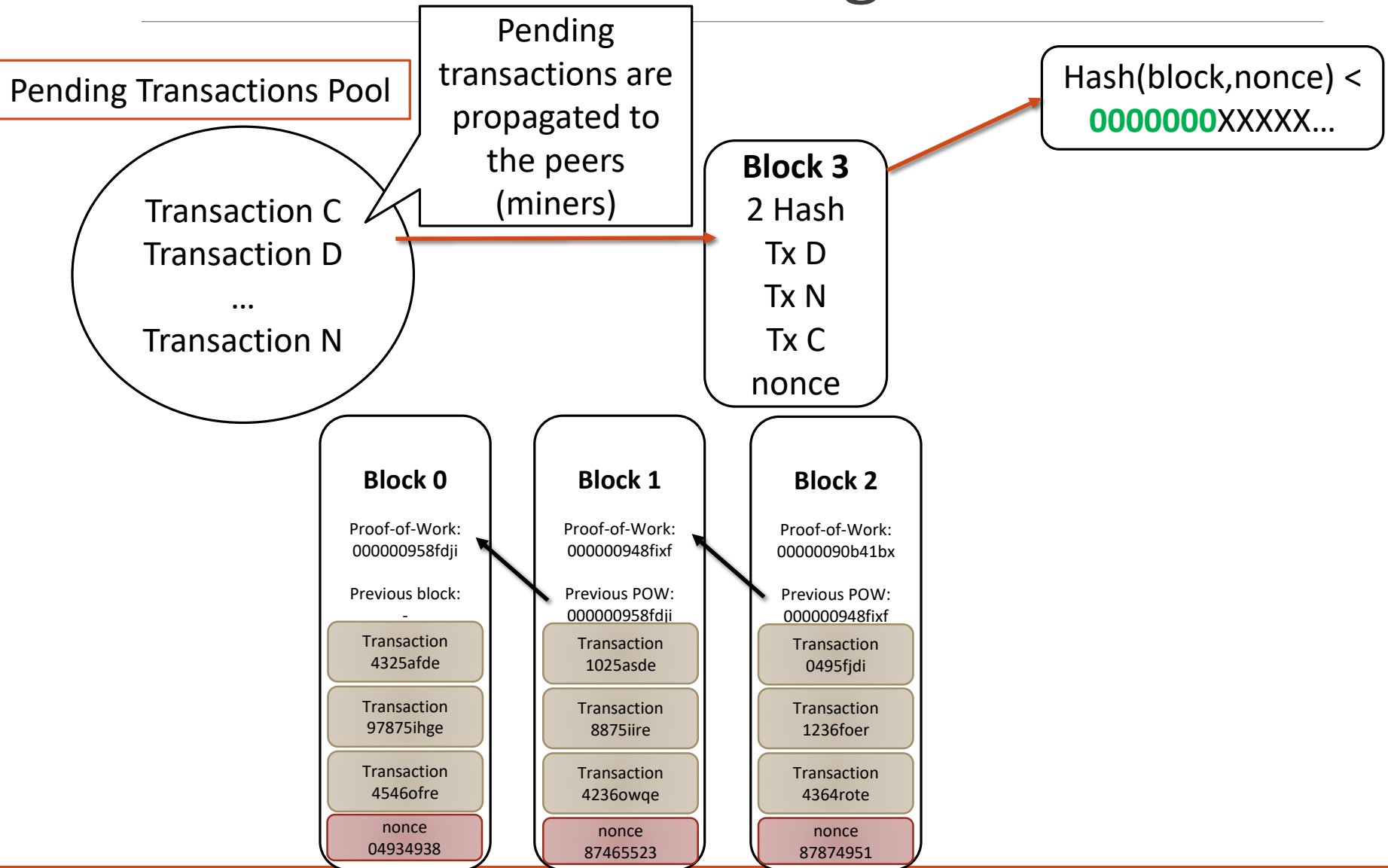
A block in
Bitcoin



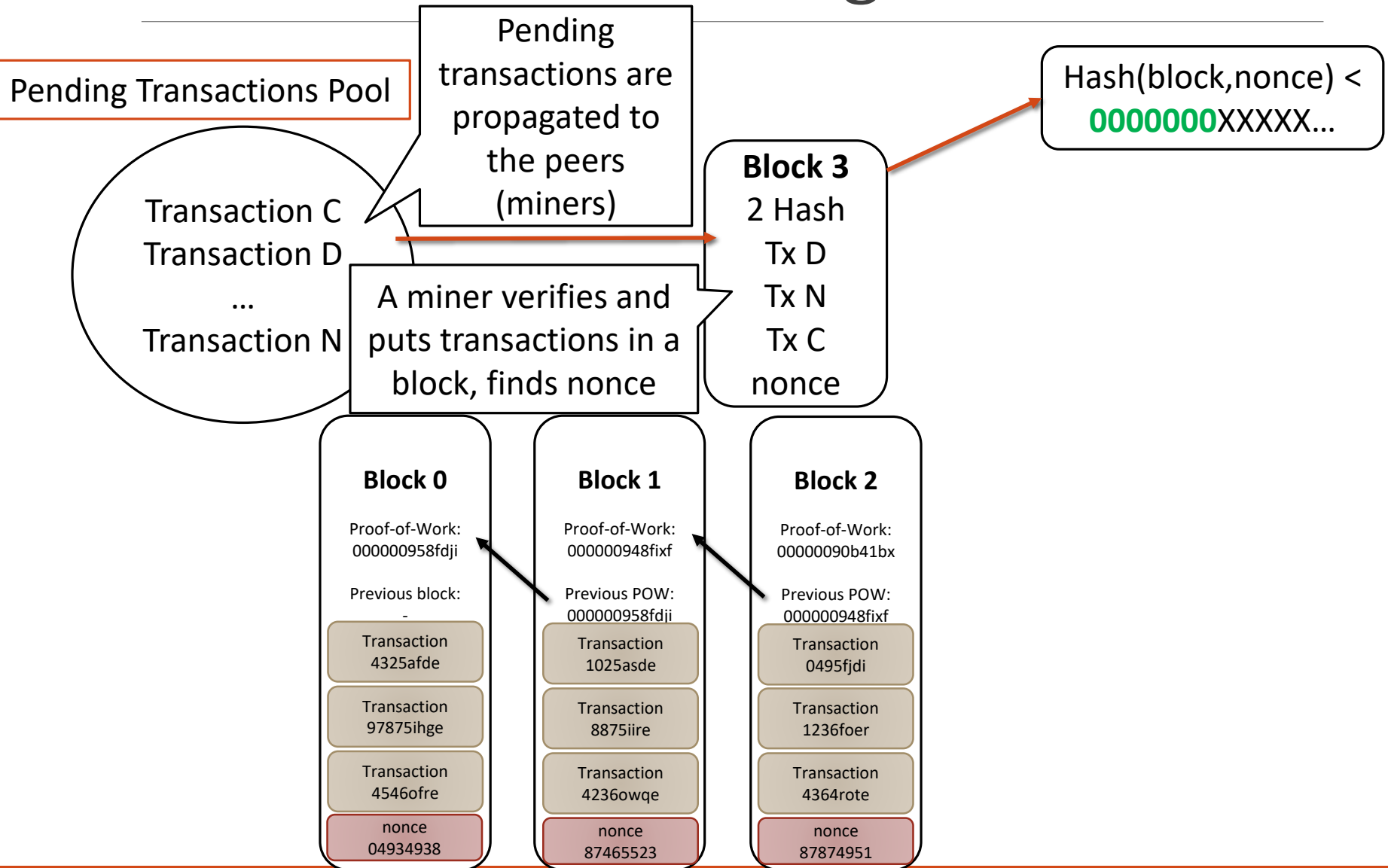
Proof-of-Work Mining in Bitcoin



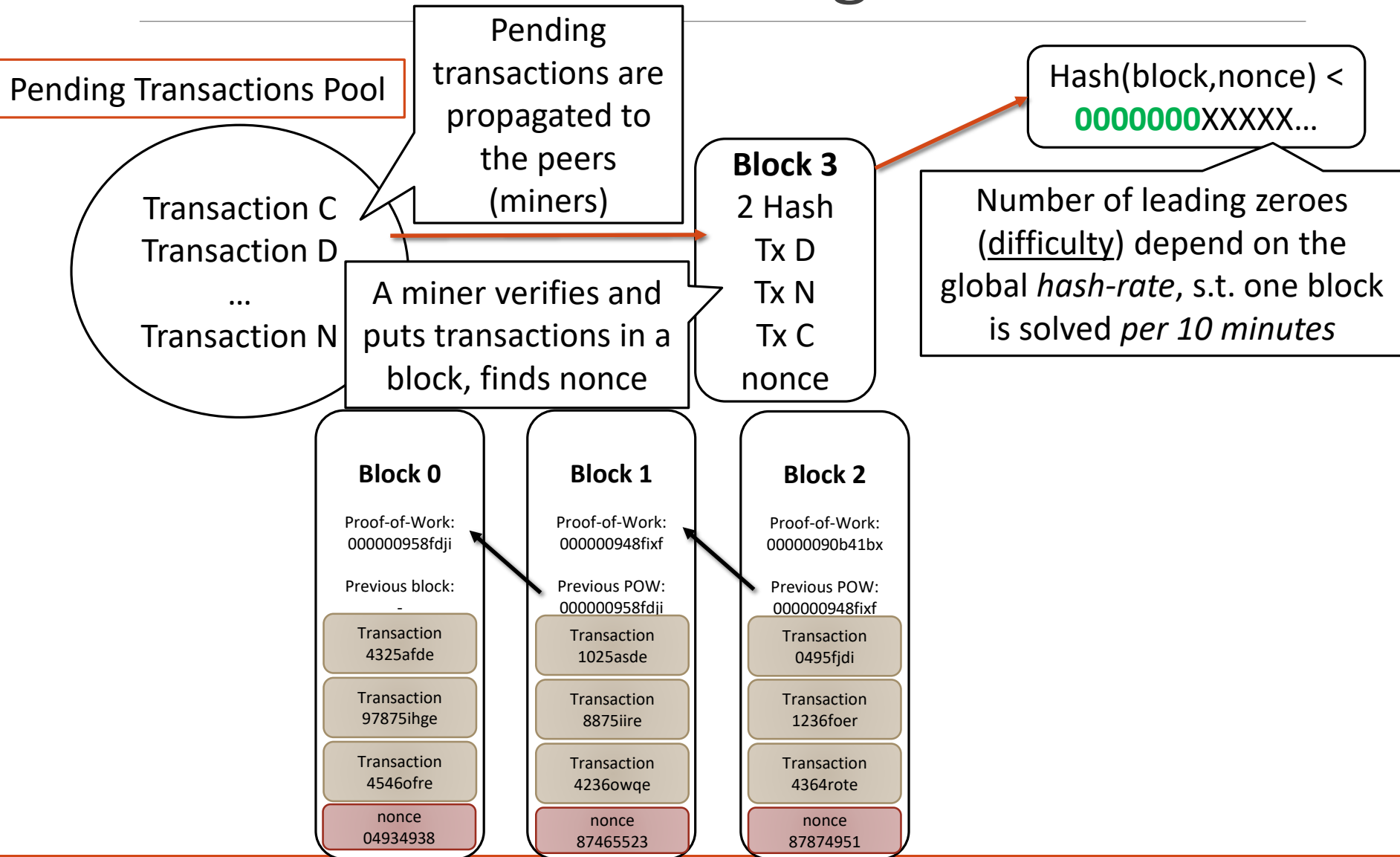
Proof-of-Work Mining in Bitcoin



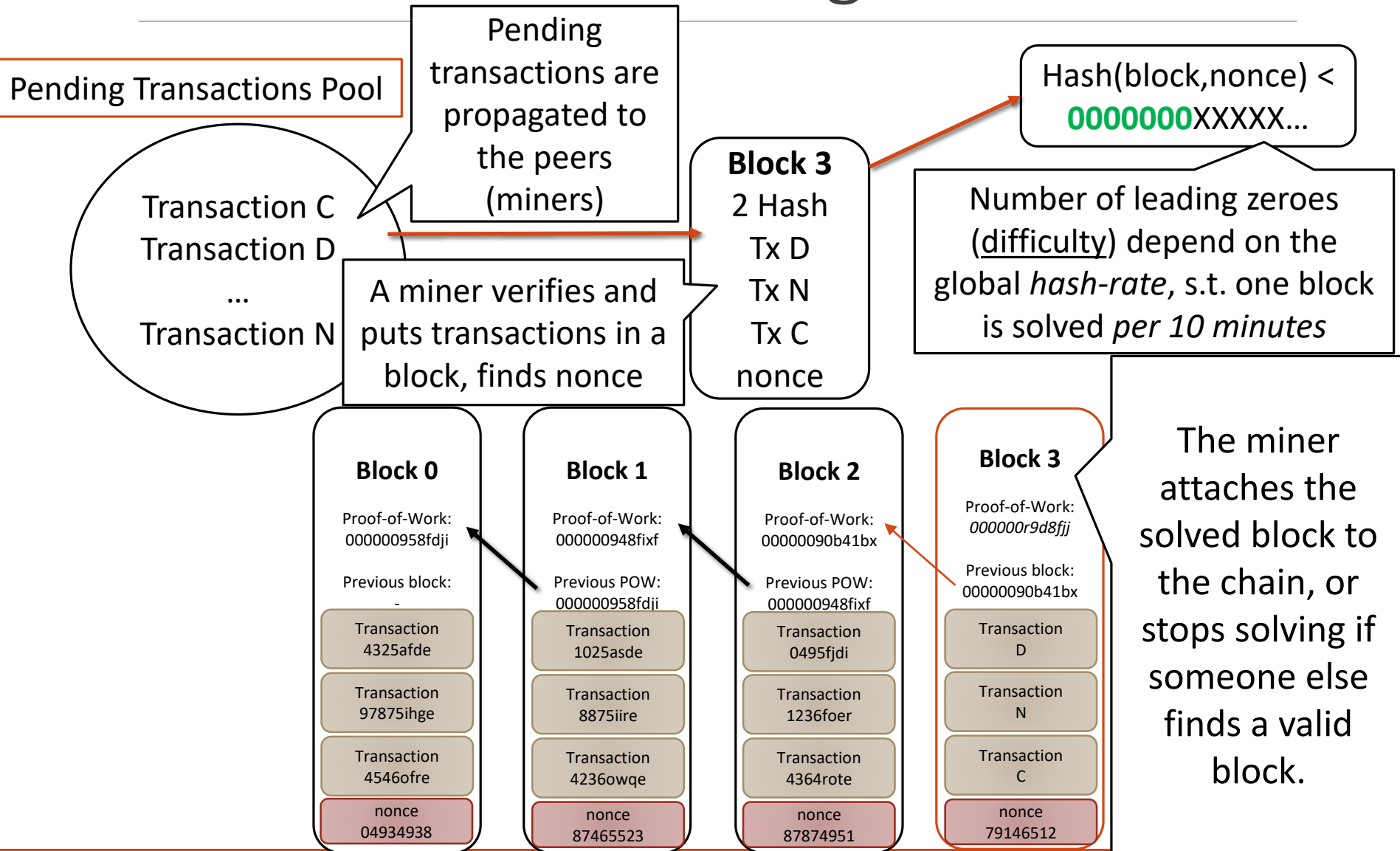
Proof-of-Work Mining in Bitcoin



Proof-of-Work Mining in Bitcoin



Proof-of-Work Mining in Bitcoin



Challenge 2: Why propose non-empty blocks?

Two incentive mechanisms in Bitcoin

- Block creation reward: a block proposal creates a number of new bitcoins and transfers them to the proposer
 - Included as a separate transaction in the block
 - Ensures that each proposer solves a different cryptopuzzle
 - The only way to create new bitcoins
 - The amount is predefined and gets halved every 210,000 blocks
 - Predicted to go down to zero before year 2140
 - The geometric progression totals to 21 million bitcoins
 - The rules may change in the future
- Transaction inclusion fee: Alice can decide to pay a small fee to the block creator as part of her transaction
 - Voluntarily, there is no predefined amount

Cryptoeconomy of Mining

Incentives give rise to the mining industry in Bitcoin

- Miners: cracking cryptopuzzles and listening to transaction broadcasts

Expenses: *mining rig + operating costs (electricity, cooling, repairs)*

- Paid in real currency
- Operating costs are variable

Profits: *block reward + transaction fee * # of transactions in a block*

- Paid in Bitcoins
- The fee and rate of transactions are unpredictable
- The mean time to next block is easy to compute
 - However, the per-miner sample is small while variations are huge

Mining pools: groups of cooperating miners

Reaching consensus in Bitcoin

A miner broadcasts the proposed block

- The block includes a hash to the latest block known to the miner

Reaching consensus in Bitcoin

A miner broadcasts the proposed block

- The block includes a hash to the latest block known to the miner

When a peer receives a proposed block

- Check that the proof of cryptopuzzle solution is valid
- Check that each transaction is valid (business logic)
- If the hash pointer is valid, append the new block to the local copy of the blockchain

Reaching consensus in Bitcoin

A miner broadcasts the proposed block

- The block includes a hash to the latest block known to the miner

When a peer receives a proposed block

- Check that the proof of cryptopuzzle solution is valid
- Check that each transaction is valid (business logic)
- If the hash pointer is valid, append the new block to the local copy of the blockchain

Local copies may diverge!

- Lost messages and concurrent blocks arriving in reverse order
- The probability depends on the network

Reaching consensus in Bitcoin

A miner broadcasts the proposed block

- The block includes a hash to the latest block known to the miner

When a peer receives a proposed block

- Check that the proof of cryptopuzzle solution is valid
- Check that each transaction is valid (business logic)
- If the hash pointer is valid, append the new block to the local copy of the blockchain
- Conflict resolution: if the proposed chain is longer than the current local copy, replace the local copy

Local copies may diverge!

- Lost messages and concurrent blocks arriving in reverse order
- The probability depends on the network

Reaching consensus in Bitcoin

A miner broadcasts the proposed block

- The block includes a hash to the latest block known to the miner

When a peer receives a proposed block

- Check that the proof of cryptopuzzle solution is valid
- Check that each transaction is valid (business logic)
- If the hash pointer is valid, append the new block to the local copy of the blockchain
- Conflict resolution: if the proposed chain is longer than the current local copy, replace the local copy

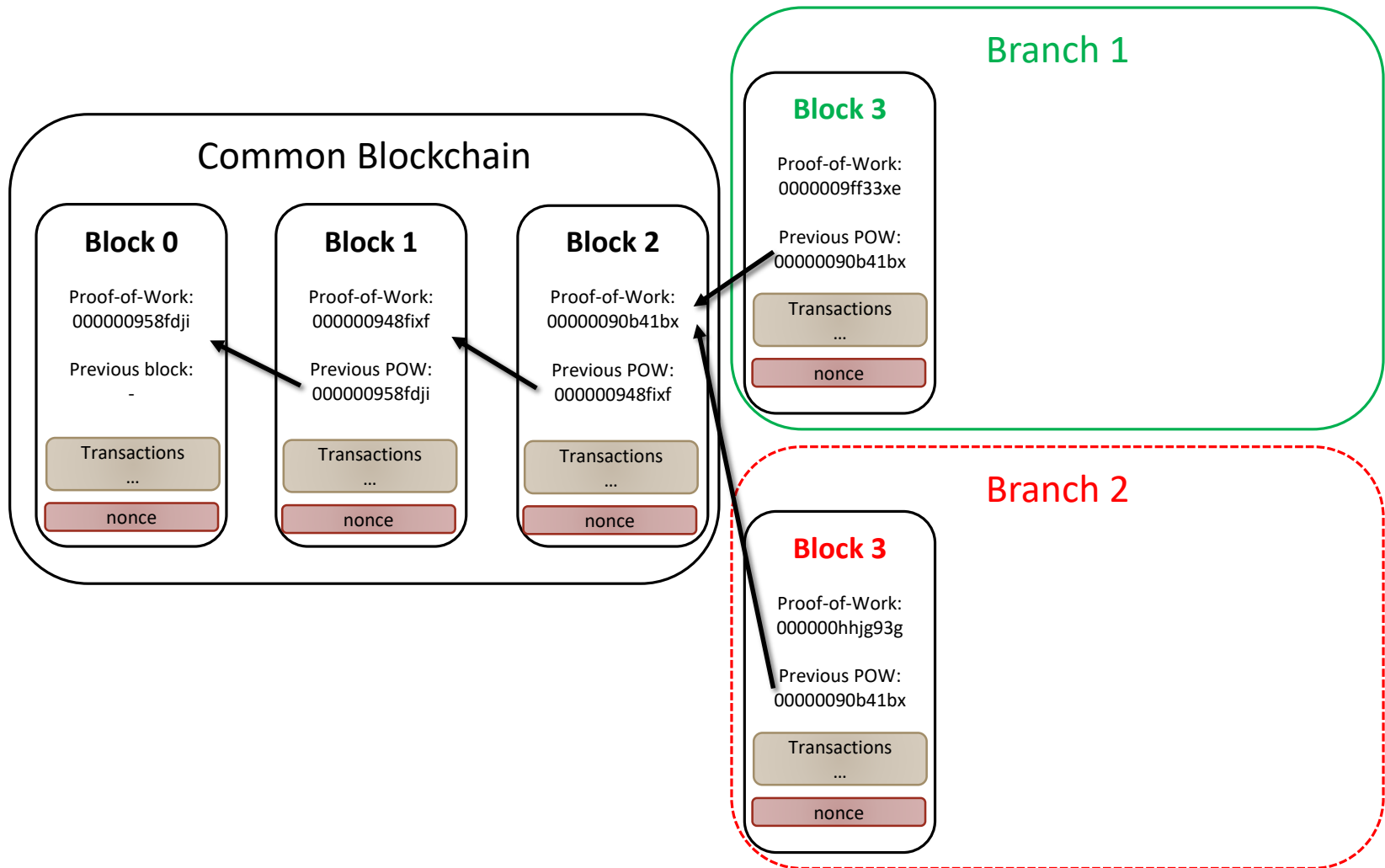
Local copies may diverge!

