Blockchain, cryptography, and consensus

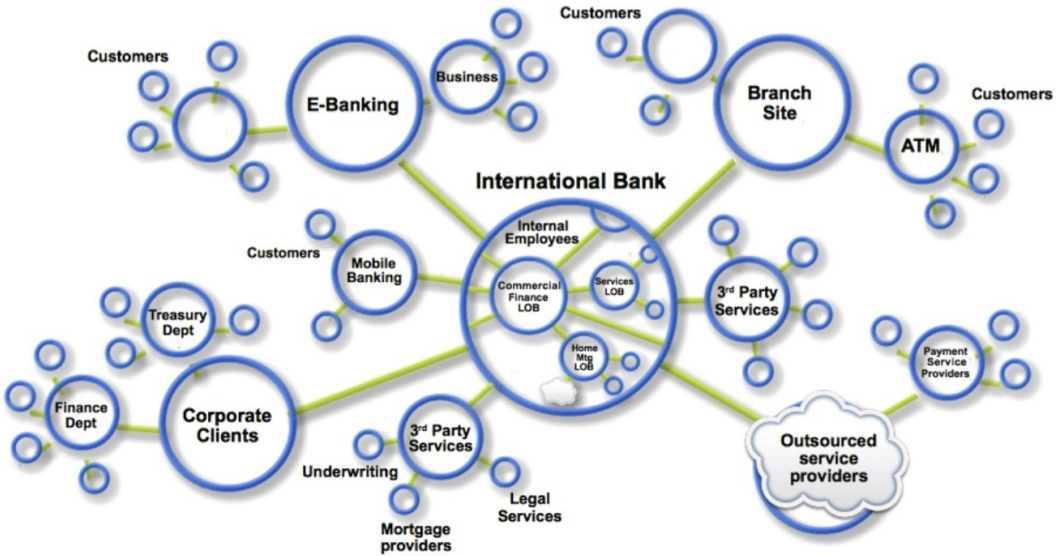
Christian Cachin
(with Elli Androulaki, Angelo De Caro, Andreas Kind, Mike Osborne, Simon Schubert, Alessandro Sorniotti, Marko Vukolic and many more)

IBM Research – Zurich

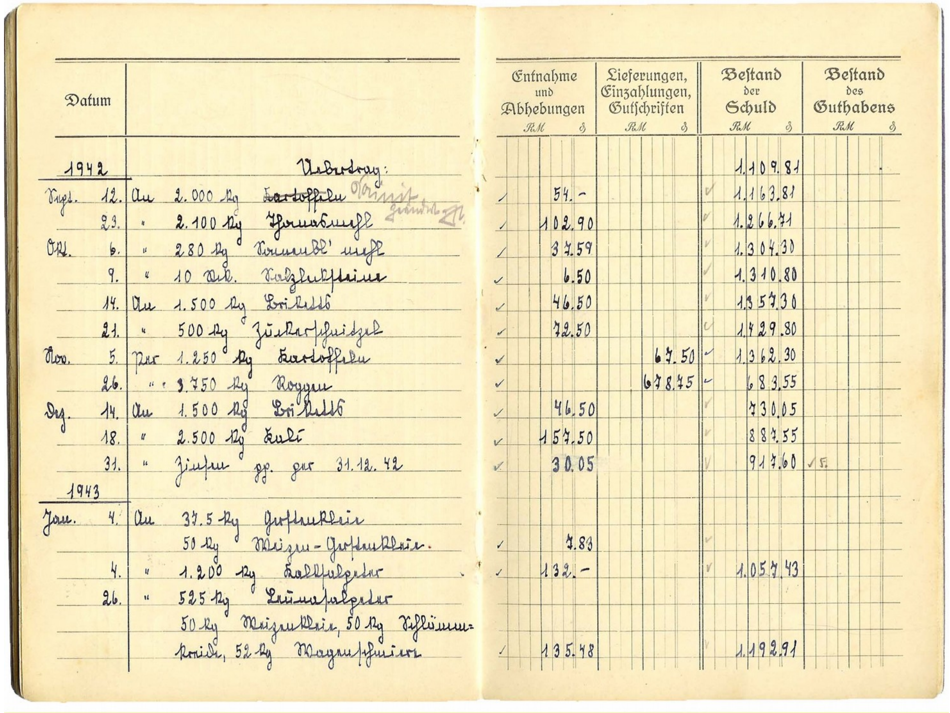
June 2017

Connected markets

- Networks connect participants
  - Customers, suppliers, banks, consumers
- Markets organize trades
  - Public and private markets
- Value comes from assets
  - Physical assets (house, car ...)
  - Virtual assets (bond, patent ...)
  - Services are also assets
- Transactions exchange assets