- Lost messages and concurrent blocks arriving in reverse order
- The probability depends on the network

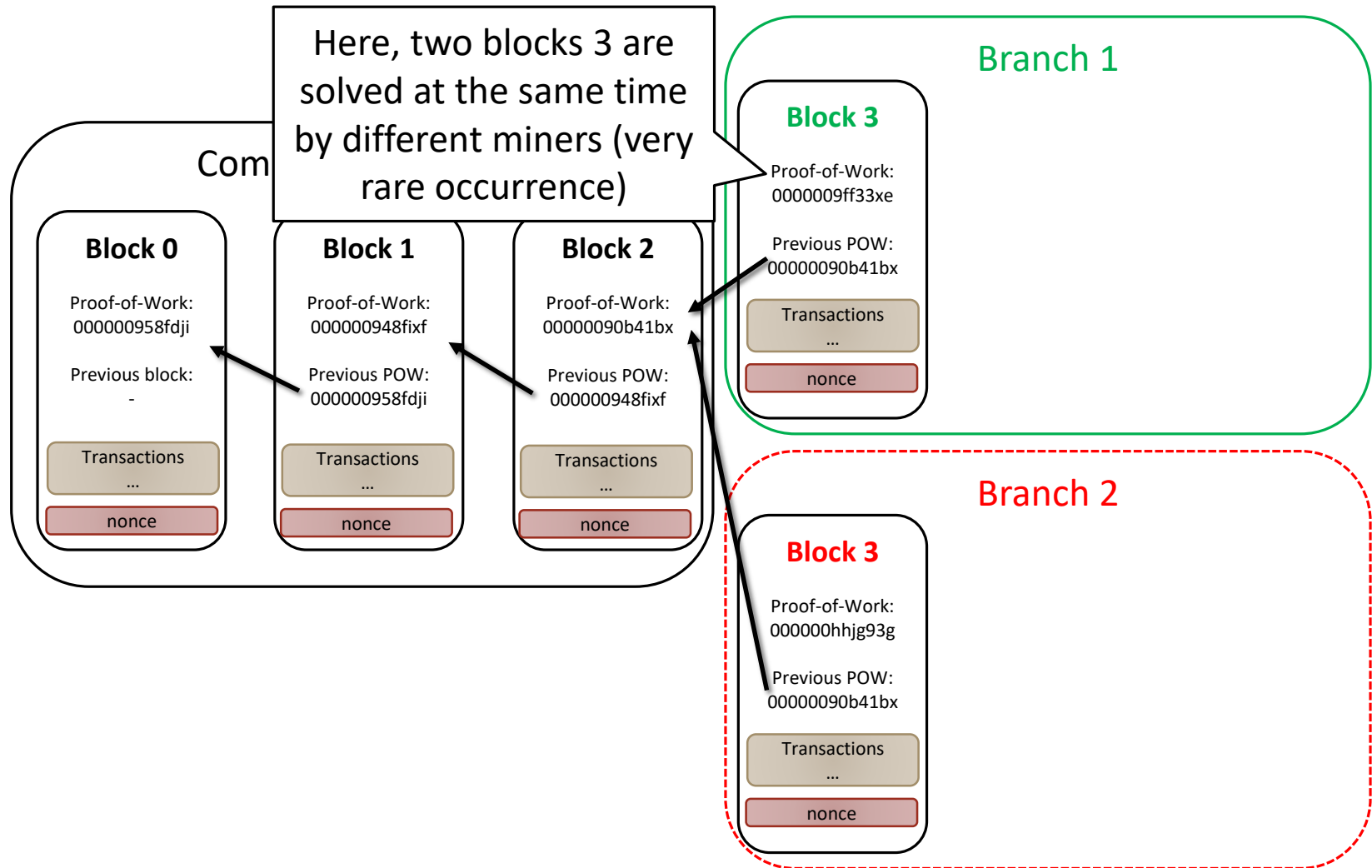
Probabilistic convergence over time is proven when using the longest chain for conflict resolution

- The probability of a block being non-final decreases exponentially with the number of later blocks stored in the chain
- The standard client sends a confirmation after six later blocks stored in the chain
- Takes an order of one hour in practice

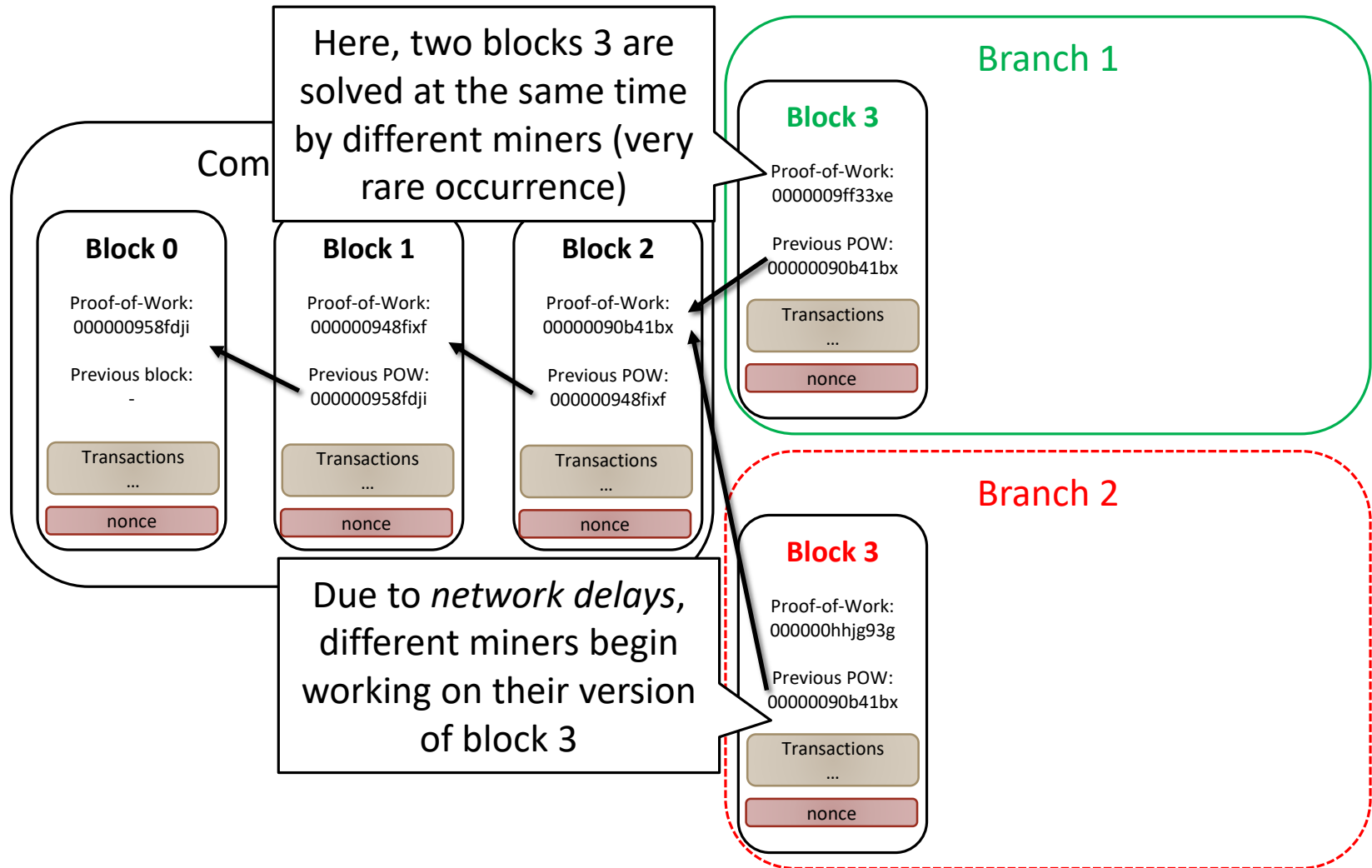
Branching



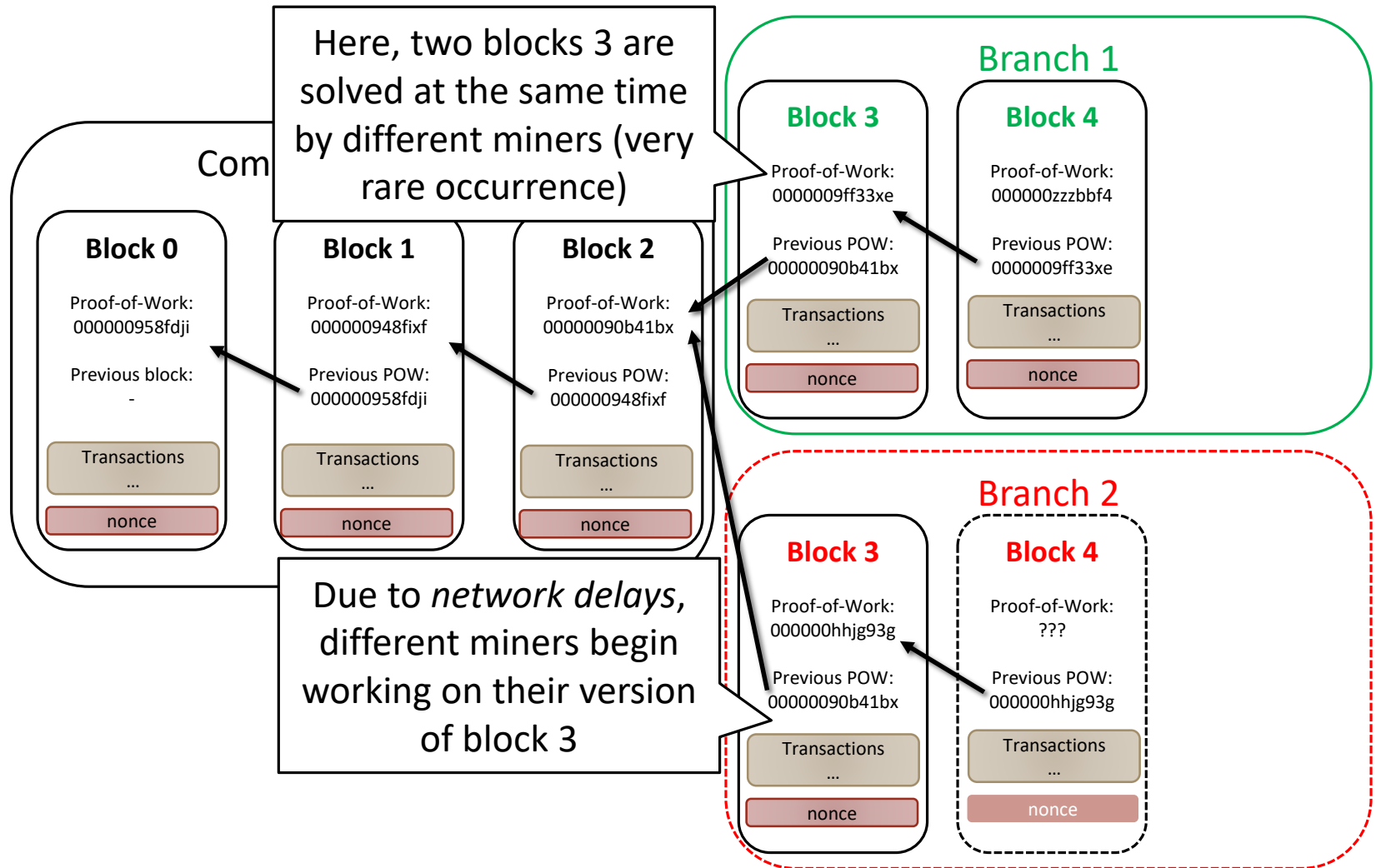
Branching



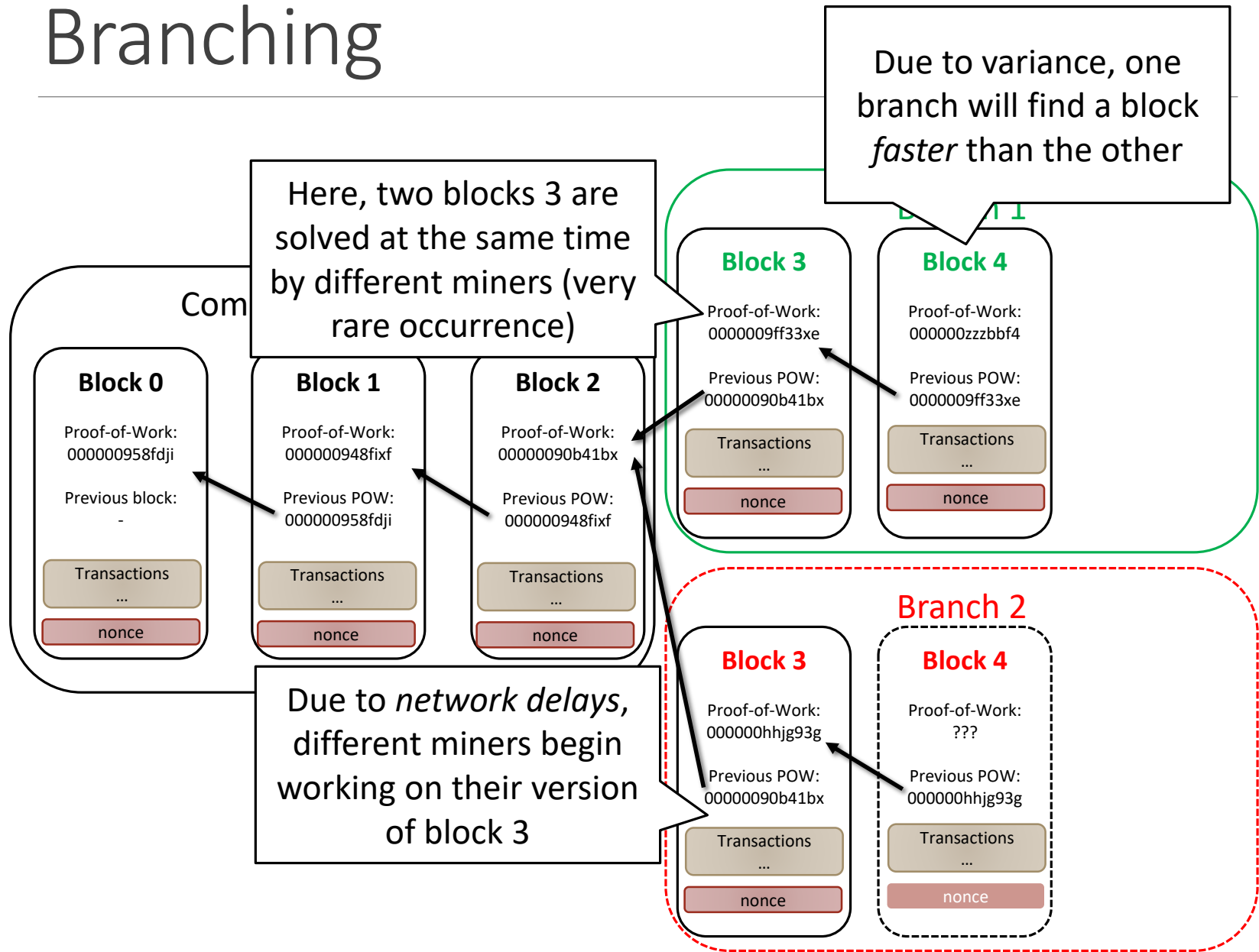
Branching



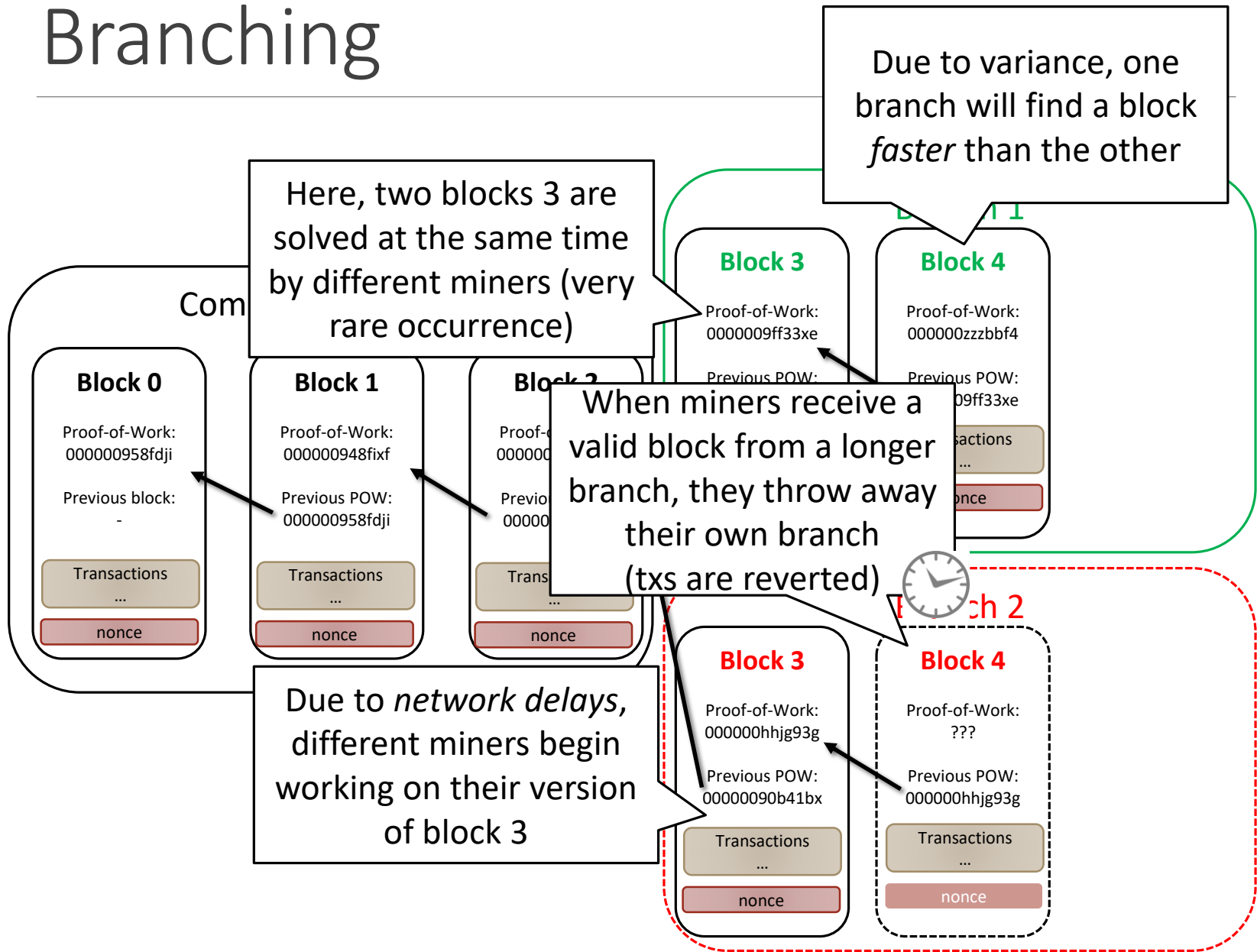
Branching



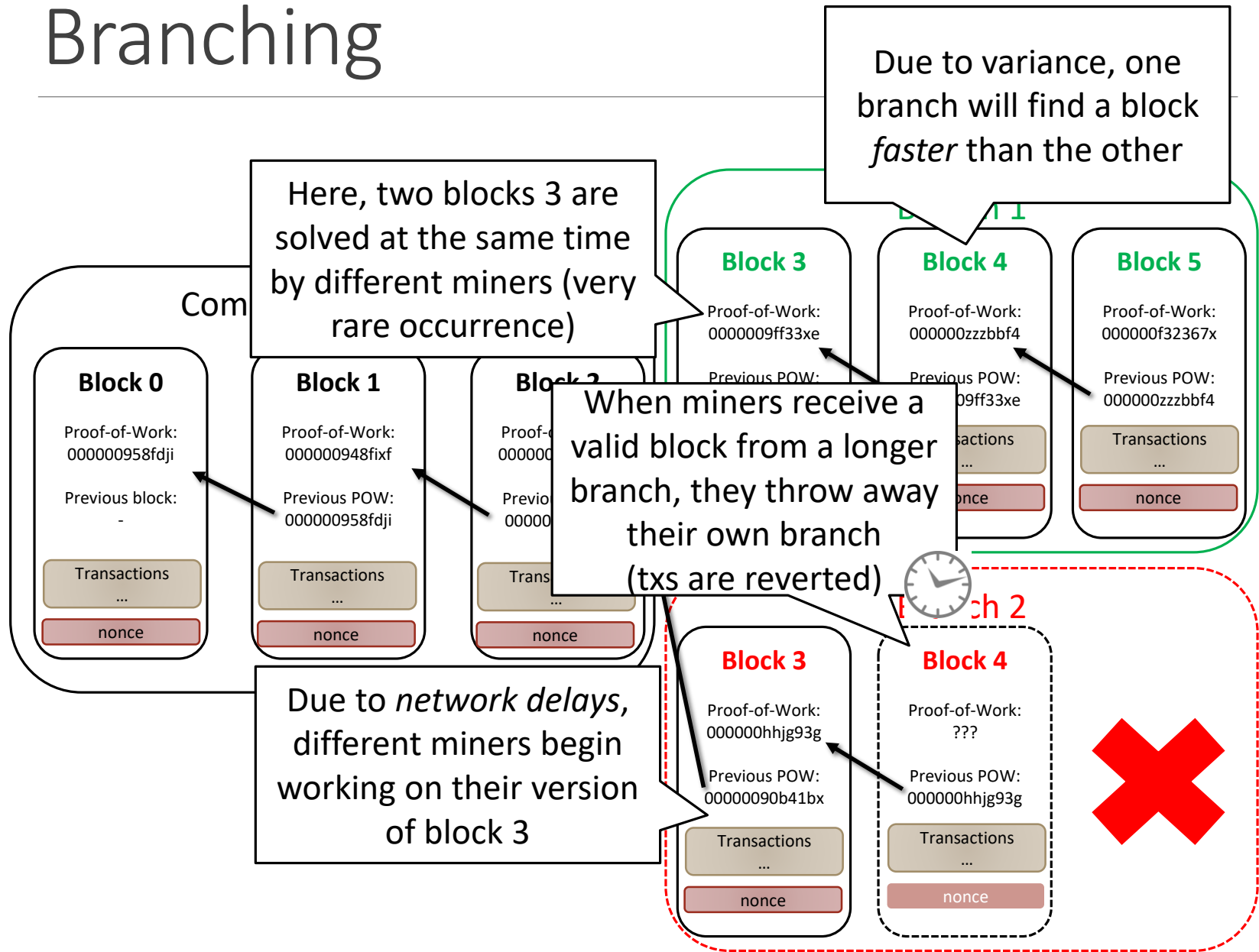
Branching



Branching

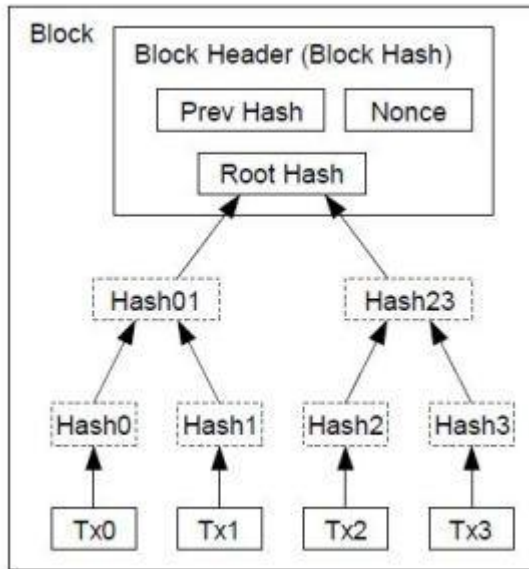


Branching



Data Structure within a Block

Merkle Tree



- ❑ To avoid hashing the entire block data when computing PoW, only the *root hash* of the Merkle tree is included.
- ❑ For users without a full copy of the blockchain, *simple payment verification (SPV)* is used to verify if a specific transaction exists.
 - ❑ A *Merkle proof* only requires the transaction itself, block root hash, and all of the hashes going up along the path from the transaction to the root, e.g., Hash01, Hash2 (for Tx3).
- ❑ Spent transactions can be *pruned* in the local copy, leaving only the necessary intermediate nodes to save space.
 - ❑ E.g., if both Tx0 and Tx1 are spent, we can prune everything under Hash01

Data manipulation and queries

Reading the ledger and verifying its correctness is straightforward but time-consuming

- Publicly available, no access control whatsoever
- A copy is held by many users (over 10,000 today)
- Users are encouraged to download and run a verification

Transparency is a boon for data integrity but a bane for privacy

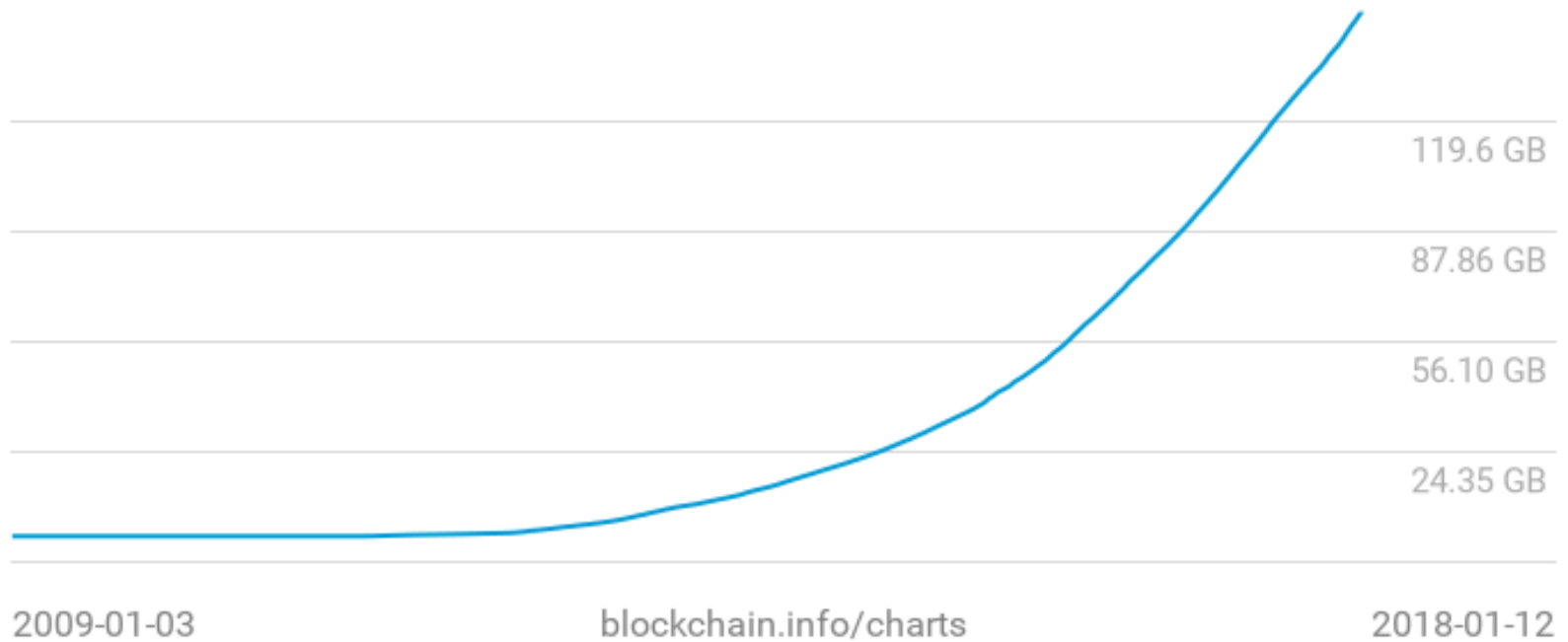
- Public keys are used as user identities
- A key can serve as a pseudonym, difficult to link to real identity
- A user can use a different pseudonym for each transaction
- The main threat comes from analyzing the history of transactions and linking them together

Temper-resistance is mostly a blessing

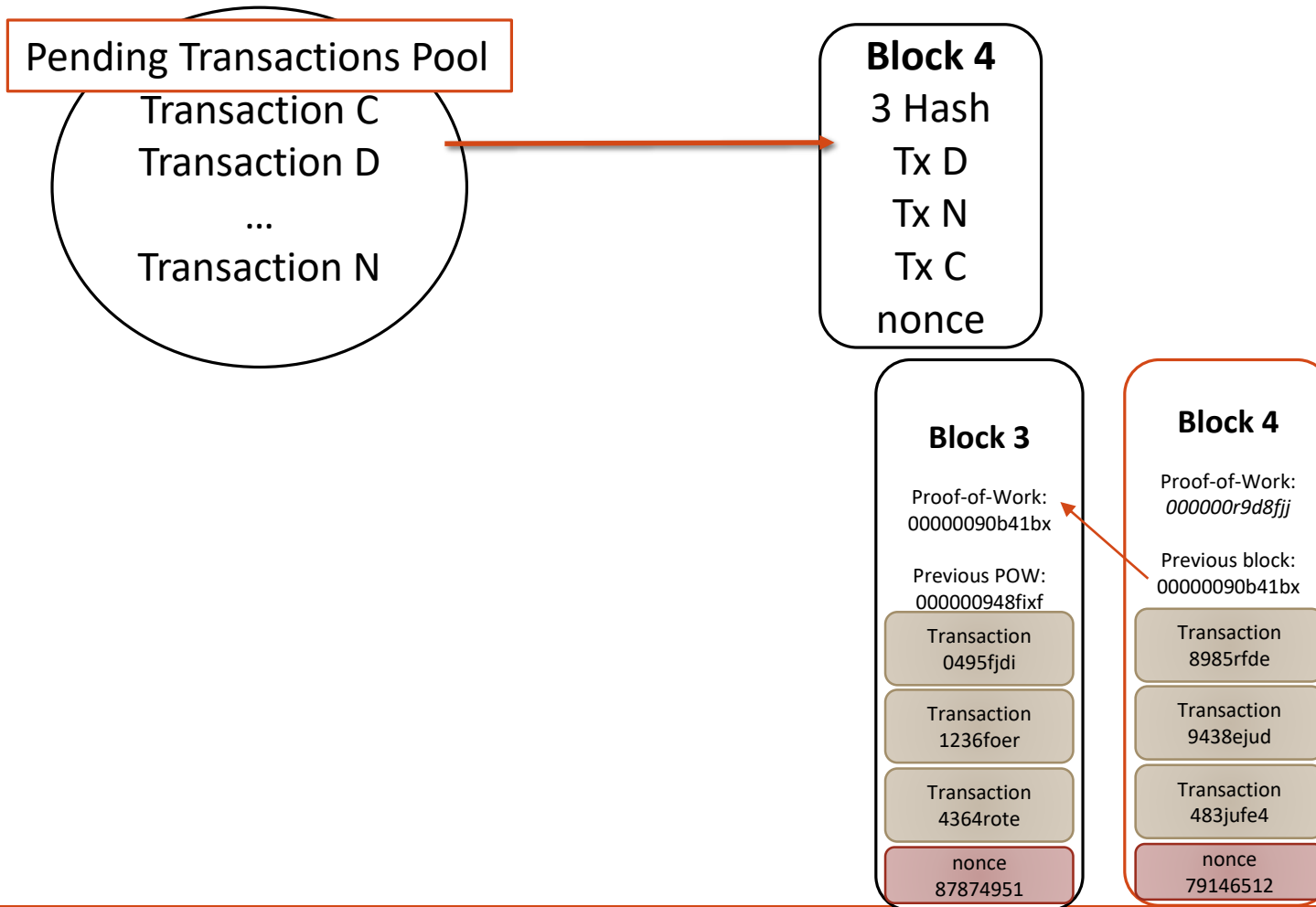
- But also a curse: difficult to compact or prune the history

Size of ledger

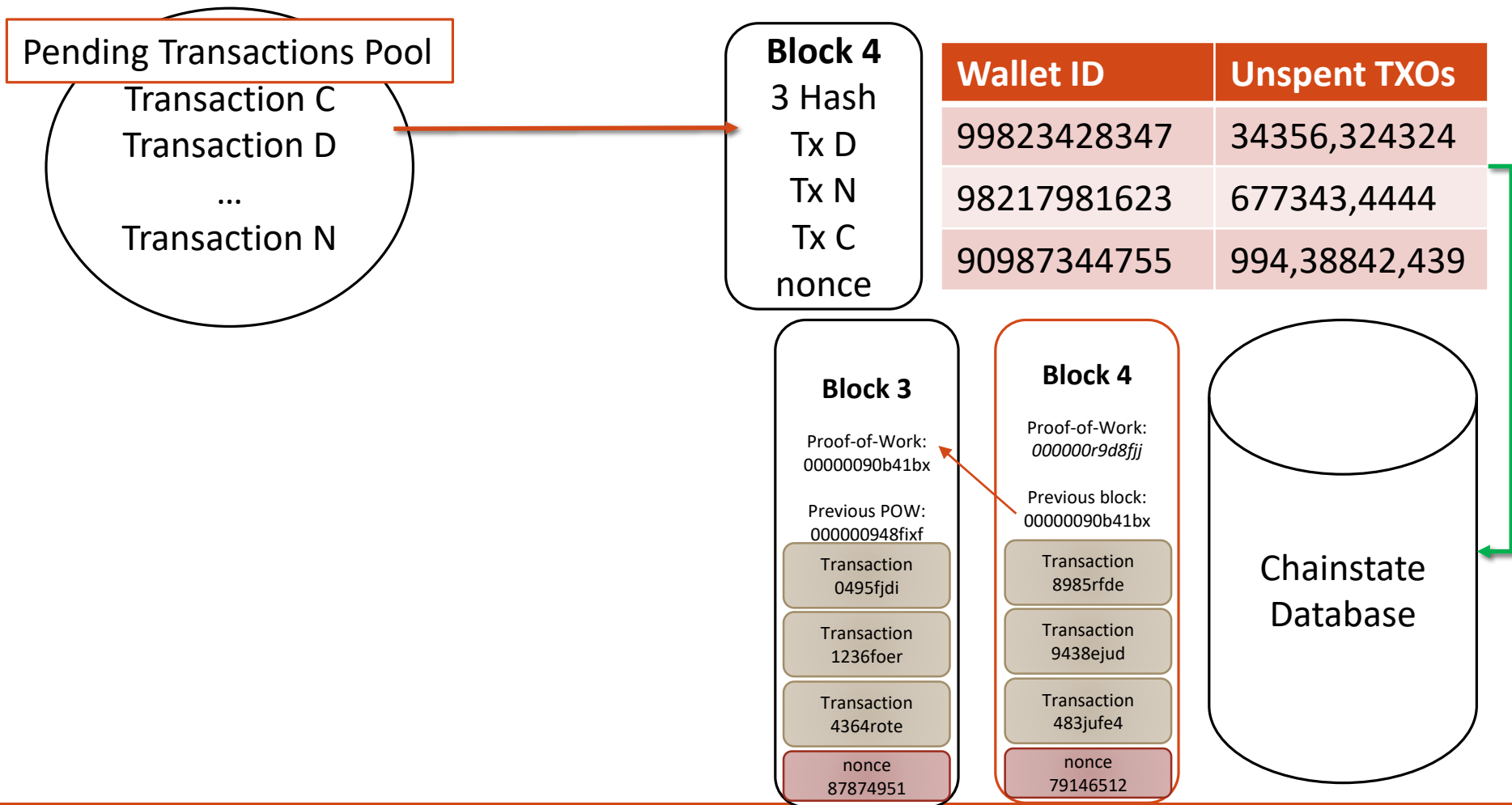
Blockchain Size
151.2 GB



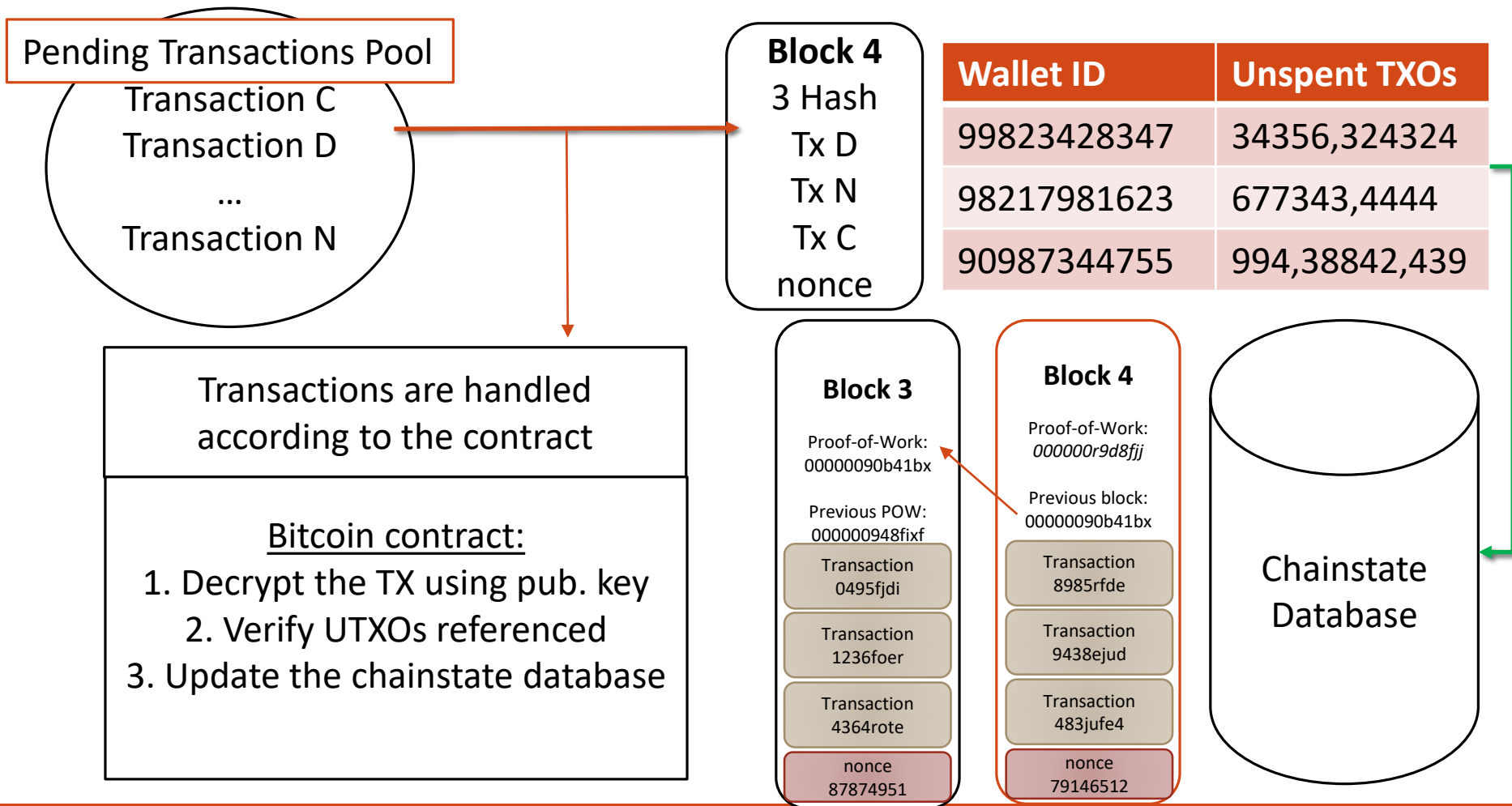
Bitcoin's "contracts"



Bitcoin's "contracts"



Bitcoin's "contracts"



Business logic in Bitcoin

The output additionally includes a verification script

- representing the conditions under which the output can be redeemed, i.e., included as an input in a later transaction
- A typical script: “can be redeemed by a public key that hashes to X, along with a signature from the key owner”

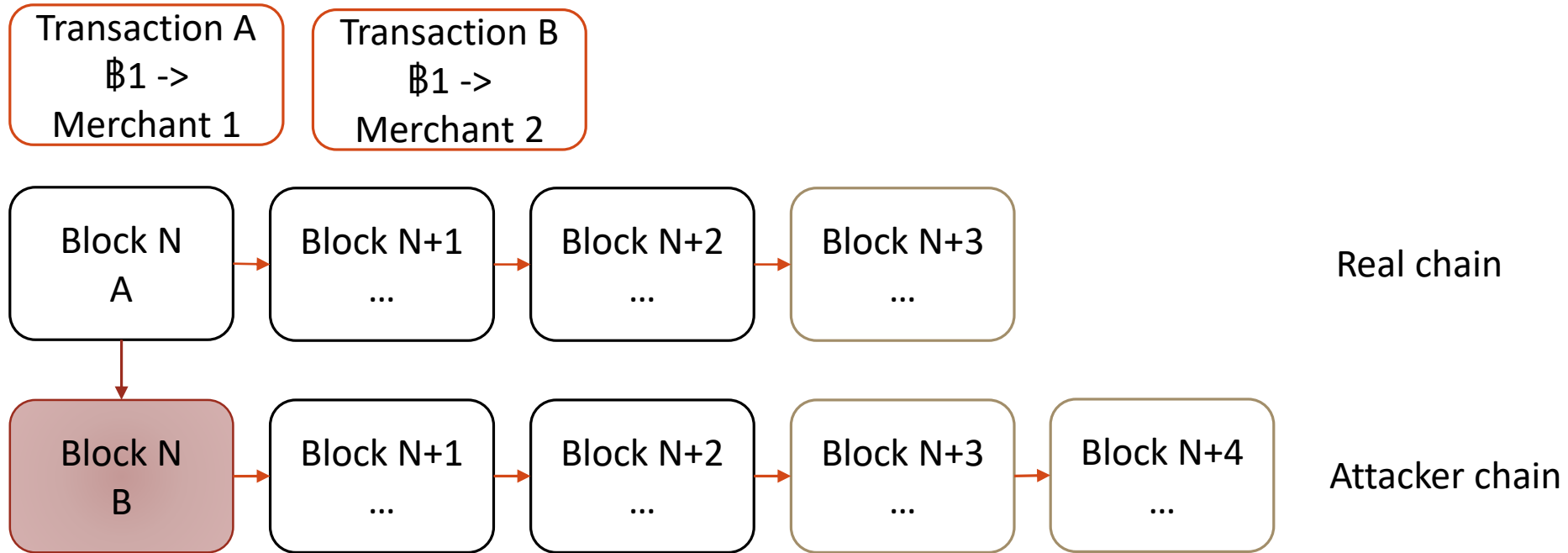
There is also a redeeming script attached to the input

Both scripts are executed by whoever verifies the redeeming transaction, such as a proposer

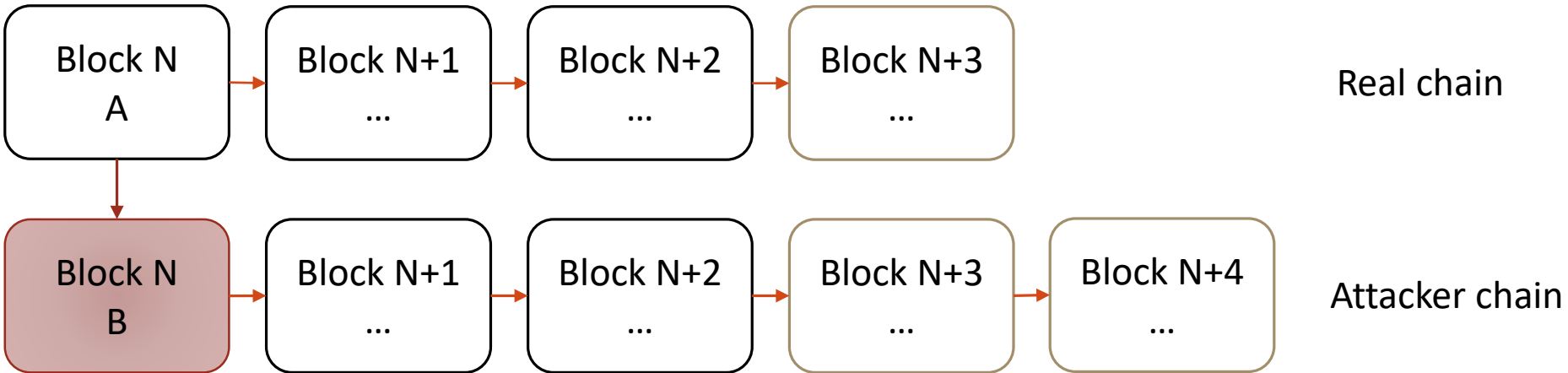
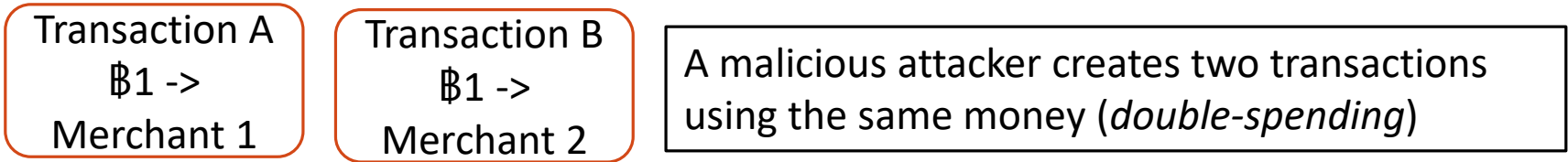
A script language with an order of 200 commands

- Support for cryptographic primitives
- Rather ad-hoc

Preventing double spending



Preventing double spending

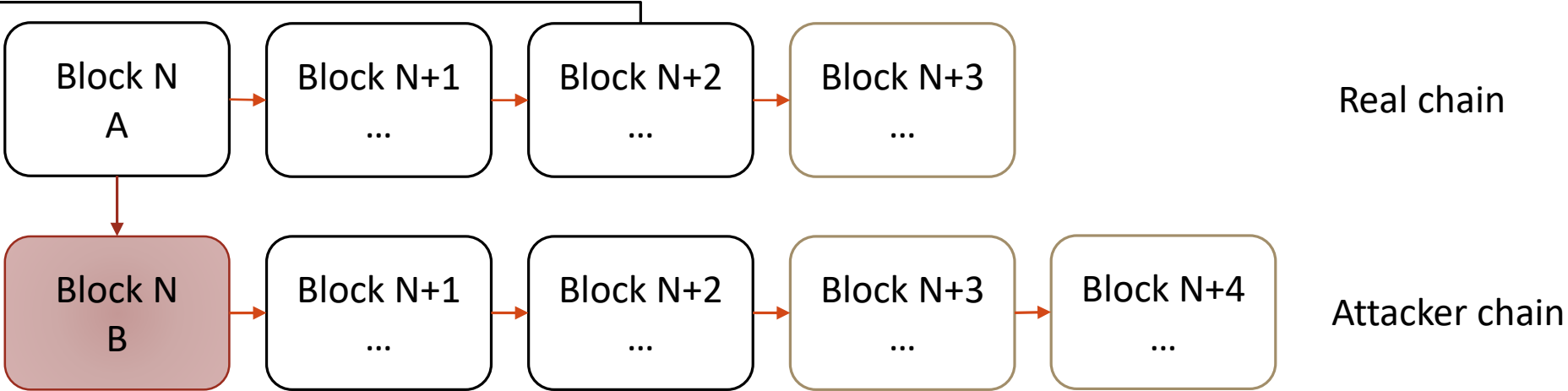


Preventing double spending

Transaction A
฿1 ->
Merchant 1

Transaction B
฿1 ->
Merchant 2

A malicious attacker creates two transactions using the same money (*double-spending*)



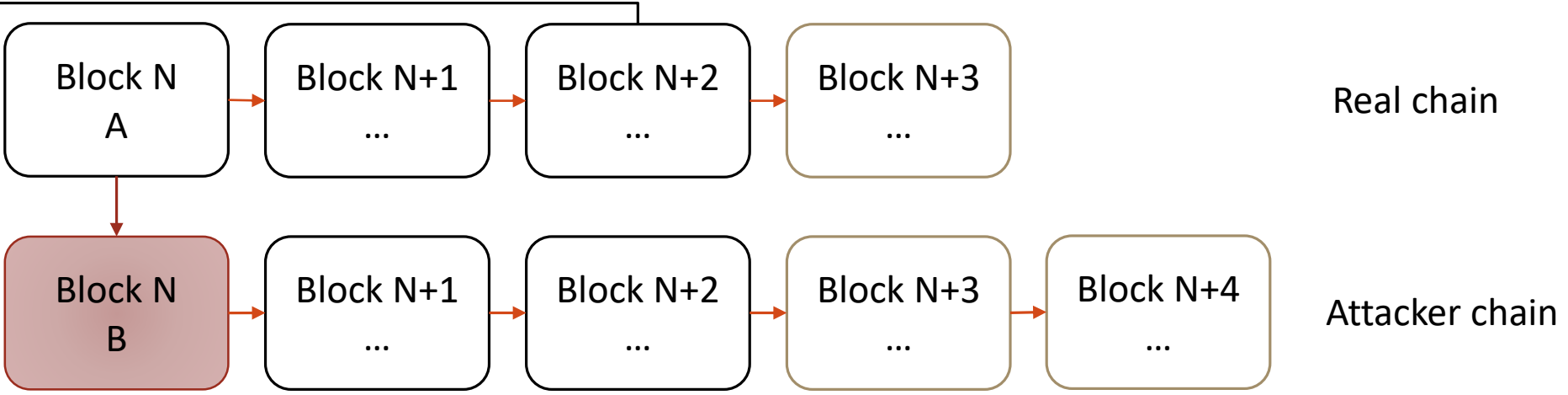
Suppose A is added to block N, and merchant 1 confirms the transaction after waiting for a few blocks