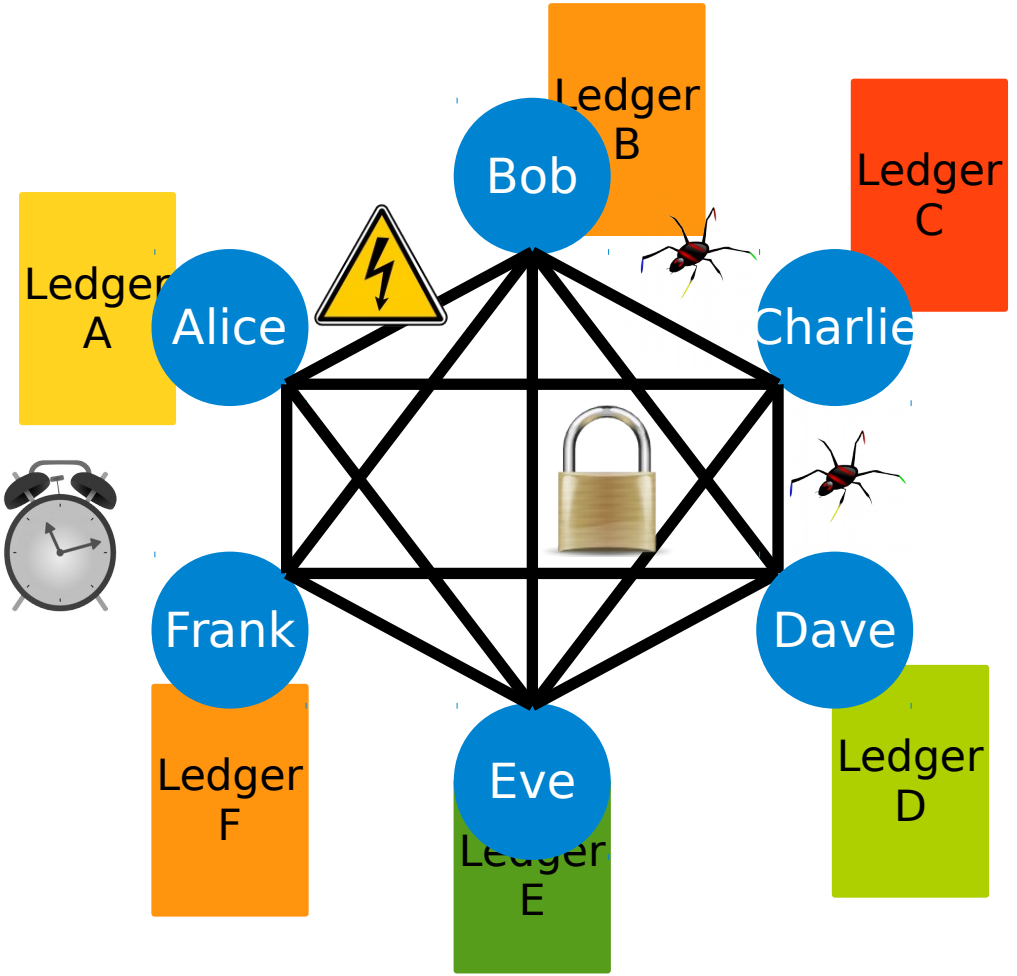


Ledger



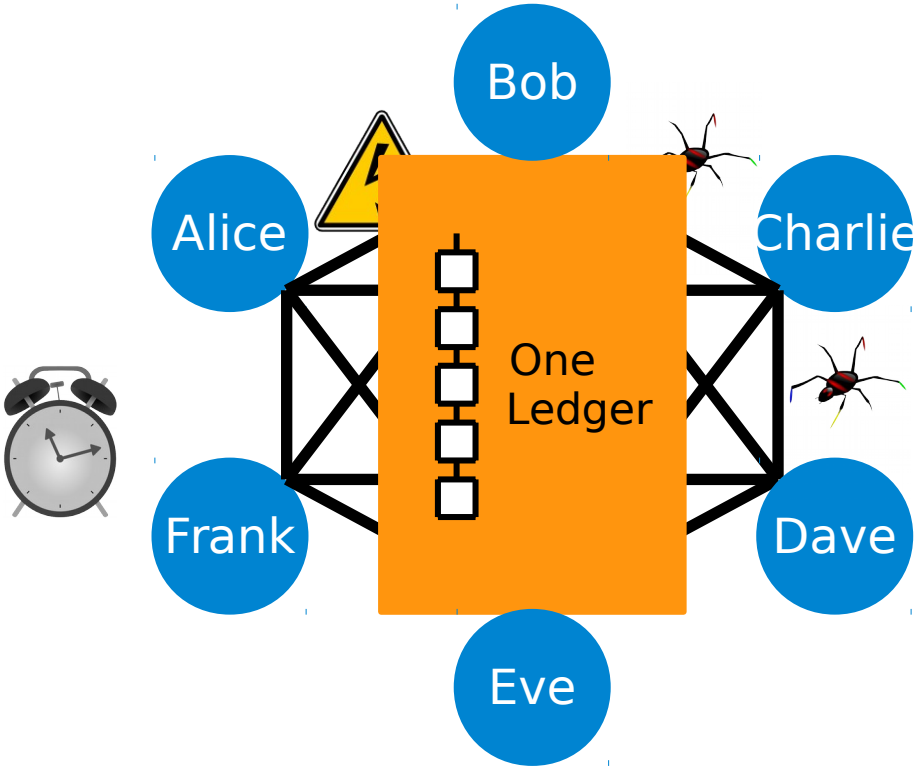
- Ledger records all business activity as transactions
  - Databases
- Every market and network defines a ledger
- Ledger records asset transfers between participants
- Problem — (Too) many ledgers
  - Every market has its ledger
  - Every organization has its own ledger

Multiple ledgers



- Every party keeps its own ledger and state
- Problems, incidents, faults
- Diverging ledgers

Blockchain provides one virtual ledger



- One common trusted ledger
- Today often implemented by a centralized intermediary
- Blockchain creates one single ledger for all parties
- Replicated and produced collaboratively
- Trust in ledger from
  - Cryptographic protection
  - Distributed validation

Four elements characterize Blockchain

Replicated ledger

- History of all transactions
- Append-only with immutable past
- Distributed and replicated

Cryptography

- Integrity of ledger
- Authenticity of transactions
- Privacy of transactions
- Identity of participants

Consensus

- Decentralized protocol
- Shared control tolerating disruption
- Transactions validated

Business logic

- Logic embedded in the ledger
- Executed together with transactions
- From simple "coins" to self-enforcing "smart contracts"

Blockchain simplifies complex transactions

Logistics

Real-time visibility
Improved efficiency
Transparency & verifiability
Reduced cost

Property records

Digital but unforgeable
Fewer disputes
Transparency & verifiability
Lower transfer fees

Capital markets

Faster settlement times
Increased credit availability
Transparency & verifiability
No reconciliation cost

Blockchain scenario features

- A given task or problem, but no (central) trusted party available
- Protocol among multiple nodes, solving a distributed task
  - The writing nodes decide and reach consensus collectively
- Key aspects of the distributed task
  - Stores data
  - Multiple nodes write
  - Not all writing nodes are trusted
  - Operations are (somewhat) verifiable
- If all writing nodes are known → permissioned or consortium blockchain
- Otherwise, when writing nodes are not known → permissionless or public blockchain

Why blockchain now?

- Cryptography has been a key technology in the financial world for decades
  - Payment networks, ATM security, smart cards, online banking ...
- Trust model of (financial) business has not changed
  - Trusted intermediary needed for exchange among non-trusting partners
  - Today cryptography mostly secures point-to-point interactions
- Bitcoin started in 2009
  - Embodies only cryptography of 1990s and earlier
  - First prominent use of cryptography for a new trust model (= trust no entity)
- The promise of Blockchain – Reduce trust and replace it by technology
  - Exploit advanced cryptographic techniques

What is a blockchain?

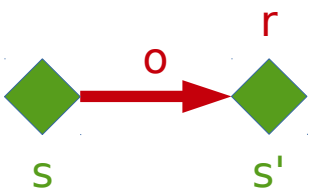
When you get too many people talking about the same thing it tends to clutter up things.

Bob Dylan, 1965

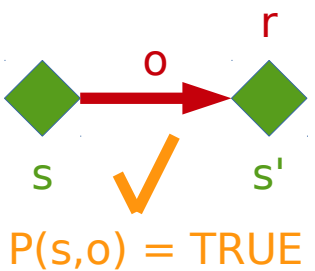
A state machine

- Functionality F
  - Operation o transforms a state s to new state s' and may generate a response r

$(s', r) \leftarrow F(s, o)$

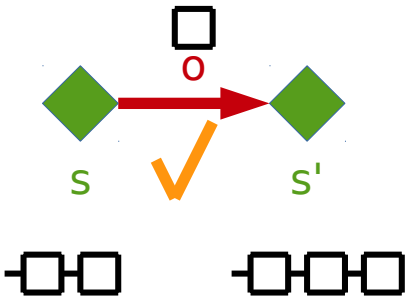


- Validation condition
  - Operation needs to be valid, in current state, according to a predicate P()



Blockchain state machine

- Append-only log
  - Every operation o appends a "block" of valid transactions (tx) to the log



- Log content is verifiable from the most recent element

- Log entries form a hash chain

  $h_t \leftarrow \text{Hash}([tx_1, tx_2, \ldots] \,||\, h_{t-1} \,||\, t)$.