Preventing double spending

Transaction A
฿1 ->
Merchant 1

Transaction B
฿1 ->
Merchant 2

A malicious attacker creates two transactions using the same money (*double-spending*)



Suppose A is added to block N, and merchant 1 confirms the transaction after waiting for a few blocks

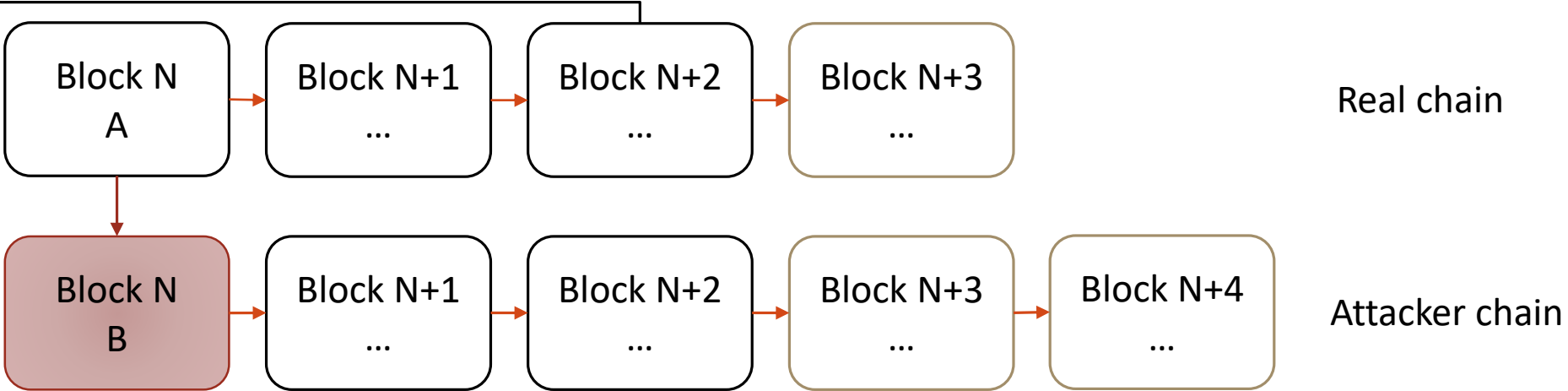
It must replace A with B in N, and solve the modified puzzles for the blocks faster than the real chain grows so that it can become longer

Preventing double spending

Transaction A
฿1 ->
Merchant 1

Transaction B
฿1 ->
Merchant 2

A malicious attacker creates two transactions using the same money (*double-spending*)



- The “Magic Watch” is the *continuous generation* of blocks in the main chain which *limits the amount of time* an attacker has to create its own chain.
- If the attacker owns *>51% of the power* in the network, the “Magic Watch” gives *enough time* to the attacker to *tamper the data!*

It must replace A with B in N, and solve the modified puzzles for the blocks faster than the real chain grows so that it can become longer



Other attacks (cursory)

Stealing bitcoins is hard because of digital signatures

- If, however, someone accumulates a lot of bitcoins, it becomes a prime target

Denial-of-service on the entire Bitcoin network is hard because of proof-of-work

- Still possible to bombard the network with invalid transactions

Starving a specific user: does not work if there is a sufficient number of honest miners

- Possibility to blackmail users with high tx fees if miners are “rational”
- cf. feather forking attacks

Economic attacks: selfish mining

- Attempts to maintain private branches longer than the public branch
- Releasing a longer private branch causes honest miners to lose revenue, “stolen” by the attacker
- 25% attack with “rational” miners

Limitations of Bitcoin

Limitations of Bitcoin

Limited expressiveness

- Cryptocurrency only
- Each app requires new platform (e.g. NameCoin, PrimeCoin, CureCoin)

Slow block time (10 mins)

- Also slow confirmation time (1+ hour for 6 confirmations)

Hard/Soft forks

- Updates to the code cause forks
- Hard forks are not compatible
- Duplicated money
- Bitcoin: Cash, Classic, Gold

Limitations of Bitcoin

Limited expressiveness

- Cryptocurrency only
- Each app requires new platform (e.g. NameCoin, PrimeCoin, CureCoin)

Slow block time (10 mins)

- Also slow confirmation time (1+ hour for 6 confirmations)

Hard/Soft forks

- Updates to the code cause forks
- Hard forks are not compatible
- Duplicated money
- Bitcoin: Cash, Classic, Gold

Slow transaction rate

- 7 transactions/second
- VISA Network: 2000 tps (average)
- Limited block size (Segwit2x: 1MB -> 2MB)

Weaknesses of proof-of-work

- Environmental impact: ~1000x more energy than credit card
- Currently 43th in energy consumption (comparable to Switzerland)

Long bootstrap time for a miner

- Full ledger: 164 GB (2018/04)
- CPU/IO cost to verify each transaction/block
- Takes hours/days

Blockchain Systems

ETHEREUM

HYPERLEDGER



ETHEREUM

Managing entity: Ethereum Foundation

- Major players: Deloitte, Toyota, Microsoft, ...

Focus: Open-source, flexible, platform

- Cryptocurrency: 1 Ether = $1e18$ Wei (502 USD, 2018/04)
- Smart contracts: Solidity, Remix (Web IDE), Truffle (Dev./Test), *Viper*
- Ethereum Virtual Machine (EVM)
- Permissionless (public) ledger: Proof-of-Work, *Proof-of-Stake (Casper)*

Notes

- GHOST Protocol: Merging of branches
- DOA Event: \$150 million lost, hard forked into Eth. Classic
- *Scalability: Sharding and Plasma*

Evolution in business logic

Proliferation of Bitcoin spawn-offs

- Digital currency is not the only electronic object of value
 - Documents: authorizations, legal, diploma, design, various deliverables
 - Software
- Support for extended financial applications such as crowdfunding
- Support for multi-party escrow transactions

Ethereum envisioned that a single platform supporting the above is better than hundreds of specialized systems

- Provided a verifiable Turing-complete script language
- With script templates
- Scripts can be stateful, with a state stored on the chain

Benefits of smart contracts

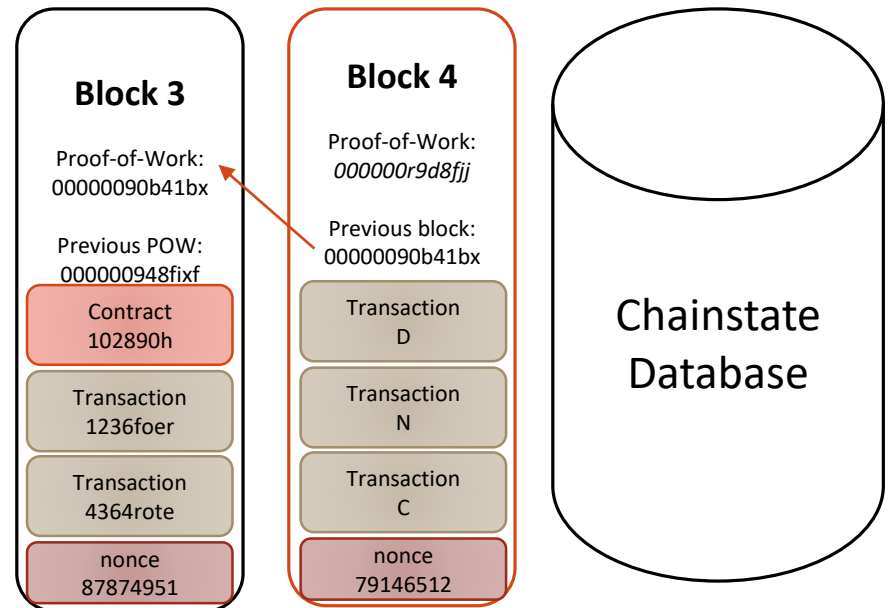
Compared to a human intermediary

- Cheaper
- Open and transparent program that fulfils the contract and does nothing else
 - Does not peek into your data
- Highly resistance to attacks

Compared to distributed databases

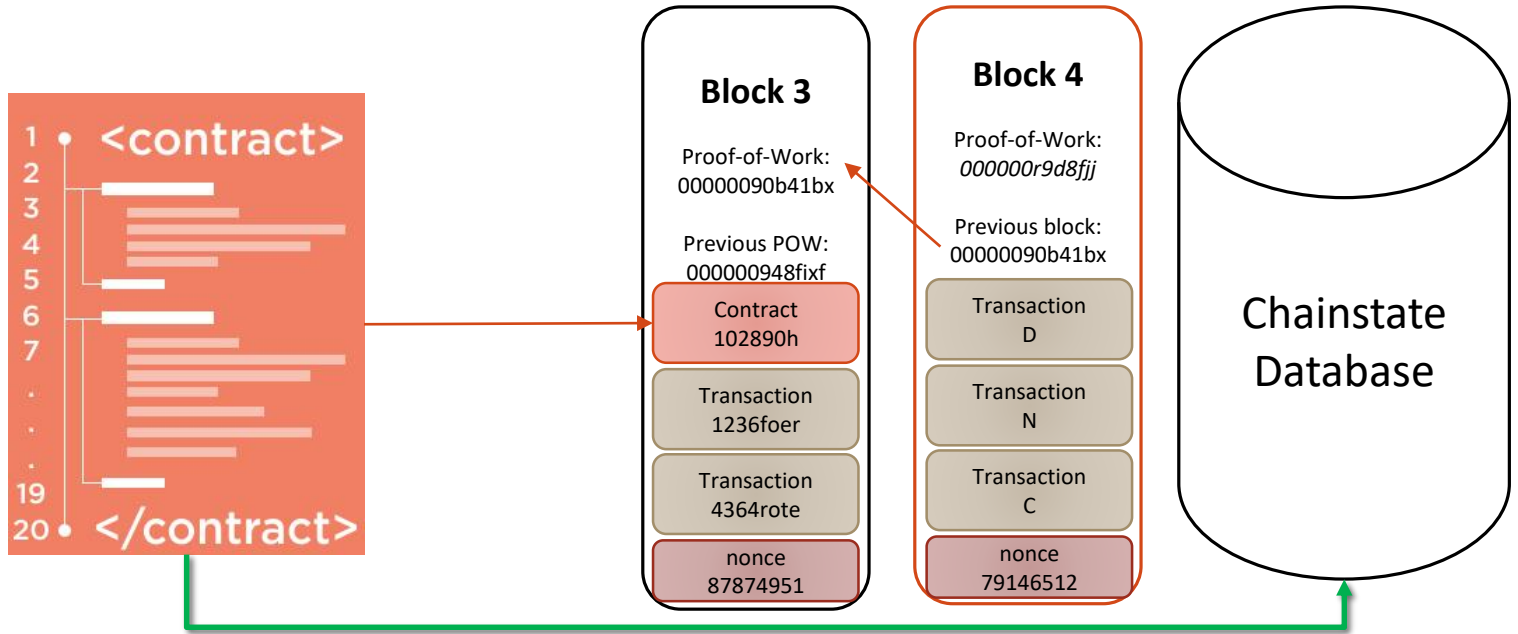
- Rule-based rather than data-based
- Rich language and (relative) easy of development
- The collection of rules is transparent and reusable
- May initiate and play an active role in the communication
- May integrate and fuse data from multiple sources

Smart Contracts



Smart Contracts

- Contracts contain *executable bytecode*
- Created with a blockchain tx
- Contracts have internal storage

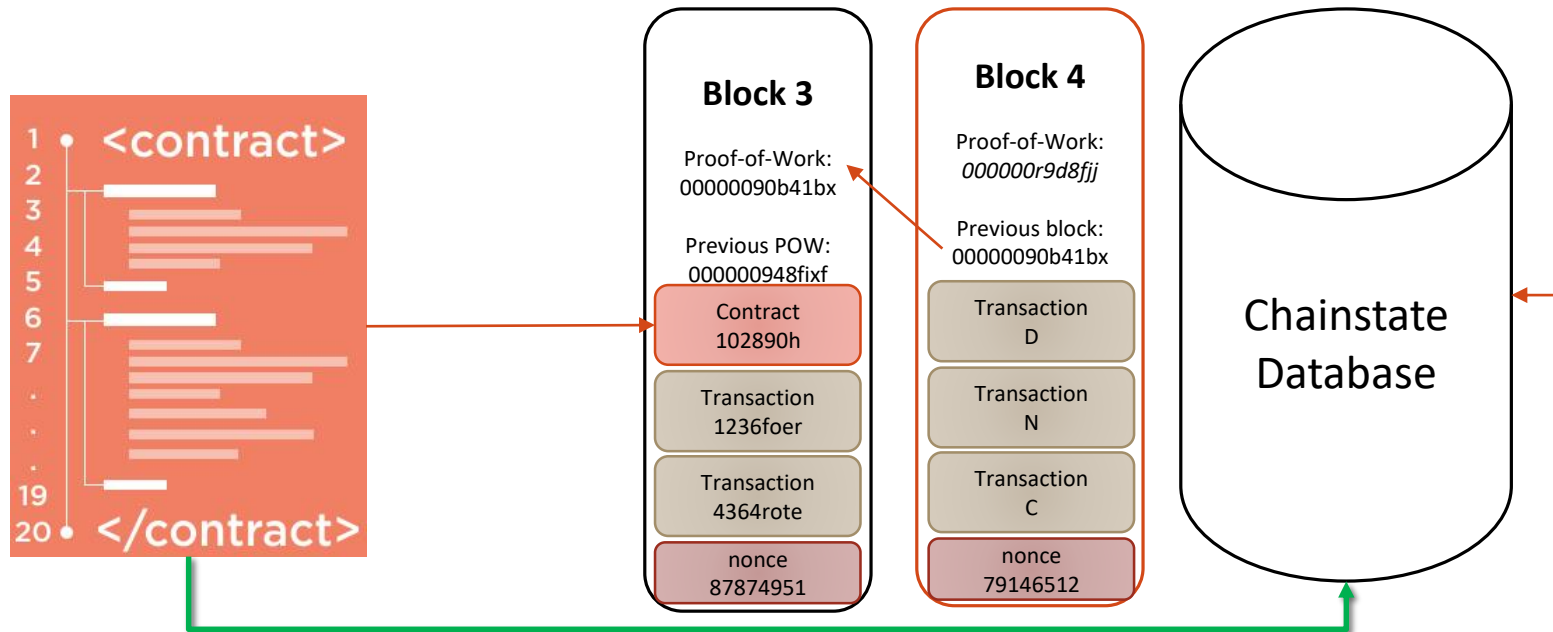


Smart Contracts

- Contracts contain *executable bytecode*
- Created with a blockchain tx
- Contracts have internal storage

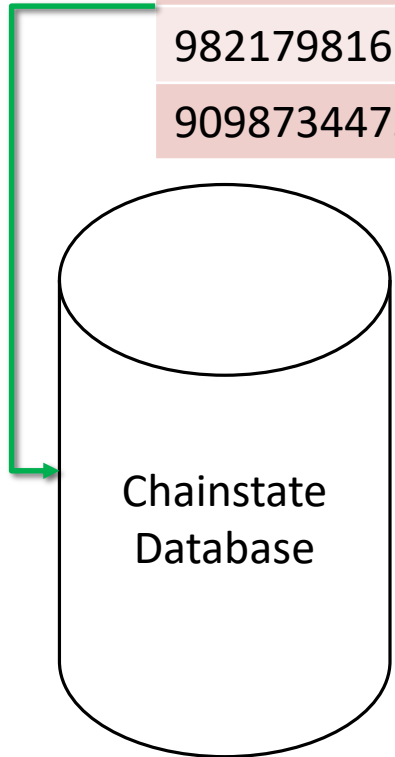
Contracts execute when triggered by a transaction (or by another contract)
 Execution time is limited by *gas*
Example: Land registry

Wallet ID	Held Titles
99823428347	34356,324324
98217981623	677343,4444
90987344755	994,38842,439



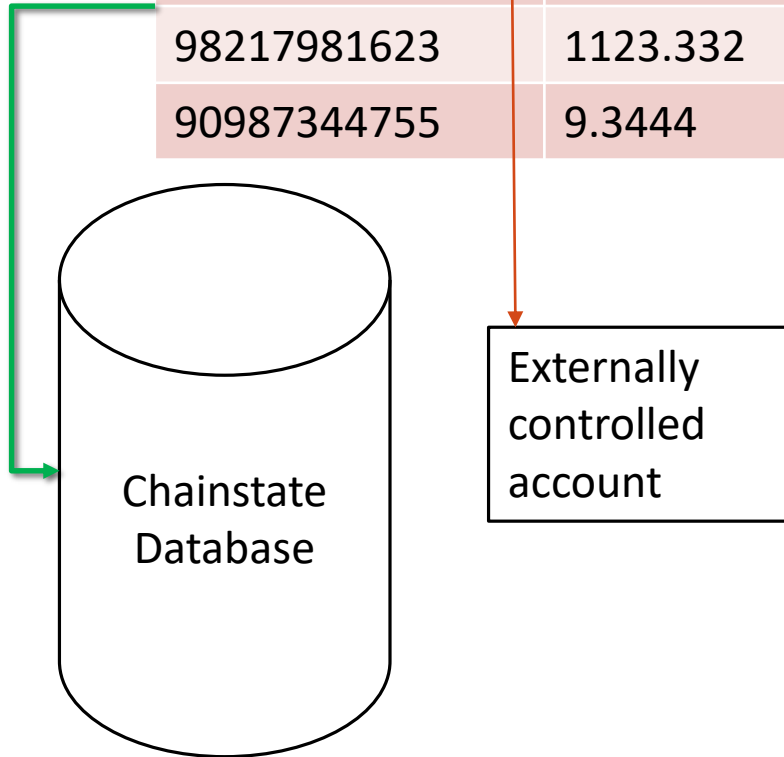
Account State (“World State”)

Wallet ID	Balance	Code Hash	Internal State
99823428347	45.12	-	99554HGJ
98217981623	1123.332	9ERU12T4	3453ADFG
90987344755	9.3444	0490CNDJ	132GJR4



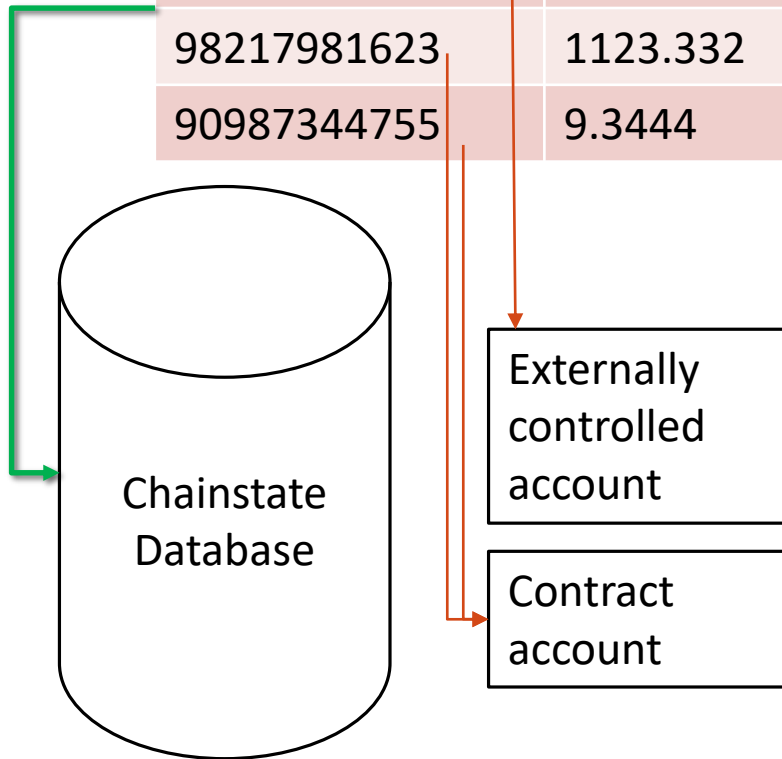
Account State (“World State”)

Wallet ID	Balance	Code Hash	Internal State
99823428347	45.12	-	99554HGJ
98217981623	1123.332	9ERU12T4	3453ADFG
90987344755	9.3444	0490CNDJ	132GJR4



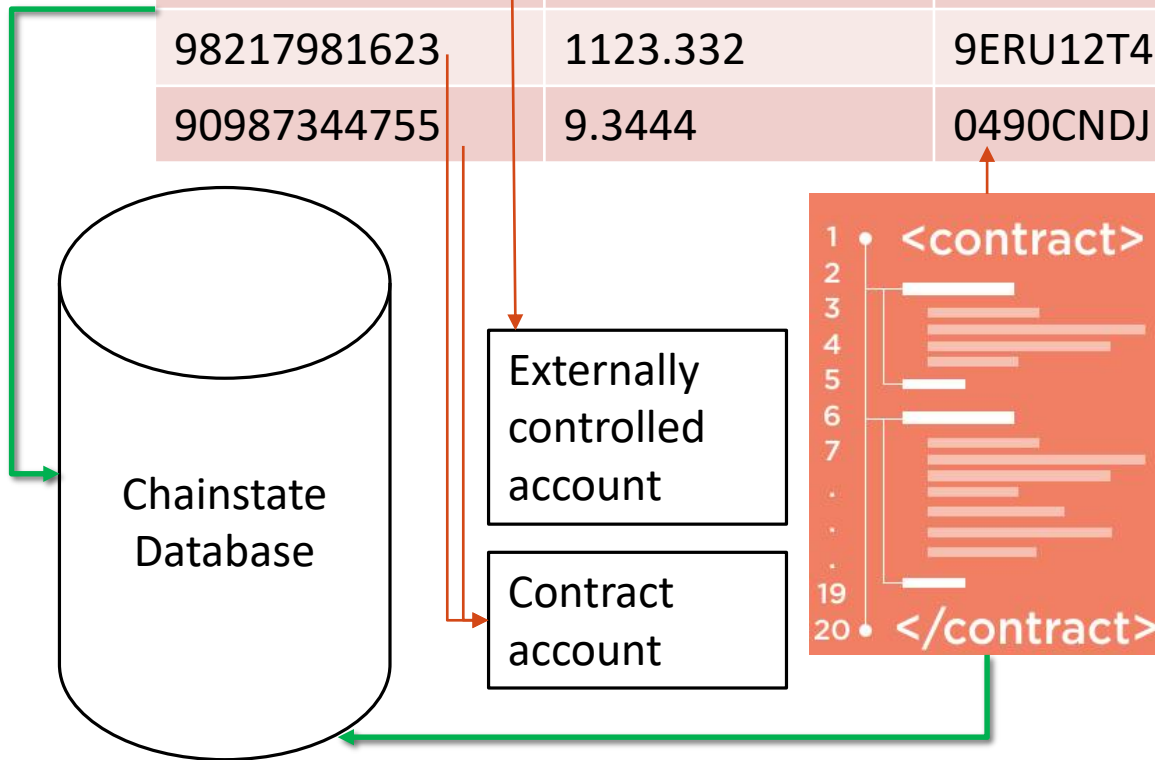
Account State (“World State”)