Example – The Bitcoin state machine

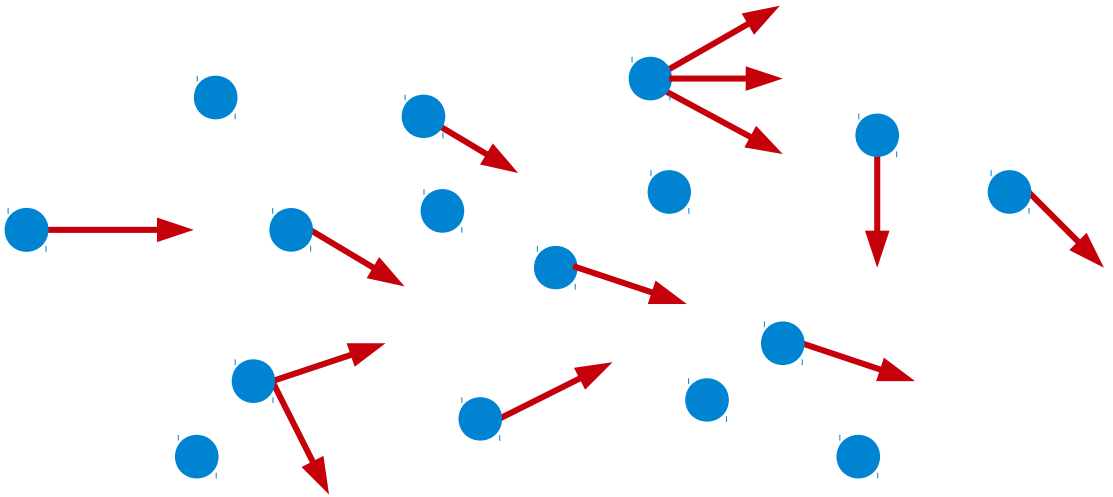
- Bitcoins are unforgeable bitstrings
  - "Mined" by the protocol itself (see later)

- Digital signature keys (ECDSA) own and transfer bitcoins
  - Owners are pseudonymous, e.g., 3JDs4hAZeKE7vER2YvmH4yTMDEfoA1trnC

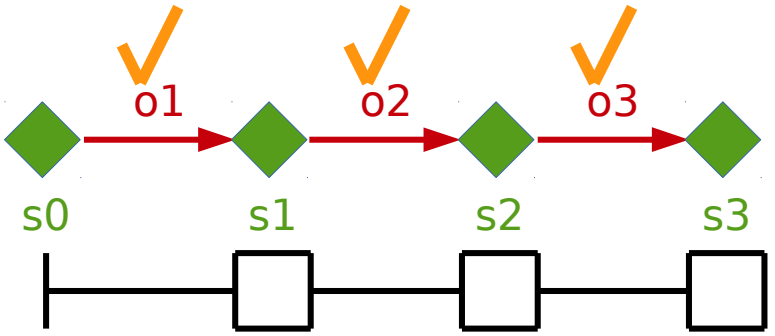
- Every transaction transfers a bitcoin (fraction) from current to next owner
  - "This bitcoin now belongs to 3JDs..." signed by the key of current owner
  - (Flow linkable by protocol, and not anonymous when converted to real-world assets)

- Validation is based on the global history of past transactions
  - Signer has received the bitcoin before
  - Signer has not yet spent the bitcoin

Distributed p2p protocol to create a ledger



Nodes
produce
transactions



Nodes run a protocol to construct the ledger

Blockchain protocol features

- Only "valid" operations (transactions) are "executed"

- Transactions can be simple
  - Bitcoin tx are statement of ownership for coins, digitally signed
    "This bitcoin now belongs to K2" signed by K1

- Transactions can be arbitrary code (smart contracts)
  - Embody logic that responds to events (on blockchain) and may transfer assets in response
  - Auctions, elections, investment decisions, blackmail ...

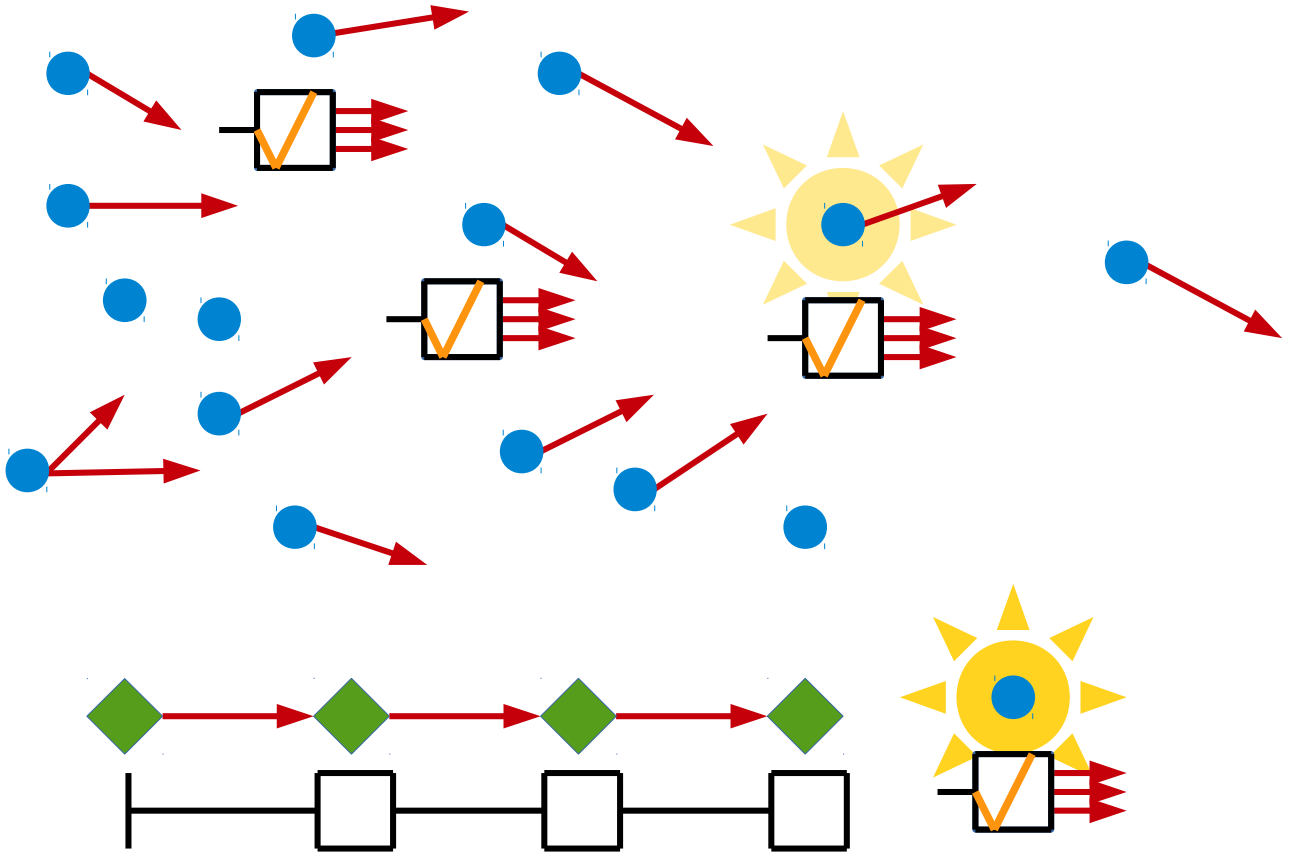
Consensus

Three kinds of consensus for blockchain

- Decentralized / permissionless
  - Bitcoin, Ethereum
- Somewhat decentralized
  - Ripple, Stellar
- Consortium / permissioned
  - BFT (Byzantine fault tolerance) consensus

Decentralized – Nakamoto consensus/Bitcoin

- Nodes prepare blocks
  - List of transactions (tx)
  - All tx valid
- Lottery race
  - Solves a hard puzzle
  - Selects a random winner/leader
  - Winner's operation/ block is executed and "mines" a coin
- All nodes verify and validate new block
  - "Longest" chain wins