Wallet ID	Balance	Code Hash	Internal State
99823428347	45.12	-	99554HGJ
98217981623	1123.332	9ERU12T4	3453ADFG
90987344755	9.3444	0490CNDJ	132GJR4



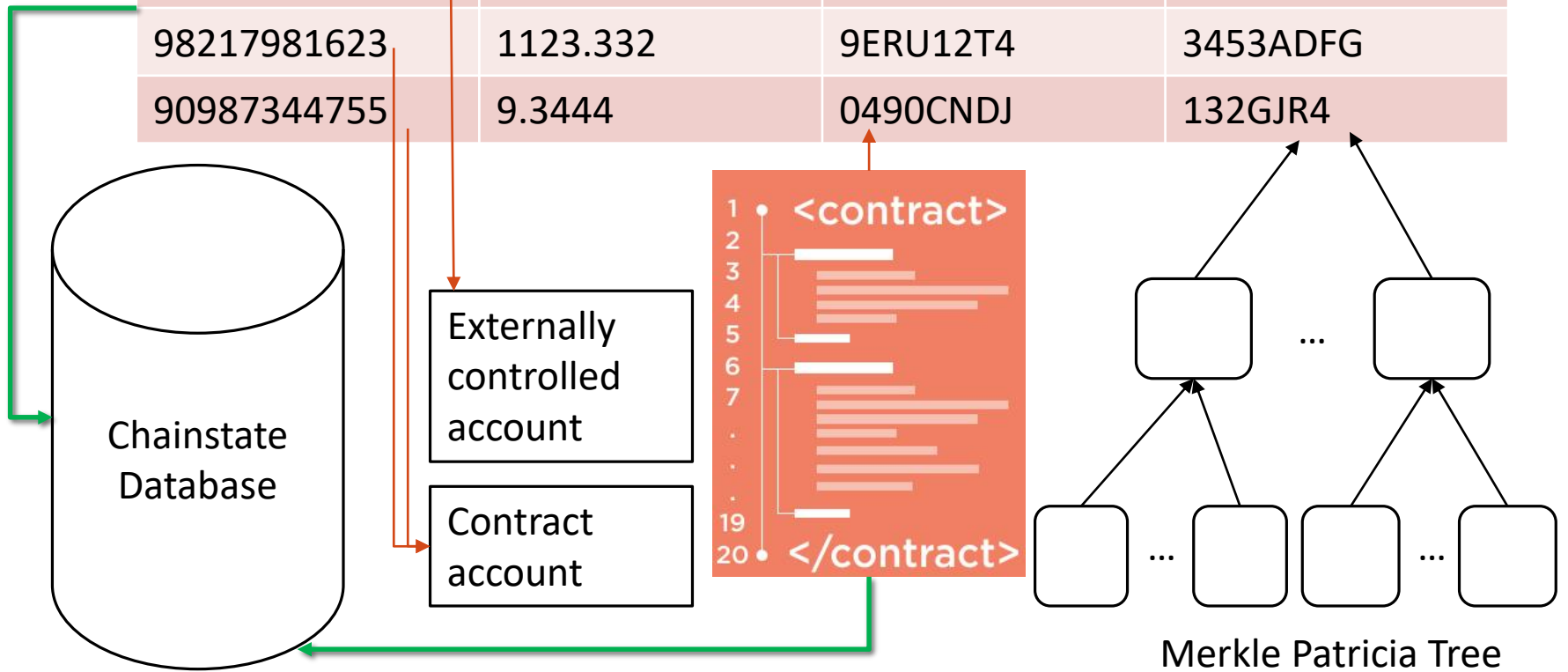
Account State (“World State”)

Wallet ID	Balance	Code Hash	Internal State
99823428347	45.12	-	99554HGJ
98217981623	1123.332	9ERU12T4	3453ADFG
90987344755	9.3444	0490CNDJ	132GJR4

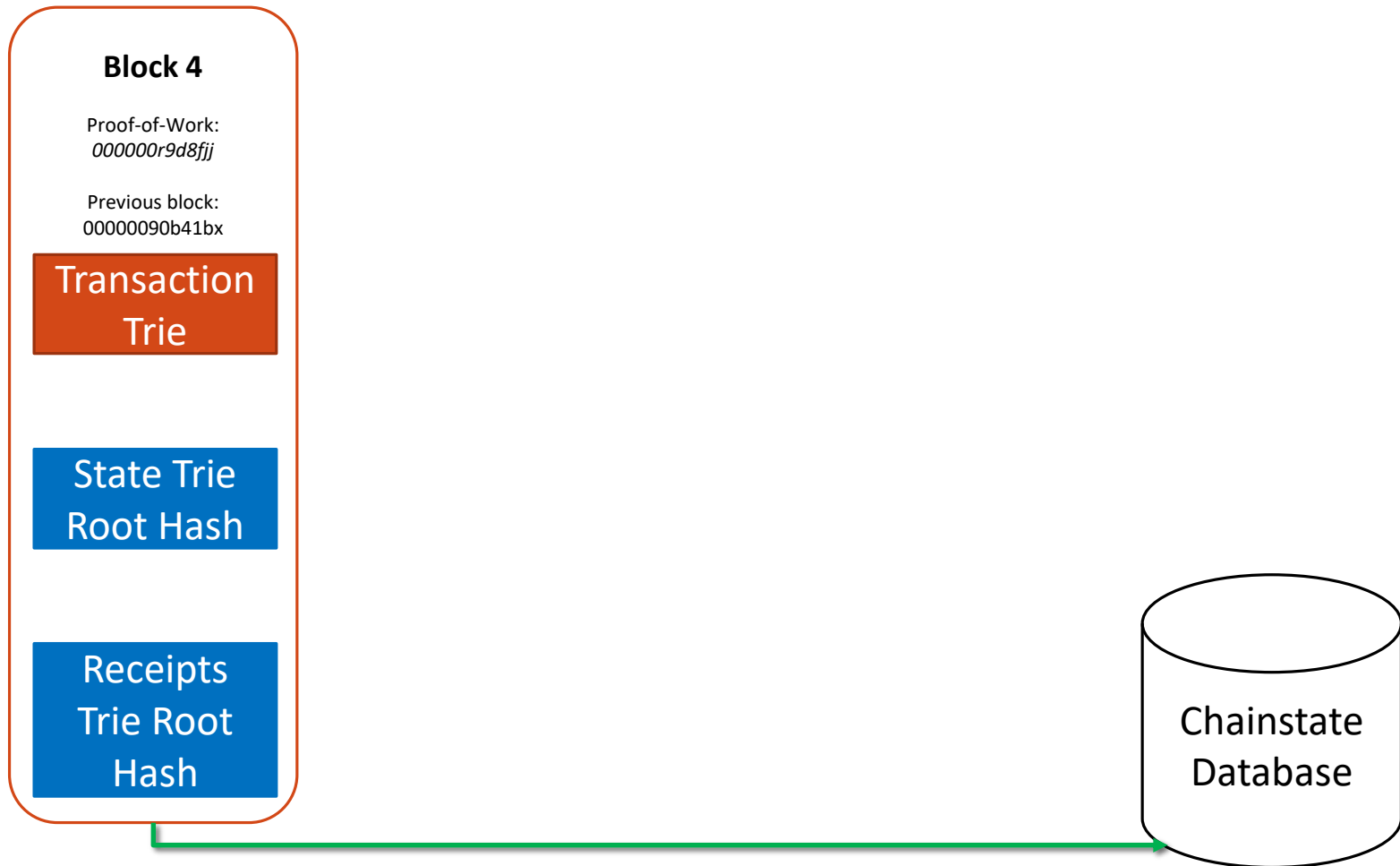


Account State (“World State”)

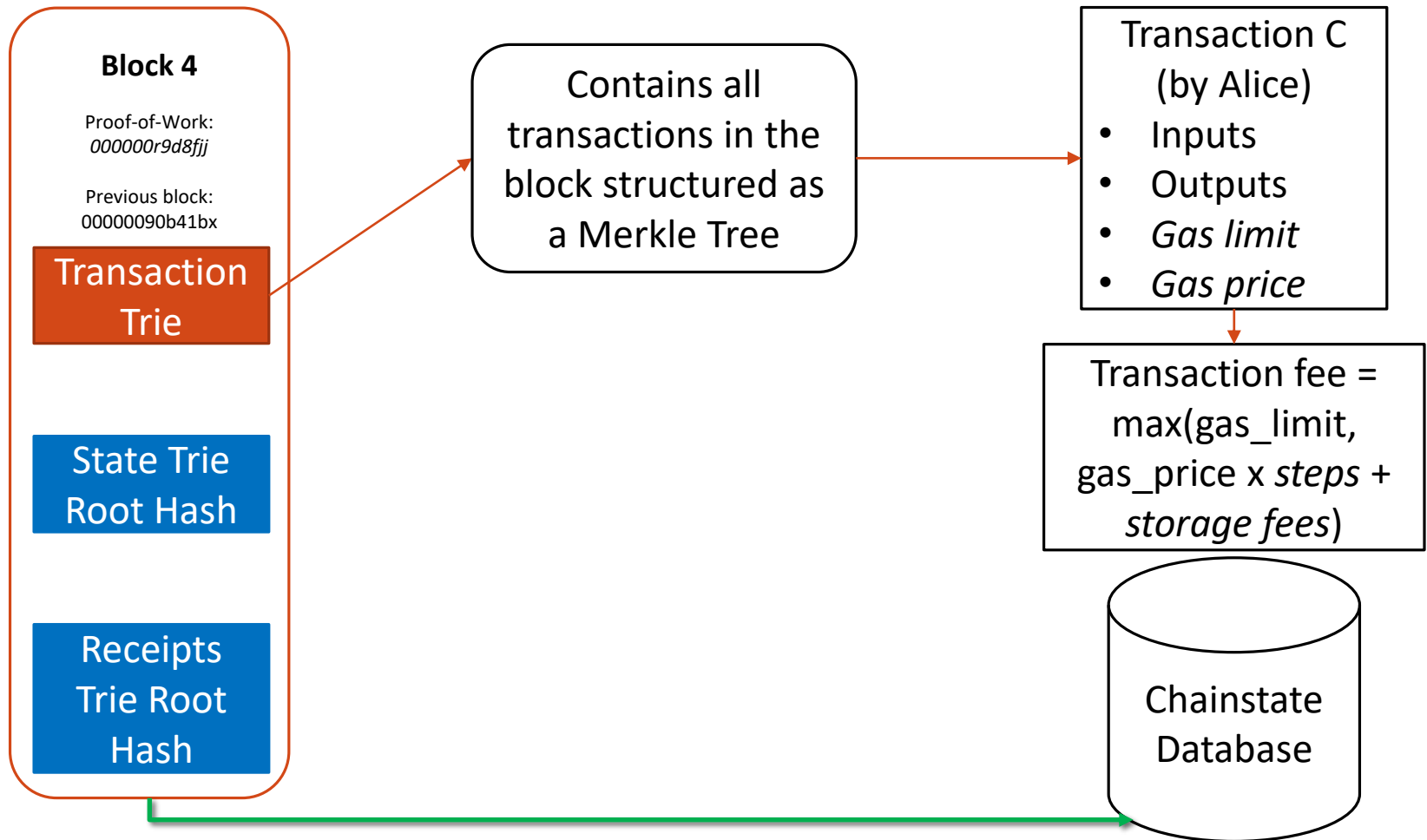
Wallet ID	Balance	Code Hash	Internal State
99823428347	45.12	-	99554HGJ
98217981623	1123.332	9ERU12T4	3453ADFG
90987344755	9.3444	0490CNDJ	132GJR4



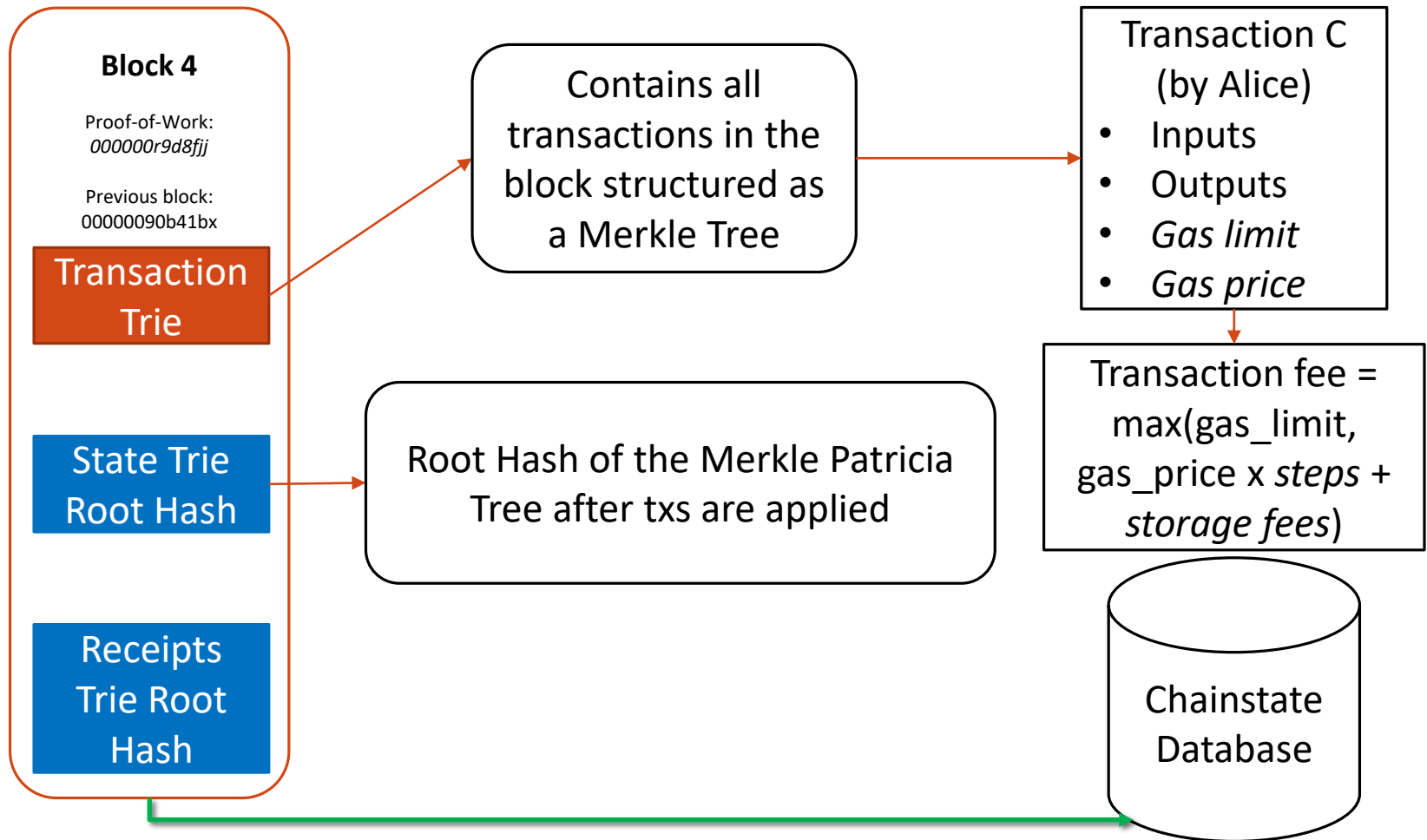
Execution and Mining



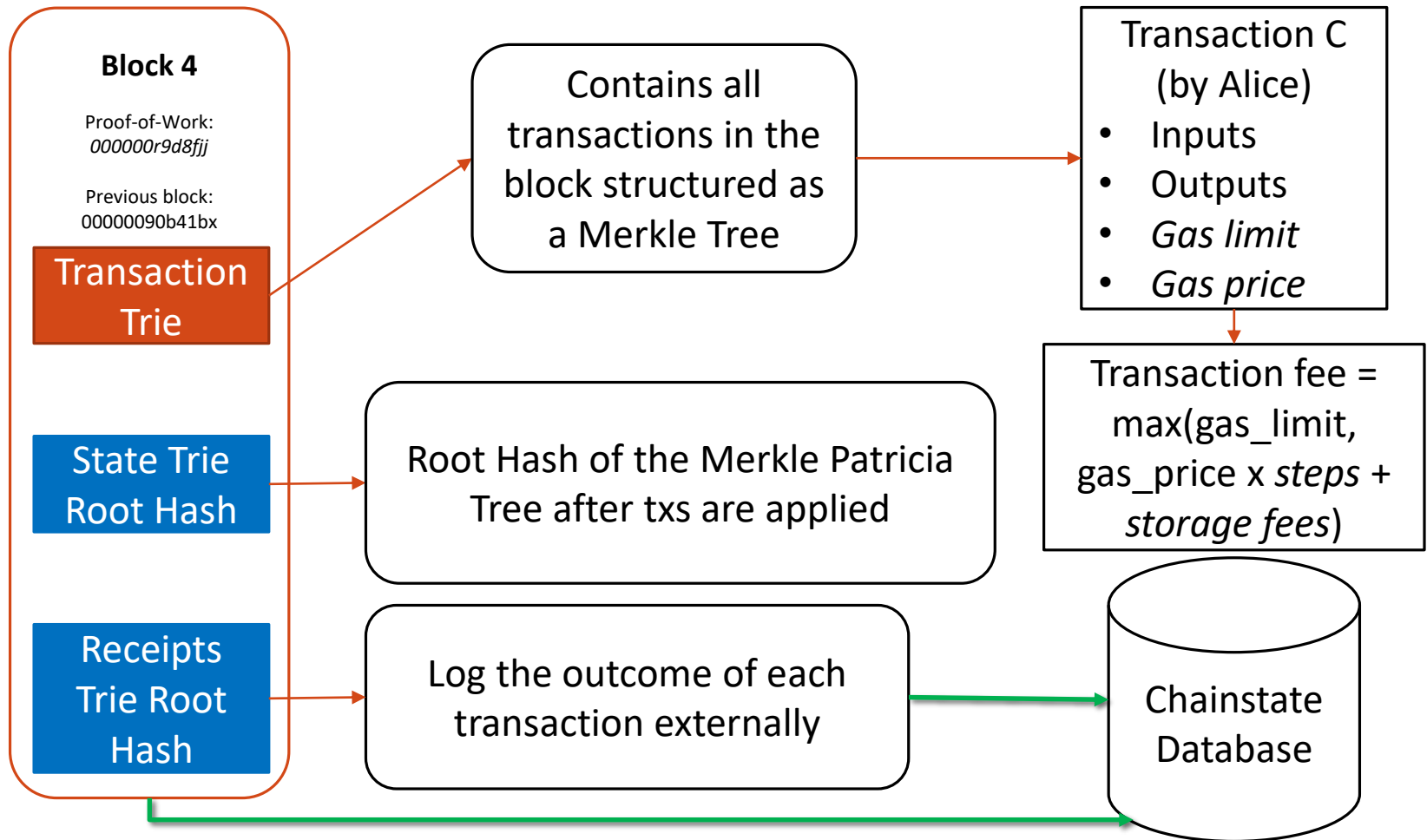
Execution and Mining



Execution and Mining



Execution and Mining



Comparison with Bitcoin

	Bitcoin	Ethereum
Transactions	Transfer of bitcoins	<i>Contract creation, transfer of ether, contract calls, internal transactions</i>
Accounts	User wallets	Externally owned accounts, <i>contract accounts</i>
Transaction fees	Amount specified by sender	Gas calculated using sender's values
Block content	Transactions trie	Transactions, <i>State Root Hash, Receipts Root Hash</i>
Chainstate Database	World state: UTXOs for wallets	World state, <i>receipts, bytecodes for contracts</i>
Querying	Simple Payment Verification	Merkle proofs for <i>events, transactions, balance, etc.</i>



HYPERLEDGER

Managing entity: Hyperledger Consortium

- Major players: IBM, NEC, Intel, R3, ...

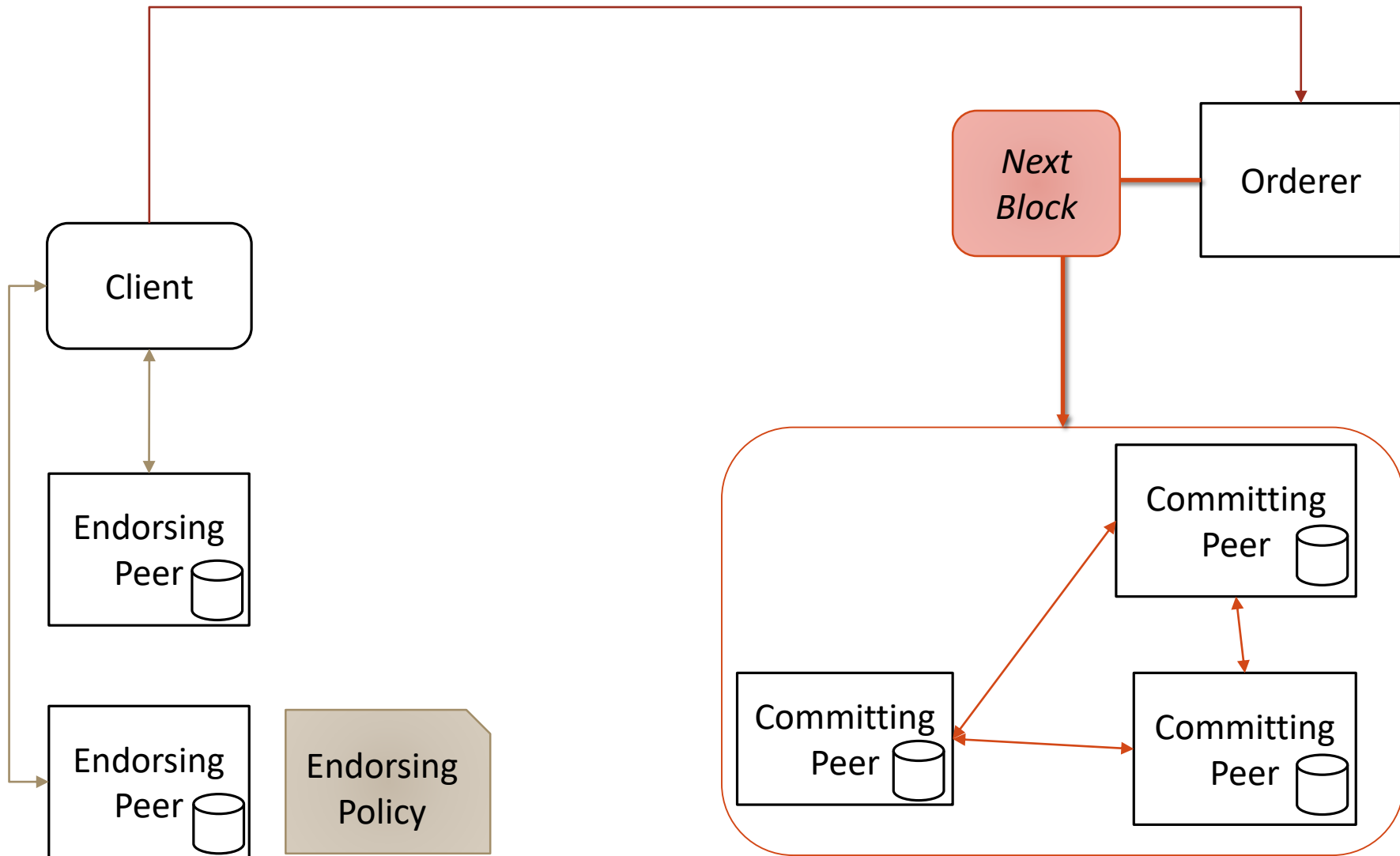
Focus: Enterprise blockchains

- Permissioned ledger (private/consortium network)
- Smart contracts
- Open-source
- World state on CouchDB, event listener