Decentralized = permissionless

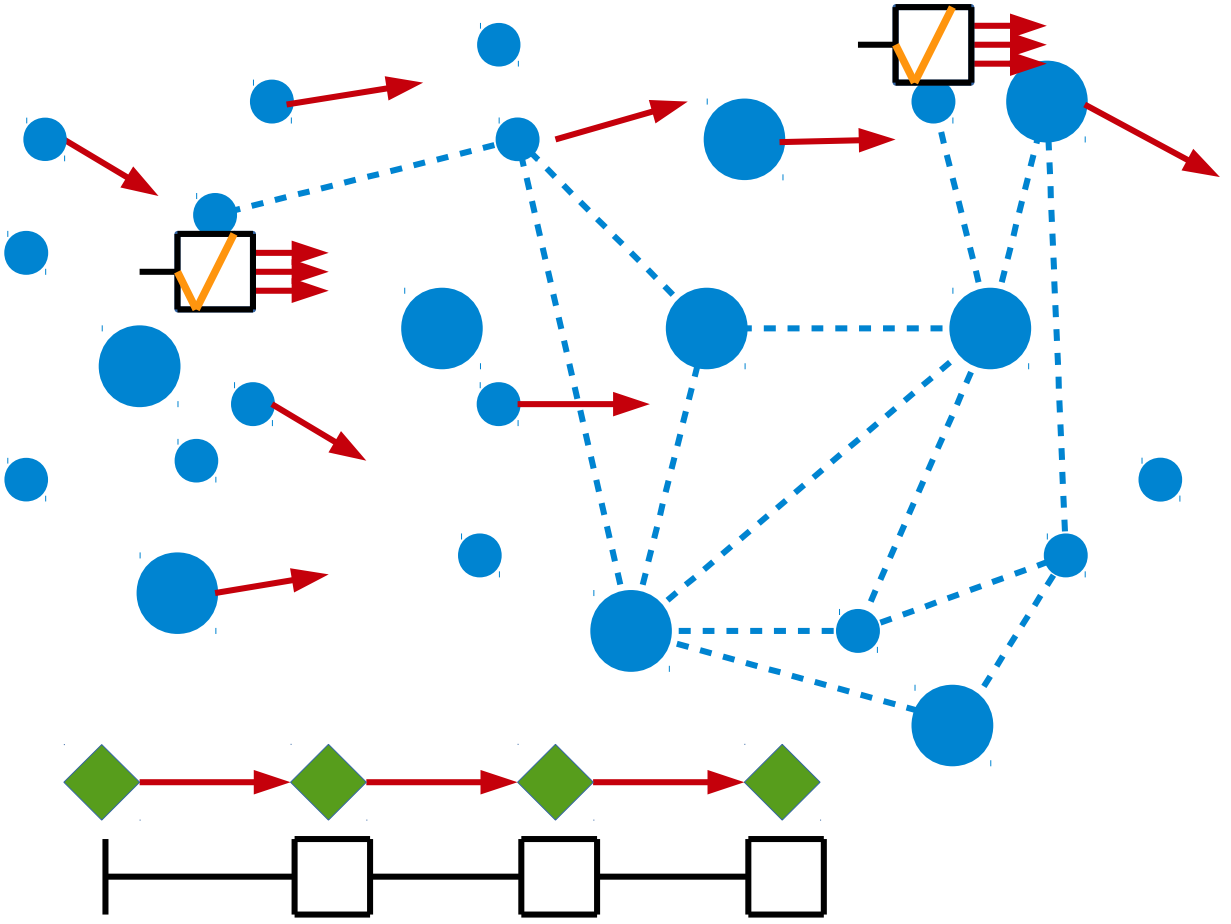
- Survives censorship and suppression
  - No central entity
- Nakamoto consensus requires proof-of-work (PoW)
  - Original intent: one CPU, one vote
  - Majority of hashing power controls network
  - Gives economic incentive to participate (solution to PoW is a newly "mined" Bitcoin)
- Today, total hashing work consumes a lot of electricity
  - Estimates vary, 250-1000MW, from a major city to a small country ...
- Protocol features
  - Stability is a tradeoff between dissemination of new block (10s-20s) and mining rate (new block on average every 10min)
  - Decisions are not final ("wait until chain is 6 blocks longer before a tx is confirmed")

Decentralized – deployment

- Bitcoin
  - Many (100s? 1000s?) of alt-coins and blockchains
- Ethereum
  - First digital currency with general-purpose smart contract execution
- Sawtooth ledger (Intel contribution to Hyperledger)
  - PoET consensus (proof of elapsed time)
    - Nodes run PoET program in "trusted execution environment" (Intel SGX)
    - PoET waits a random amount of time (say, E[wait] = 10min)
    - Creates an attested proof of elapsed time
    - Rest like in Bitcoin protocol

Somewhat (de)centralized – Ripple, Stellar

- Nodes decide whom to trust
  - Validator nodes
- Quorum-like protocol
  - Among different heterogeneous validators
- Consistent when "enough" nodes in path are trusted by the nodes involved in the transaction