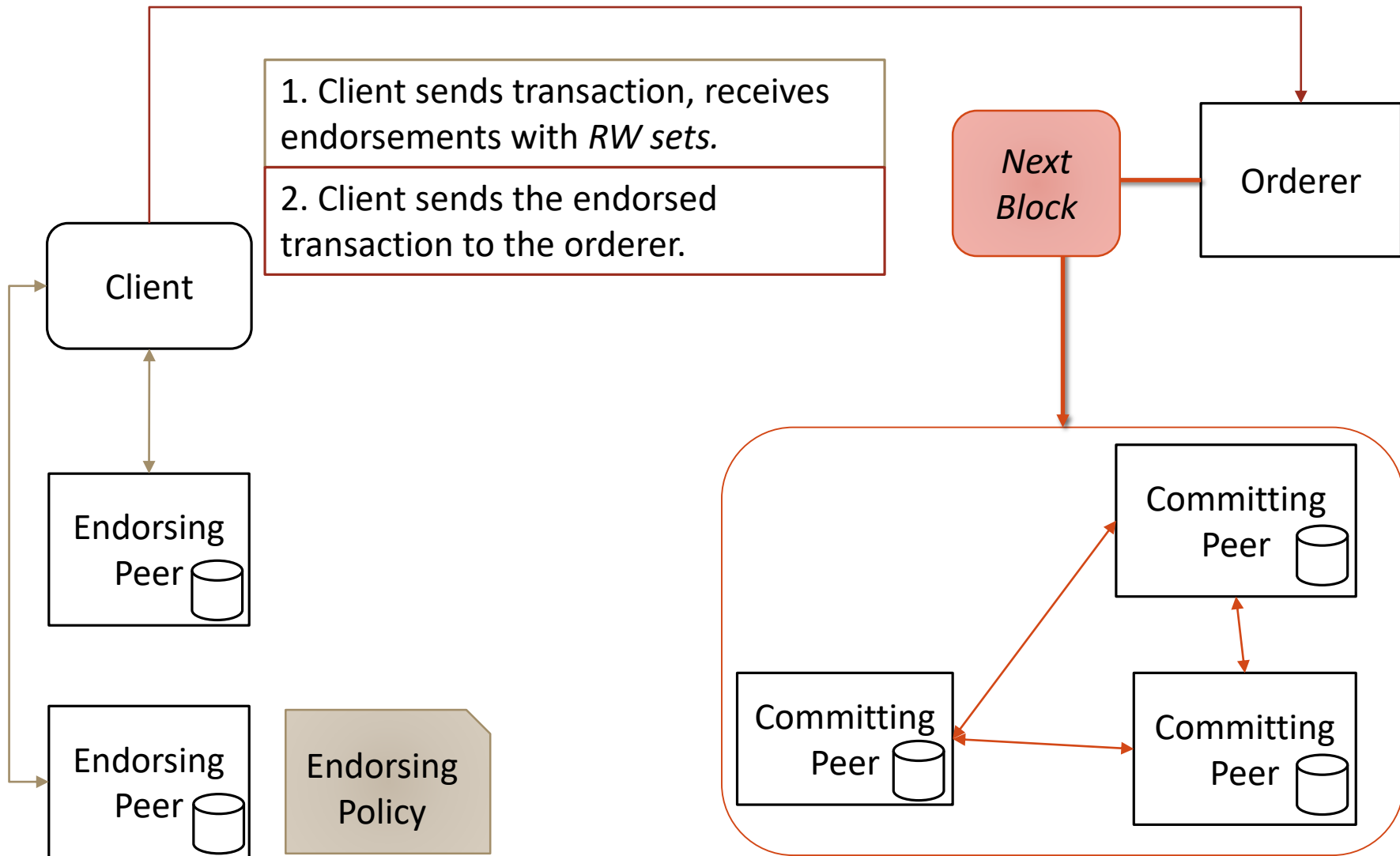
Projects

- Fabric: PBFT Consensus
- Sawtooth: Proof-of-Elapsed-Time (using Intel SGX)
- Composer: Smart contract language and development tool
- Cello: Blockchain-as-a-Service framework
- R3 Corda: Financial applications

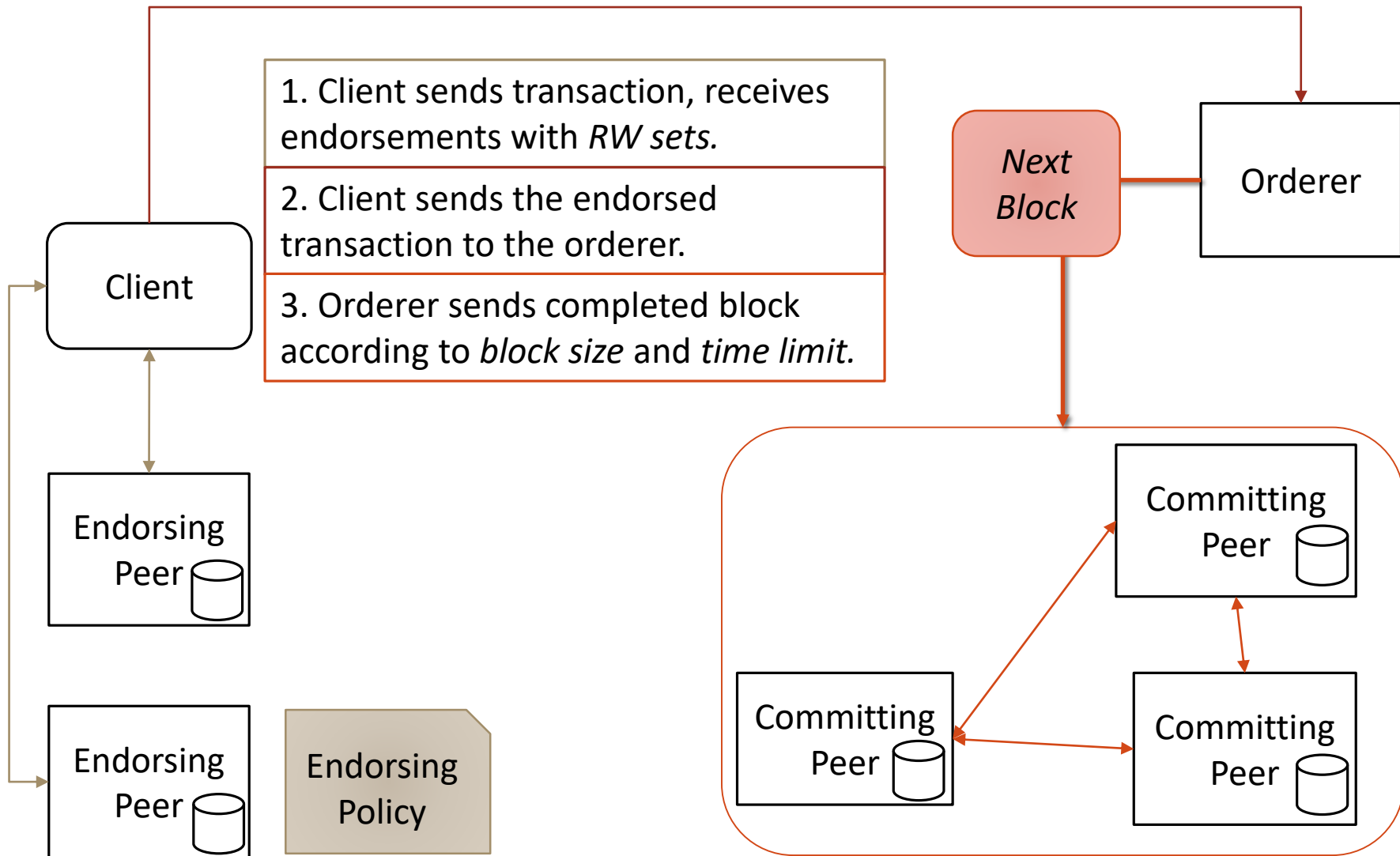
Fabric: Transaction processing flow



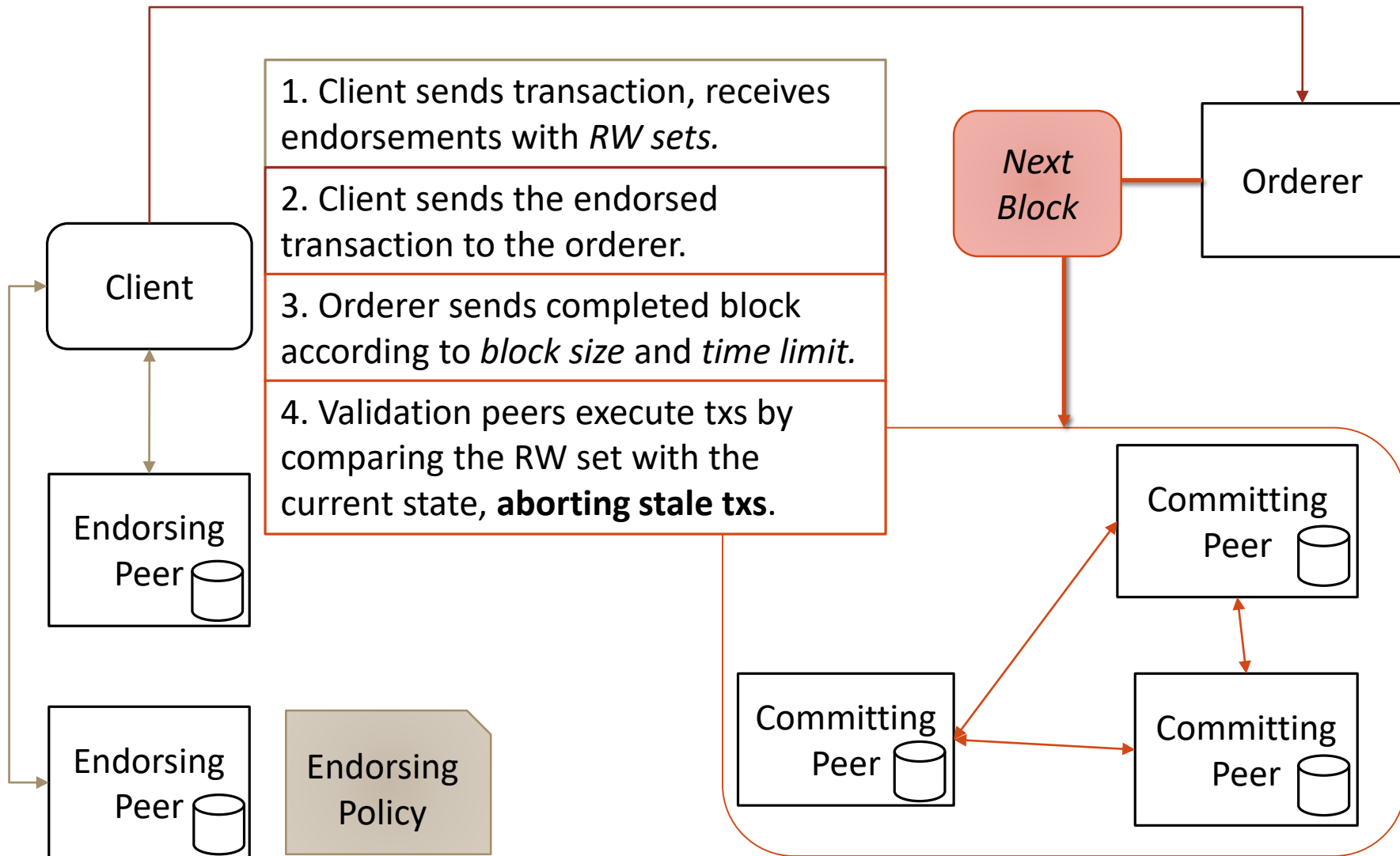
Fabric: Transaction processing flow



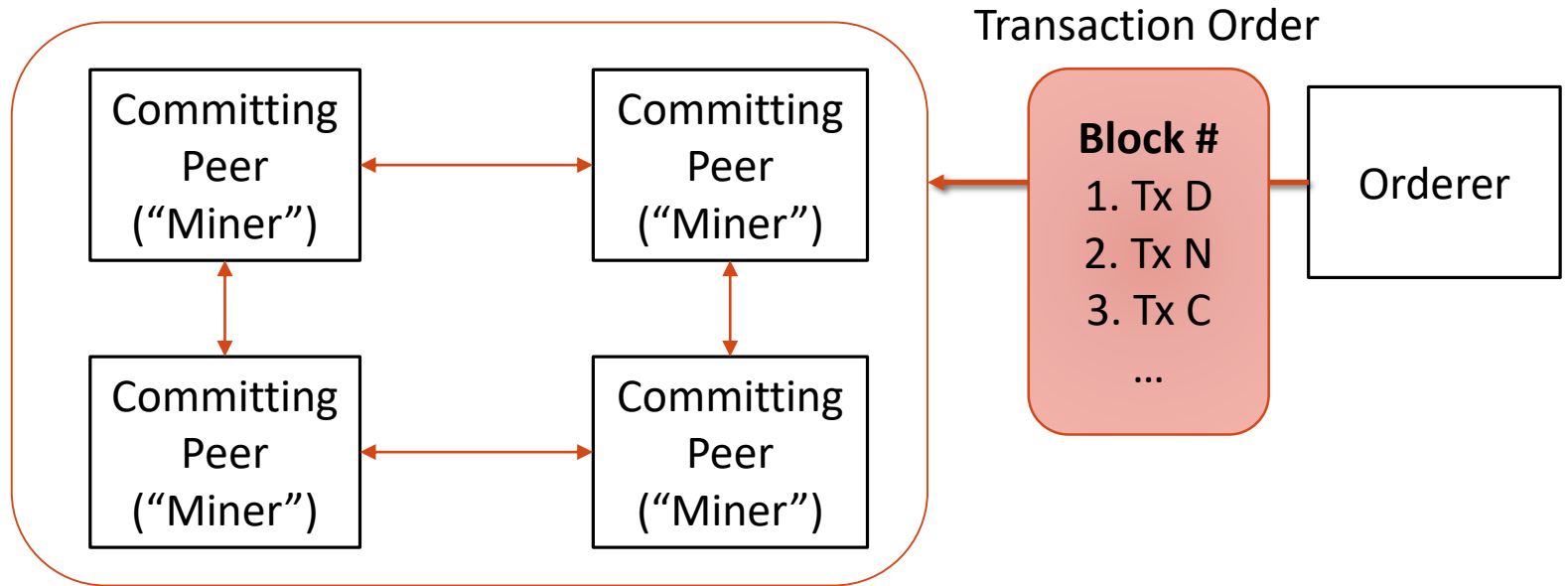
Fabric: Transaction processing flow



Fabric: Transaction processing flow



Fabric: Practical Byzantine Fault Tolerance



- Each peer executes transactions in order.
- The resulting block hash is broadcasted.
- After 2/3 responses, the block is committed locally (v1.0)

Blockchain Insights

BENEFITS AND CHALLENGES
TAXONOMY OF BLOCKCHAINS
RESEARCH OPPORTUNITIES

New challenges introduced by DLTs

Compared to databases

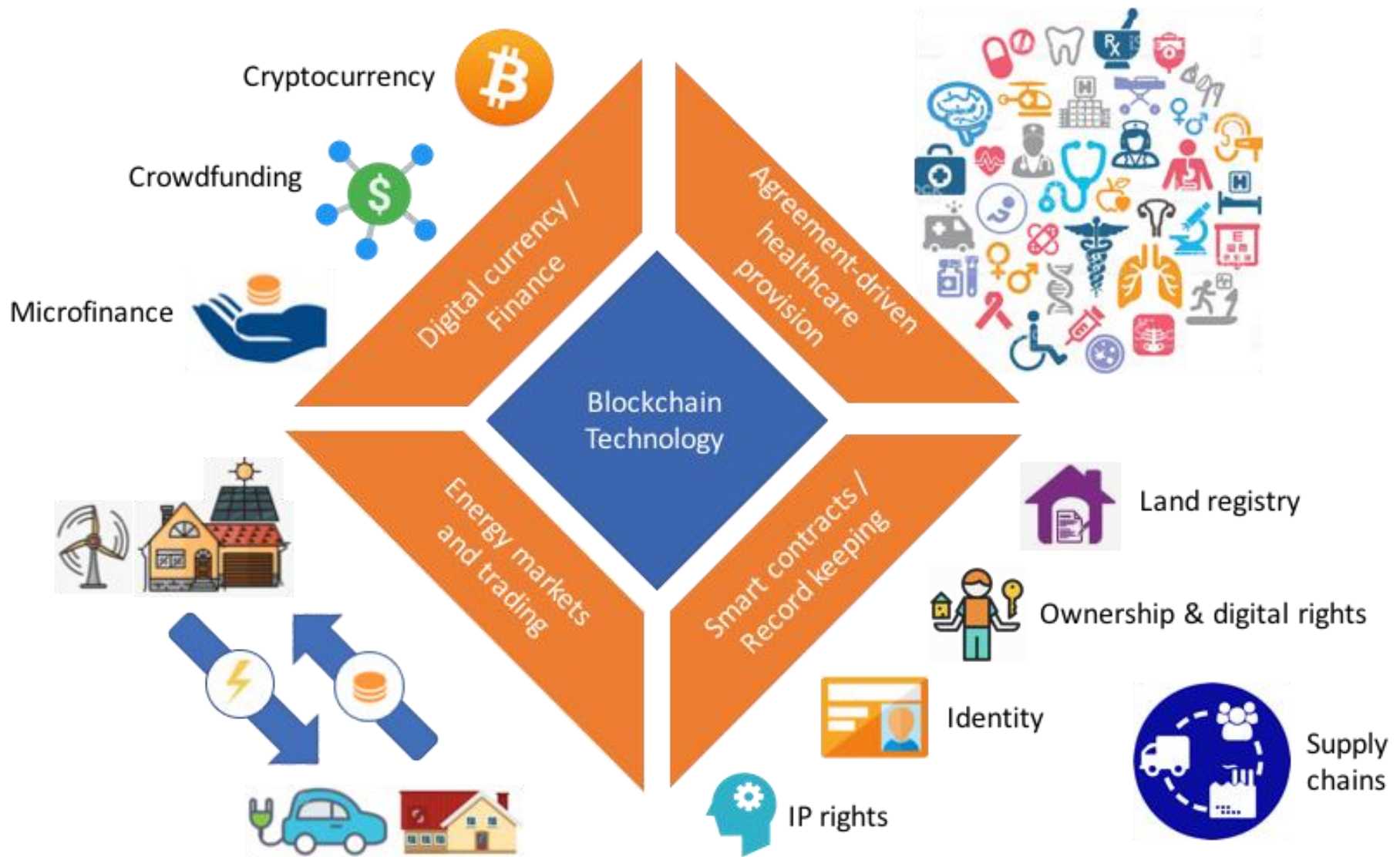
- Slower
- Lower rate of transactions
- Less compact storage

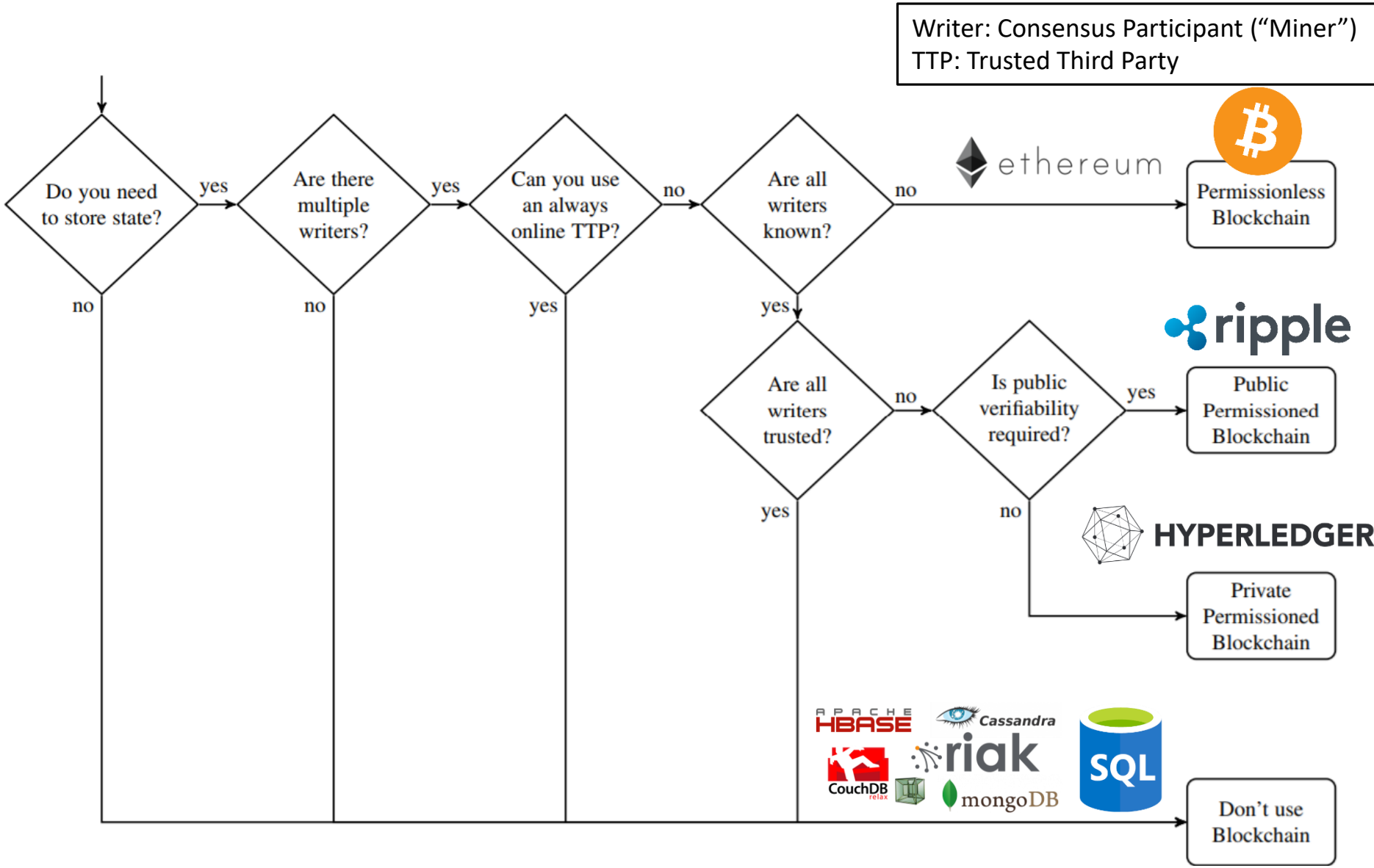
The technology and even standards (and even terminology) are still developing