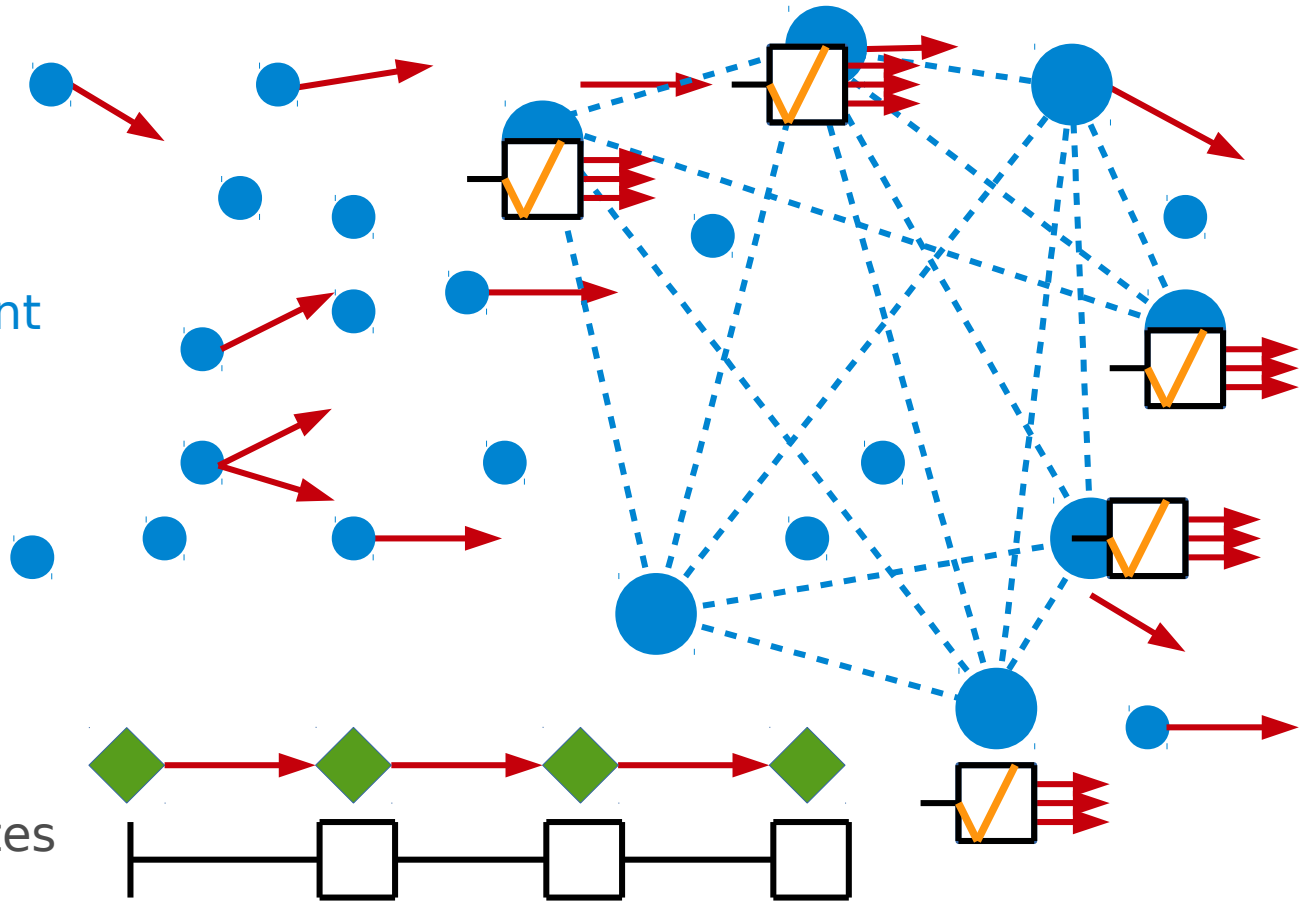


Somewhat (de)centralized

- Every node designates other, well-connected nodes that it trusts
  - Decentralization
- Transactions among two nodes are agreed if validated by "strong" majority in the overlap of the trusted node sets
  - In practice, leads to centralization
- Stellar evolved as fork of Ripple protocol
  - Issues with ledger forks and protocol correctness have been reported
- Open questions
  - Relation to Byzantine quorum systems [MR98]
  - How to formally specify properties

Consortium consensus (quorums & BFT)

- Designated set of homogeneous validator nodes
- BFT/Byzantine agreement
  - Tolerates f-out-of-n faulty/ adversarial nodes
  - Generalized quorums
- Tx sent to consensus nodes
- Consensus validates tx, decides, and disseminates result



Consortium consensus = permissioned

- Central entity controls group membership
  - Dynamic membership changes in protocol
  - Membership may be decided inline, by protocol itself
- Well-understood problem in distributed computing
  - BFT and consensus studied since ca. 1985
    - Clear assumptions and top-down design
    - 700 protocols and counting [AGK+15]
    - Textbooks [CGR11]
    - Open-source implementations (BFT-SMaRT)
  - Many systems already provide crash tolerant consensus (Chubby, Zookeeper, etcd ...)
  - Requires $\Omega(n^2)$ communication (OK for 10-100 nodes, not > 1000s)
- Revival of research in BFT protocols
  - Focus on scalability and communication efficiency