Additional challenges related to smart contracts

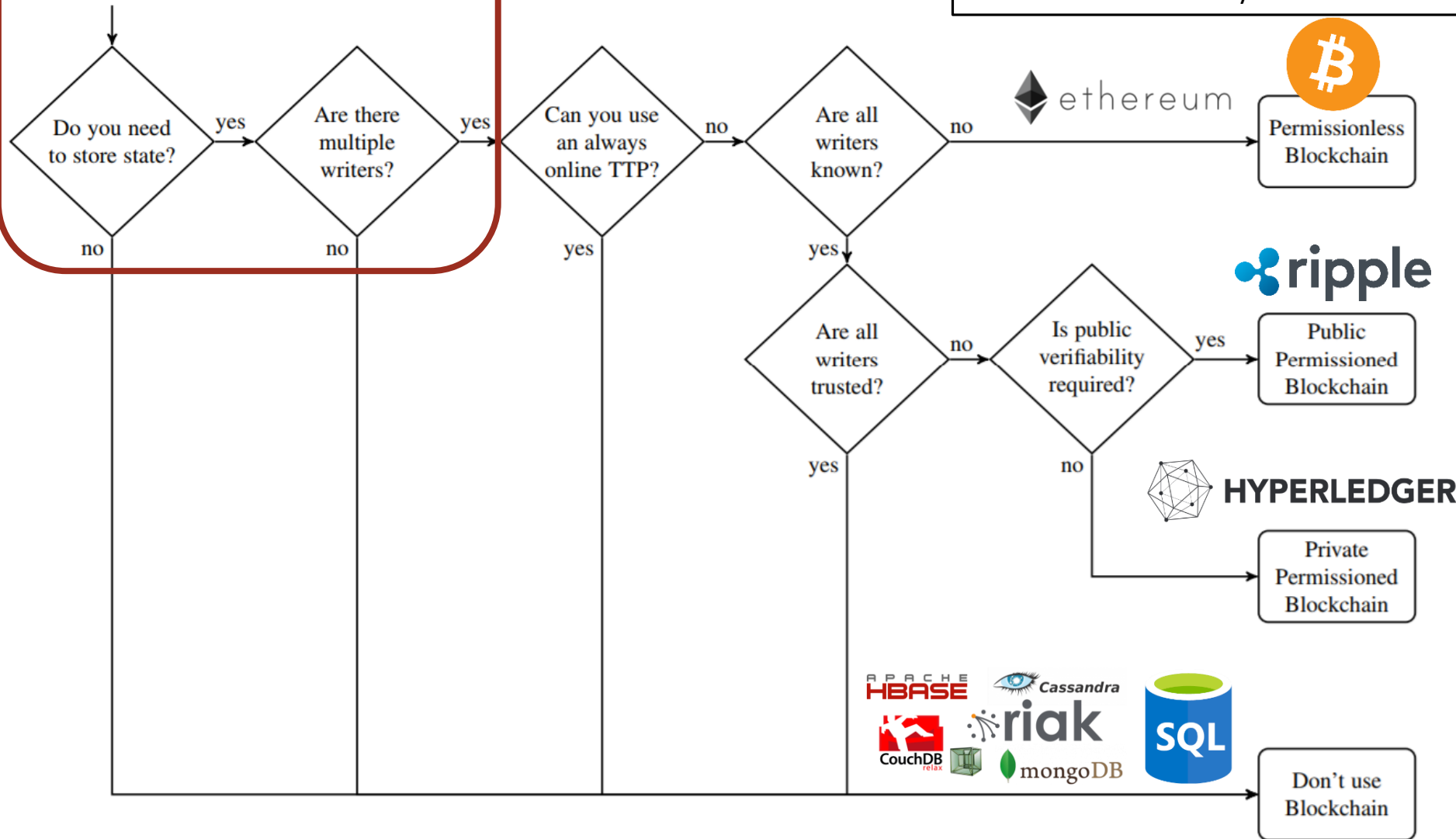
- Bug prone, no established programming or verification practices
- State machine execution, with each contract replica performing every action
- If a contract interacts with an external non-blockchain service, this service needs to be designed with this in mind

Versatility and potential



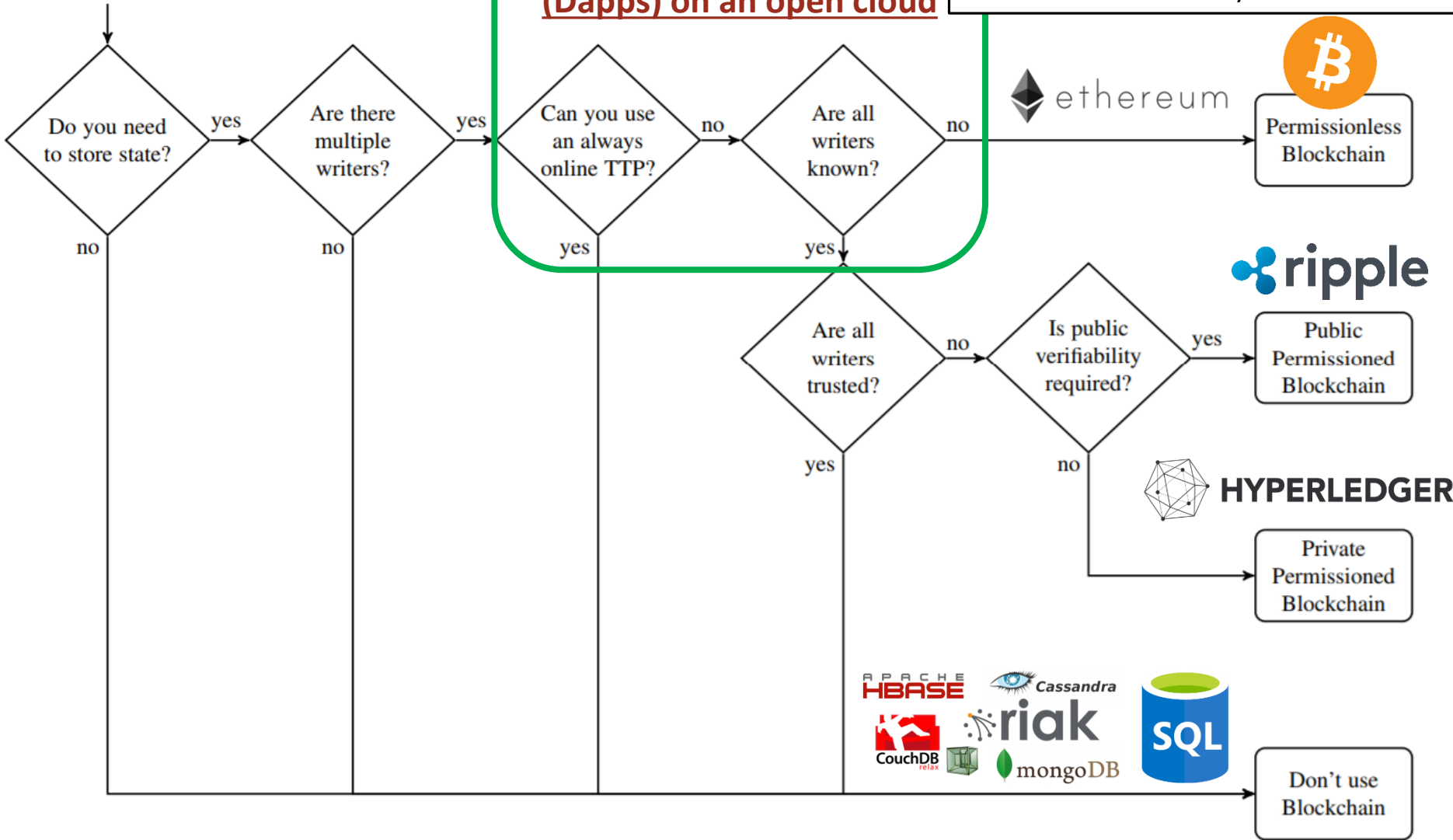


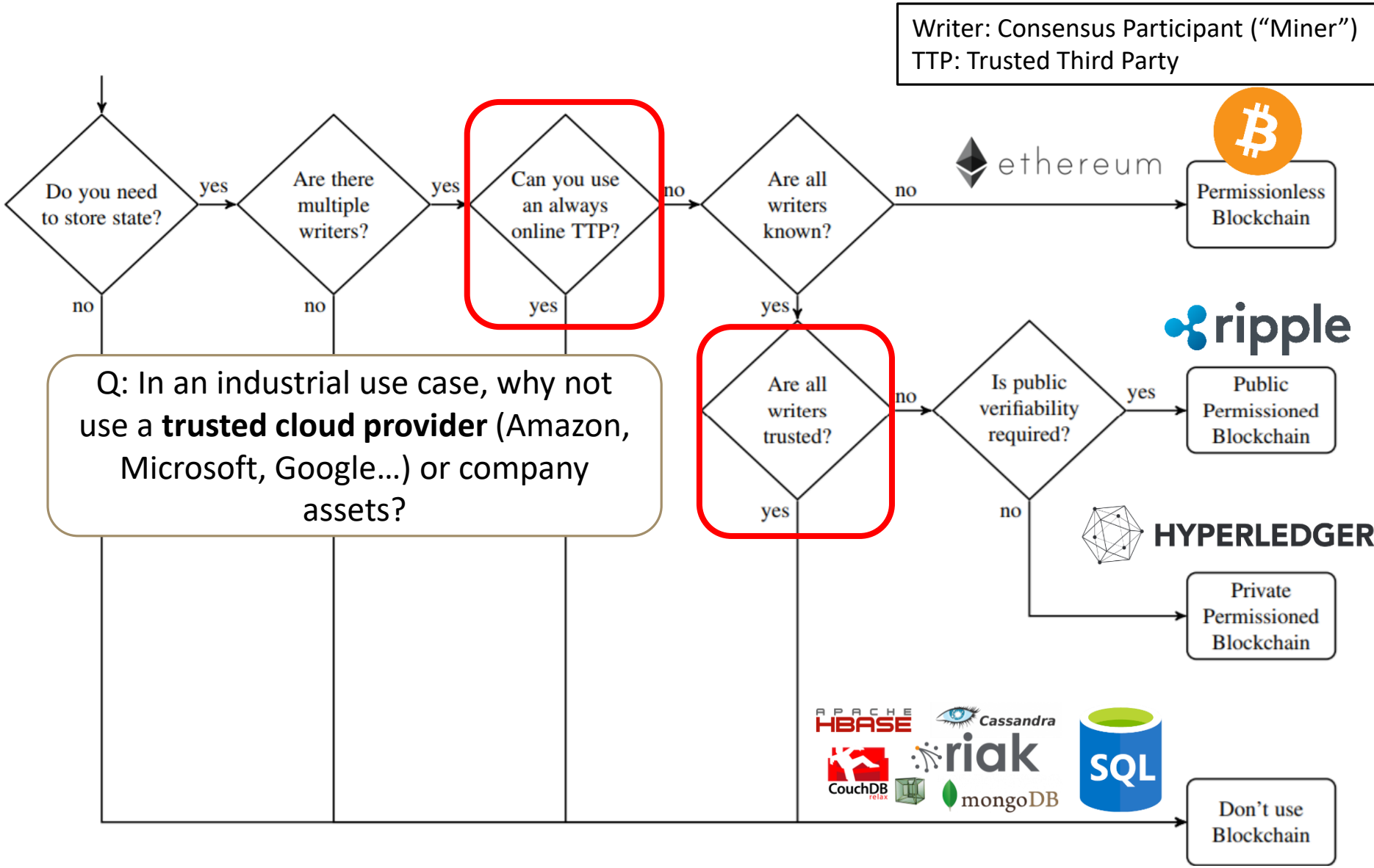
Achieved using a shared database

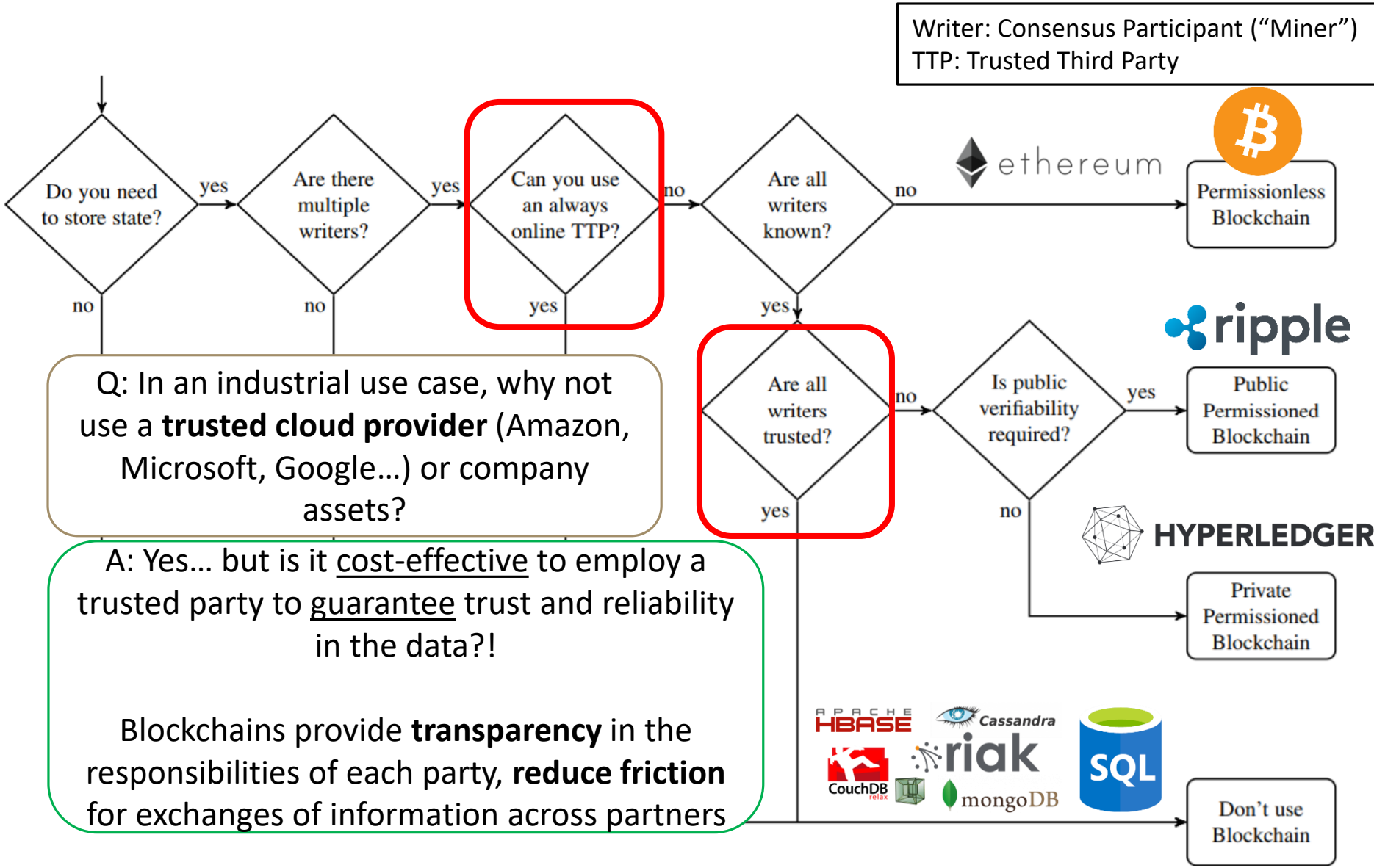


**Censorship-free
Decentralized Applications
(Dapps) on an open cloud**

Writer: Consensus Participant ("Miner")
TTP: Trusted Third Party







Taxonomy

	Anyone can read	Read access restricted
Anyone can propose updates	Bitcoin, Ethereum	Ethereum (Smart Contracts)
Update access restricted	Ripple	Hyperledger, Corda

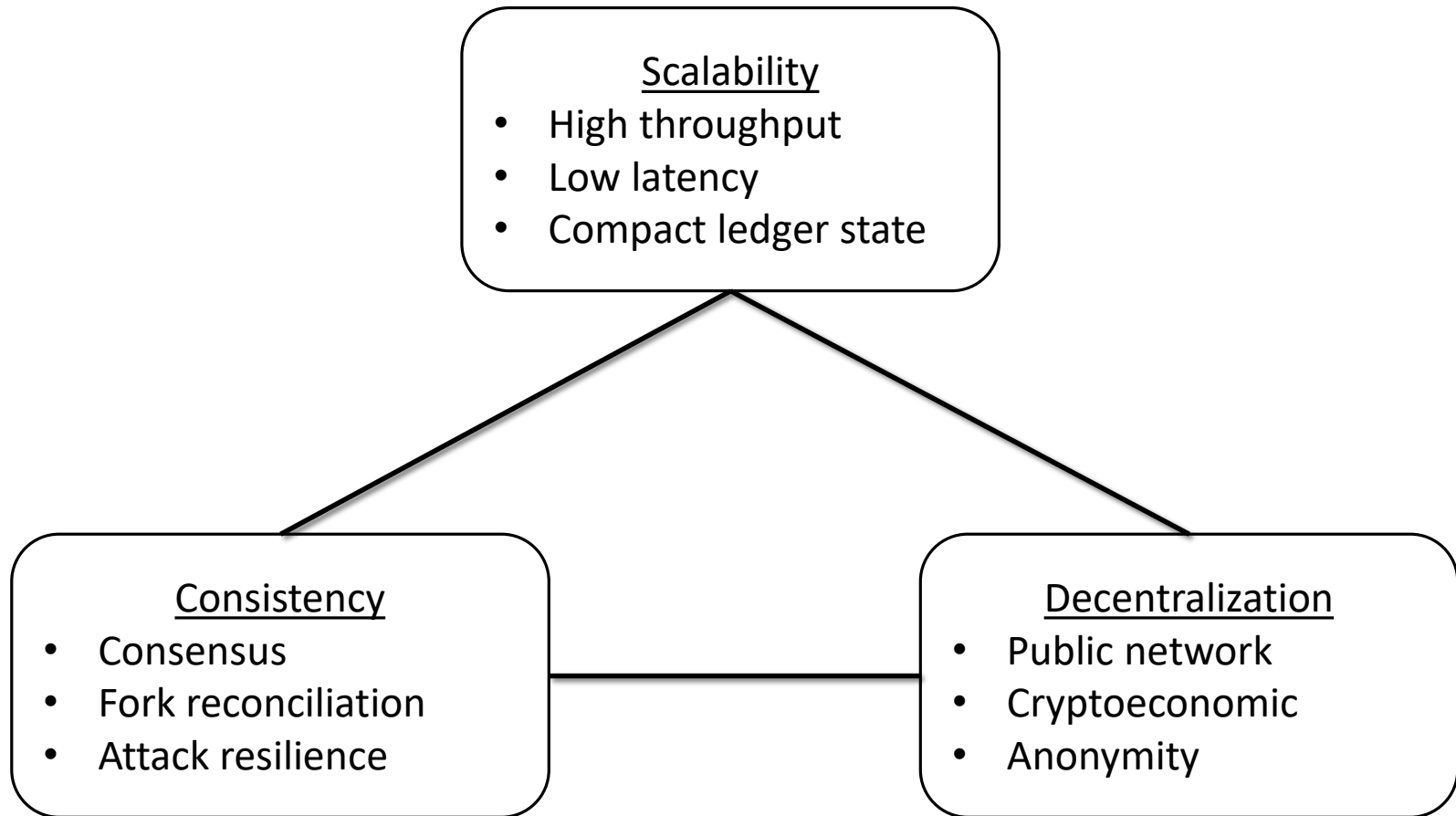
A related feature is if authentication is required

The above is well defined, but has no common terminology associated with it

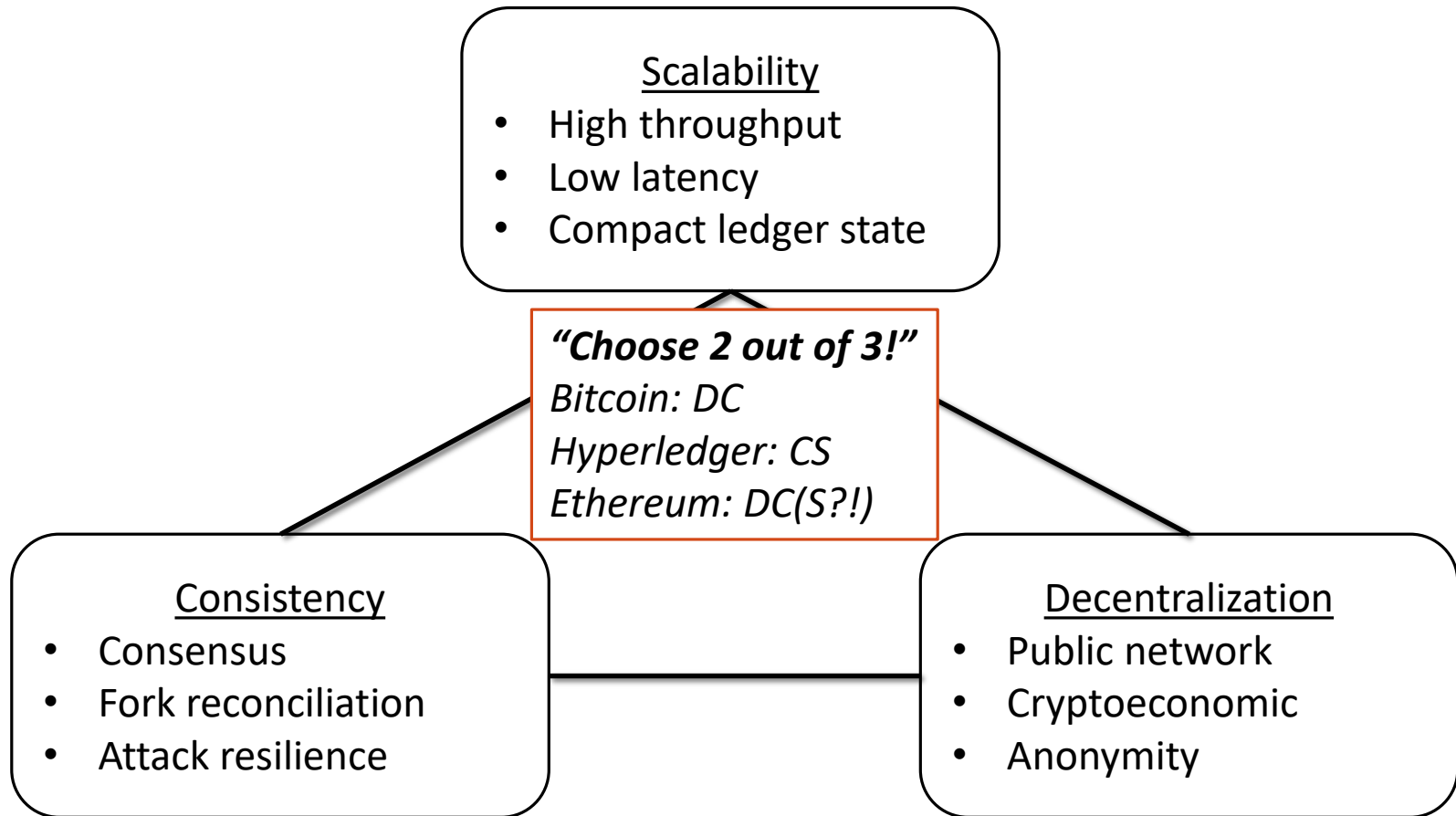
Journalists use other terms instead: open/closed, permissioned/permissionless, public/private

Decentralization: centralized, large-scale decentralized, and consortium blockchains

“CAP Theorem” for DLTs

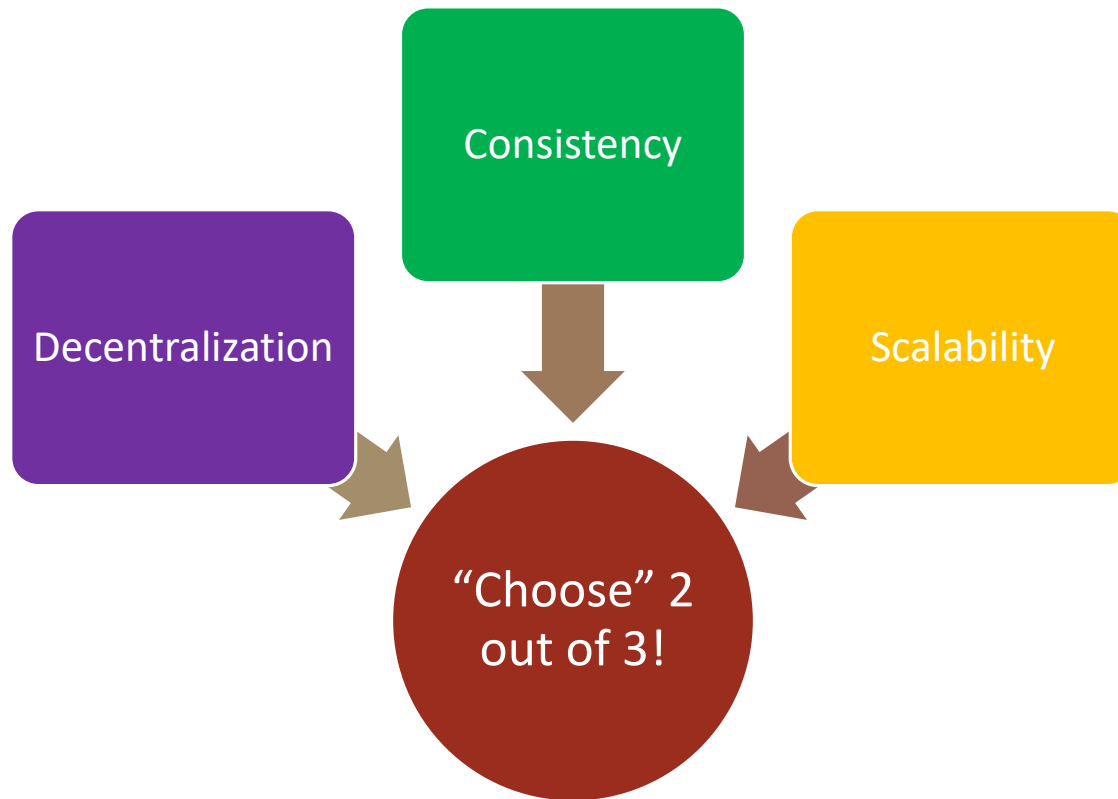


“CAP Theorem” for DLTs



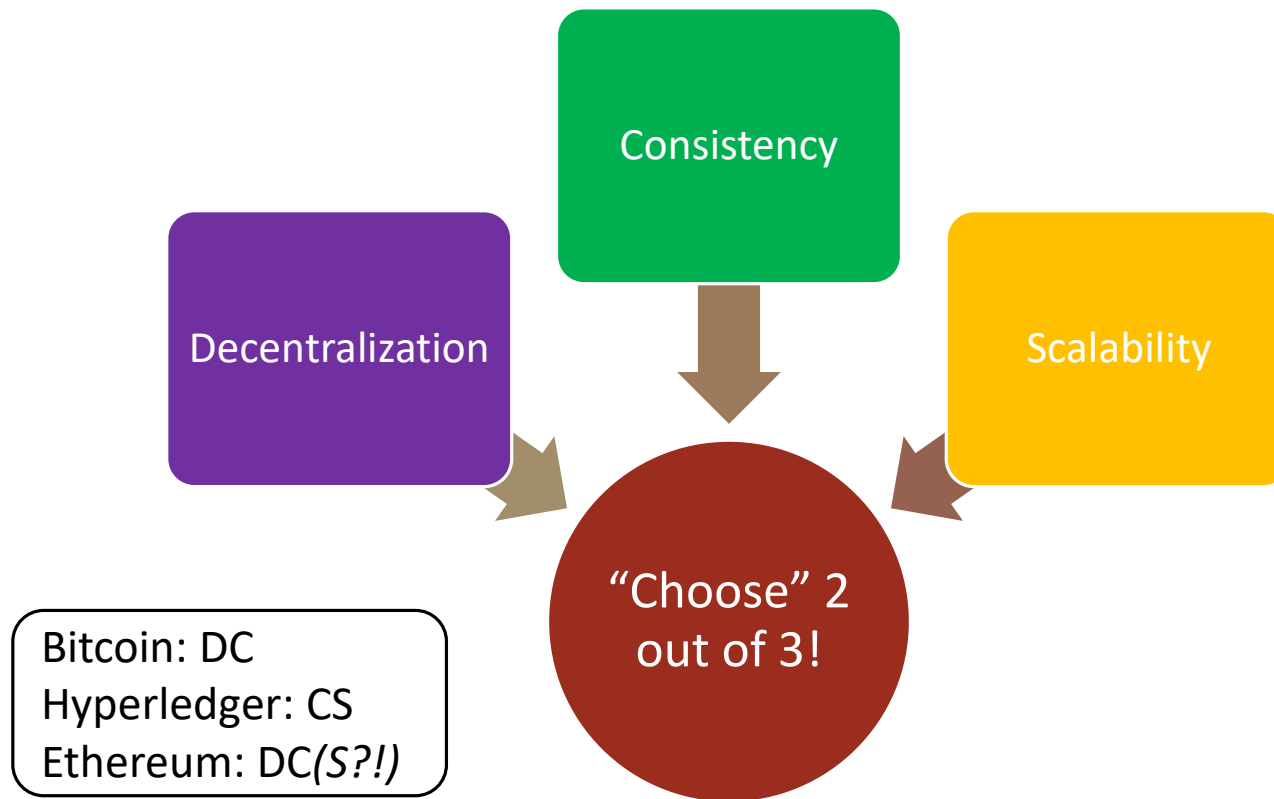


The “CAP” Theorem of Blockchains (DCS)



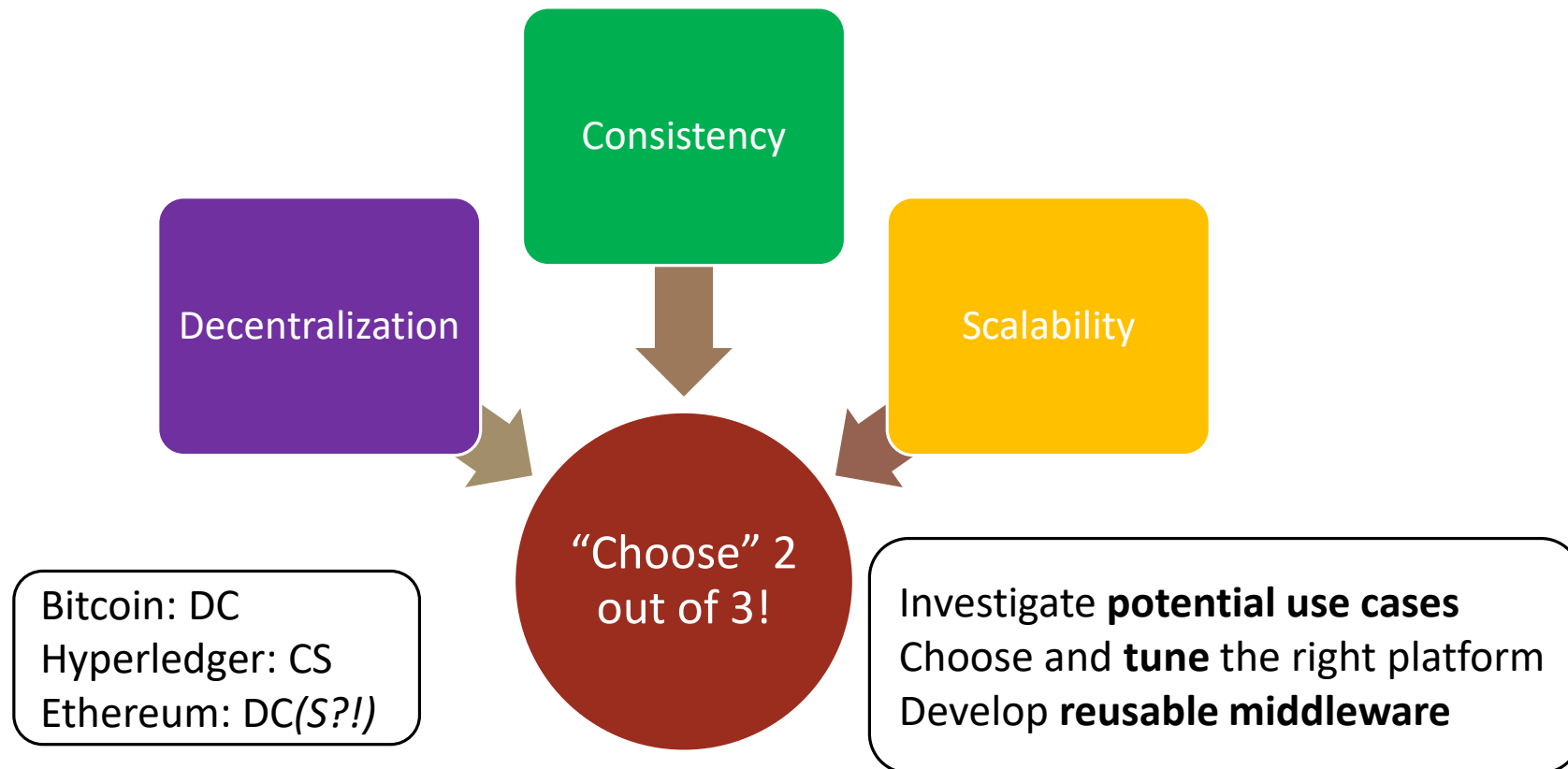


The “CAP” Theorem of Blockchains (DCS)



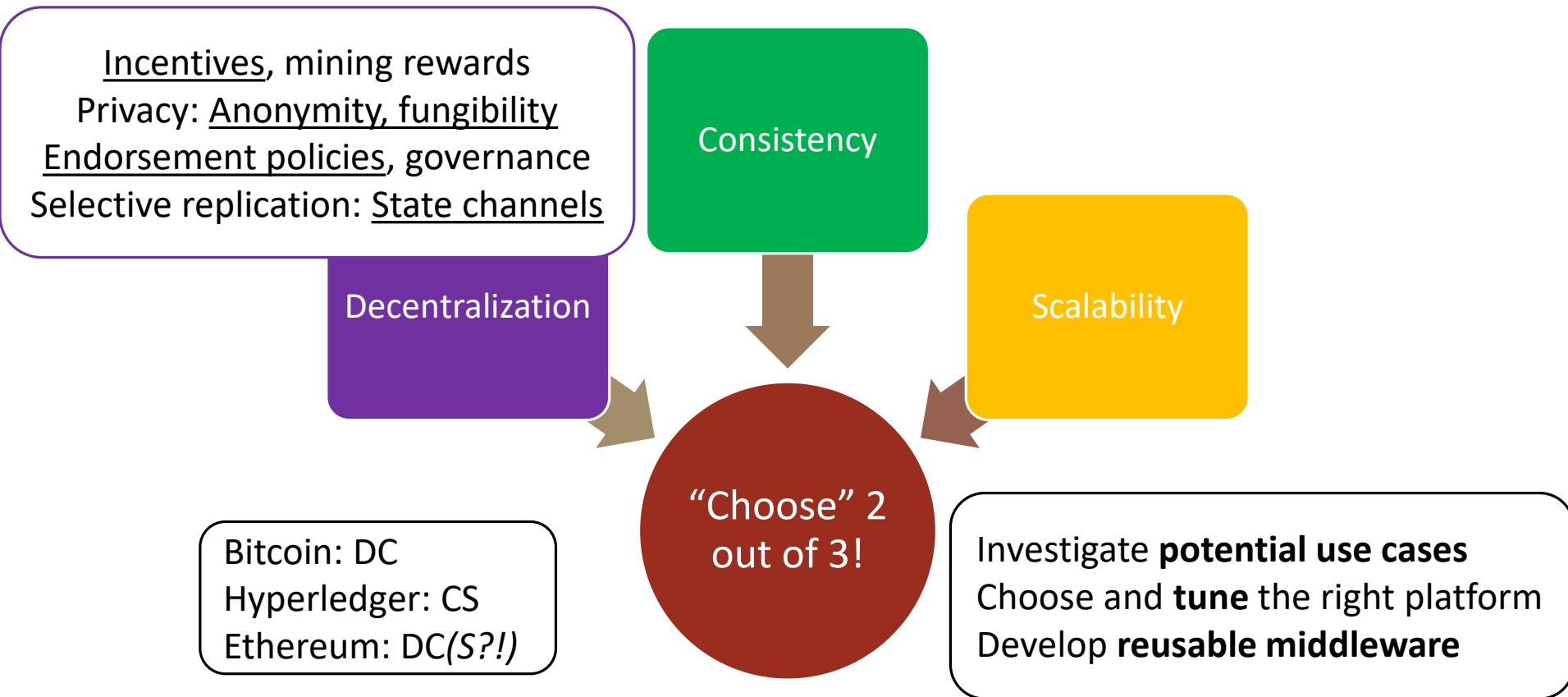


The “CAP” Theorem of Blockchains (DCS)





The “CAP” Theorem of Blockchains (DCS)





The “CAP” Theorem of Blockchains (DCS)

Safe and verifiable smart contracts
Attacker models: <51% attacks
Security of off-chain services (e.g. exchanges)
“Garbage in, garbage out”: IoT barrier

Incentives, mining rewards
Privacy: Anonymity, fungibility
Endorsement policies, governance
Selective replication: State channels

Decentralization

Consistency

Scalability

“Choose” 2
out of 3!

Bitcoin: DC
Hyperledger: CS
Ethereum: DC(S?!)

Investigate **potential use cases**
Choose and **tune** the right platform
Develop **reusable middleware**



The “CAP” Theorem of Blockchains (DCS)

Safe and verifiable smart contracts
Attacker models: <51% attacks
Security of off-chain services (e.g. exchanges)
“Garbage in, garbage out”: IoT barrier

Incentives, mining rewards
Privacy: Anonymity, fungibility
Endorsement policies, governance
Selective replication: State channels

Decentralization

Consistency

Sharding, sidechains, tree-chains, ...
Large-scale chainstate storage
Big Data analytics
Layer 2 Network: Lightning, Raiden
Proof-of-Stake, POET, PBFT, ...

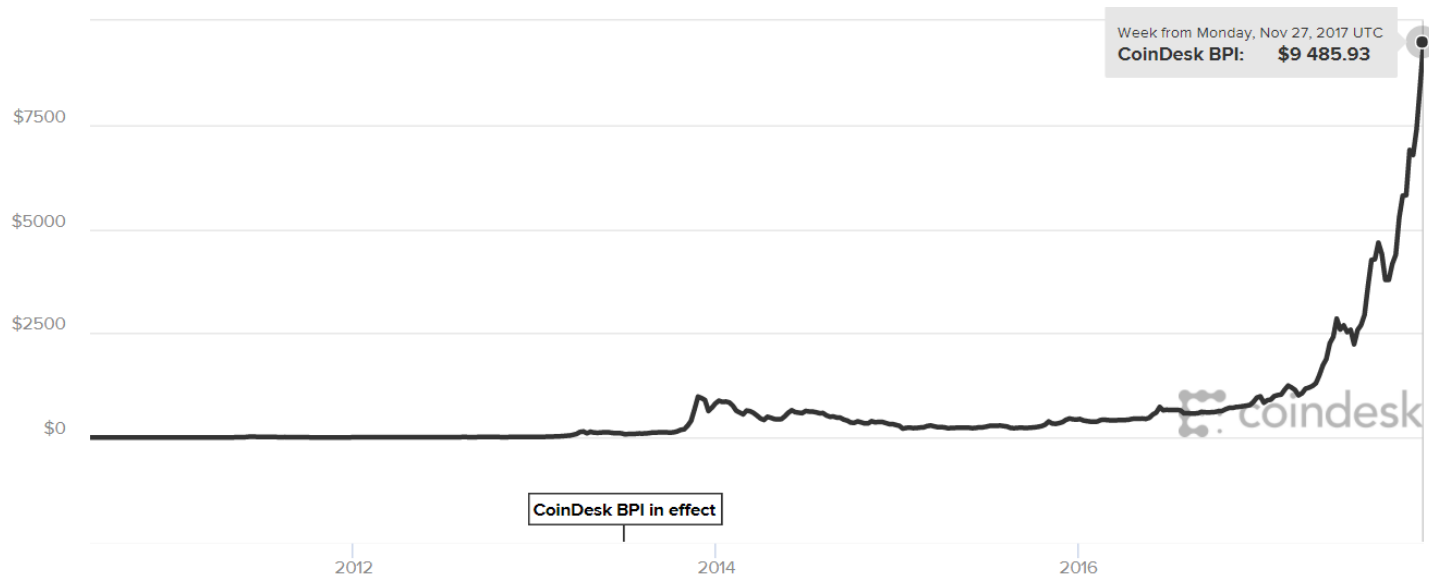
Scalability

“Choose” 2
out of 3!

Bitcoin: DC
Hyperledger: CS
Ethereum: DC(S?!)

Investigate **potential use cases**
Choose and **tune** the right platform
Develop **reusable middleware**

Blockchain 1.0: Currency



\$9,485.93 ▲ 1.71%

Today's Open	\$9,326.59	Change	▲ \$159.34
Today's High	\$9,732.76	Market Cap	\$0.158T
Today's Low	\$9,326.59	Supply	16,704,138

Bitcoin cryptocurrency (2008)

Research for 1.0 Apps

Formally analyze the *security* model of Bitcoin

- 51% attack
- DoS attacks on: mining pools, currency exchanges, ...

Conduct *performance modelling*

- Simulate various Bitcoin scenarios
- Understand impact of network topologies (e.g. partitions)

Develop *scalable* mechanisms with *legacy support* to maintain the *sustainability* of Bitcoin

- SegWit2x
- Bitcoin-NG (NSDI '16)
- Off-chain (Lightning network)
- Algorand (SOSP '17)

Blockchain 2.0: Decentralized Apps

ƉApps are applications built on blockchain platforms using smart contracts (e.g. Ethereum)

Ɖ apps



ETHEREUM

Blockchain 2.0: Decentralized Apps

ƉApps are applications built on blockchain platforms using smart contracts (e.g. Ethereum)



ETHEREUM

The image shows the 'EtherTweet' logo and tagline on a dark green rectangular background. The word 'EtherTweet' is written in a large, white, bold, sans-serif font. Below it, the tagline 'Decentralized Microblogging' is written in a smaller, white, sans-serif font.

Blockchain 2.0: Decentralized Apps

ƉApps are applications built on blockchain platforms using smart contracts (e.g. Ethereum)

Ɖ apps



ETHEREUM

EtherTweet
Decentralized Microblogging



Token Distribution
Crowdfunding

Blockchain 2.0: Decentralized Apps

ÐApps are applications built on blockchain platforms using smart contracts (e.g. Ethereum)



ETHEREUM

EtherTweet
Decentralized Microblogging

Token Distribution
Crowdfunding

alice
Charity donation payment

More 2.0 DApps

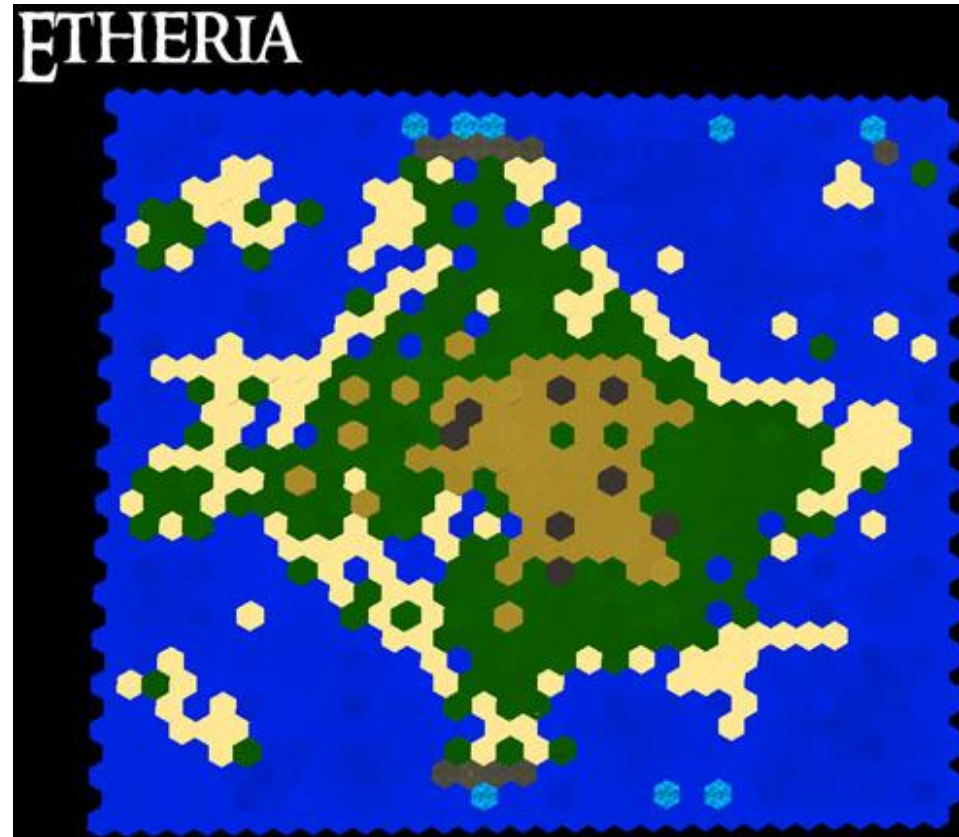
More 2.0 DApps



GNOSIS

Forecast market (e.g. betting, insurance)

More 2.0 DApps



Decentralized virtual world

Research for 2.0 Apps

Formal *verify* smart contracts, detect and repair security flaws

- Ethereum Viper

Develop *scalable consensus* mechanisms which support *smart contracts* in an *public* network (w/ *incentives*)

- Proof-of-Stake (Casper)
- Side-chain (Plasma)
- Sharding (ShardSpace)

Develop *efficient data storage* techniques to store *smart contracts* and the *chainstate*

- AVL+ (Tendermint)
- Merkle Patricia Trees (Ethereum)
- Zero-Knowledge Proofs: zk-SNARK

Blockchain 3.0: Pervasive Apps

Applications
involve entire
industries,
public sector,
and IoT.

Blockchain 3.0: Pervasive Apps



everledger

Diamonds Provenance

Applications
involve entire
industries,
public sector,
and IoT.

Blockchain 3.0: Pervasive Apps



everledger

Diamonds Provenance

Applications
involve entire
industries,
public sector,
and IoT.



FACTOM

Land Registry in Honduras

Blockchain 3.0: Pervasive Apps



everledger

Diamonds Provenance

Applications involve entire industries, **public sector**, and IoT.



FACTOM

Land Registry in Honduras



BlockchainHealth

Electronic Health Records

Blockchain 3.0: Pervasive Apps



everledger

Diamonds Provenance

Applications involve entire industries, **public sector**, and IoT.



FACTOM

Land Registry in Honduras



BlockchainHealth

Electronic Health Records



VOTEWATCHER

Transparent Voting System

Research for 3.0 Apps

Develop “*clean-slate*” scalable distributed ledgers:

- Permissioned ledgers (Hyperledger Fabric)
- Blockless DLTs (IOTA Tangles, R3 Corda Notaries, Hashgraph)

Develop *blockchain modelling tools and middleware*

- BPMN, Business Artifacts with Lifecycles, FSM
- Authentication, reputation, auction, voting, etc.

Support strict *governance, security, and privacy* requirements

- State channels
- Endorsement policies

Overcome the *cyber-physical barrier for data entry*:

- Object fingerprinting
- Secure hardware sensors

Storage system for blockchain

More demanding storage requirements

- Data updates are up to 103-105 times bigger than Bitcoin transactions
- Data updates are heterogeneous in size and other parameters
- More frequent updates and demanding faster response times

Partition the data, use many small blockchains instead of a single big one

Store only newer entries on most servers while only a small number of servers keep all the entries

Store most data off-chain with the hash being on-chain

Partition the blockchain and store different data entries on different servers

- Within the same enterprise or belonging to different organisations

Partition the off-chain data and store on different servers

- Same options as above

Calls for event-based interaction!

Due to multiple applications using the same blockchain

- Mostly disjoint but there might be some data overlap

Due to multiple co-existing blockchains

- There is a need to maintain consistency across blockchains

Due to partitioning blockchains

- Between on-chain and off-chain storage mechanisms
- When re-partitioning a blockchain, e.g., due to a policy change
- When applying partial replication