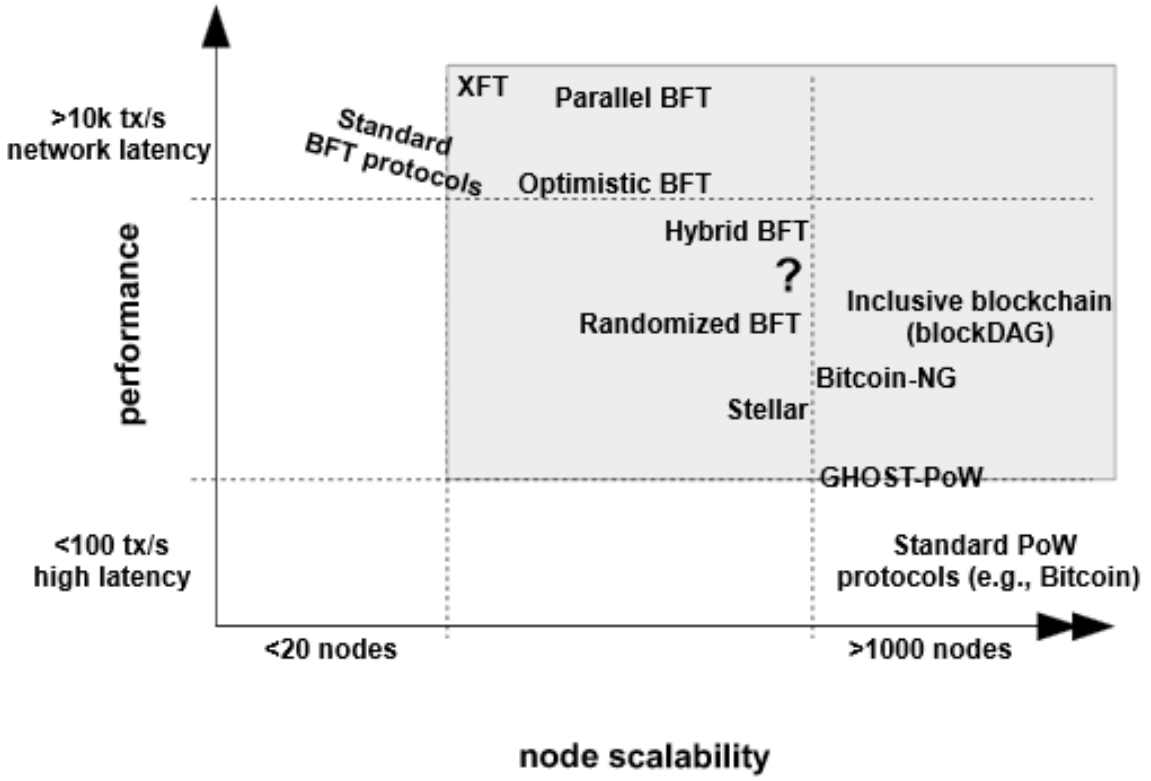
Consortium consensus – under development

- Hyperledger Fabric (originally started by IBM)
  - Includes PBFT protocol [CL02]
- Tendermint, Juno/Kadena, JPMC Quorum, Axoni, Iroha, Chain and others
- HoneyBadgerBFT [MXC+16]
  - Revisits practical randomized BFT [CKPS01], including amoritzation
- Many existing BFT libraries predate blockchain
  - BFT-SMaRT, Univ. Lisbon (github.com/bft-smart/library)
  - Prime, Johns Hopkins Univ. (www.dsn.jhu.edu/byzrep/prime.html)

More variations of consensus

- Bitcoin-NG [EGS+16]
  - Bitcoin PoW elects a leader, it is responsible for ordering the next K tx
- Proof-of-stake (explored by Ethereum)
  - Voting power relative to asset holdings (through cryptocurrency held by blockchain)
- Hybrid PoW (PeerCensus [DSW16])
  - PoW protocol to elect nodes in one consensus group
  - Group runs ordinary BFT consensus
- Hierarchical & partitioned, randomized [LNB+15]
  - Random sub-groups, nodes and tx assigned randomly to sub-groups
  - Each sub-group runs ordinary BFT consensus

Scalability–performance tradeoff



M. Vukolic: The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication. Proc. iNetSec 2015, LNCS 9591.

Validation

Validation of transactions – PoW protocols

- Recall validation predicate P on state s and operation o: P(s, o) ✓

- When constructing a block, the node
  - Validates all contained tx
  - Decides on an ordering within block

- When a new block is propagated, all nodes must validate the block and its tx
  - Simple for Bitcoin – verify digital signatures and that coins are unspent
  - More complex and costly for Ethereum – re-run all the smart-contract code

- Validation can be expensive
  - Bitcoin blockchain contains the log of all tx – 105GB as of 3/2017 (https://blockchain.info/charts/blocks-size)

Validation of transactions – BFT protocols

- Properties of ordinary Byzantine consensus
  - Weak Validity: Suppose all nodes are correct: if all propose v, then a node may only decide v; if a node decides v, then v was proposed by some node.
  - Agreement: No two correct nodes decide differently.
  - Termination: Every correct node eventually decides.

- Standard validity notions do not connect to the application!

- Need validity anchored at external predicate [CKPS01]
  - External validity: Given predicate P, known to every node, if a correct node decides v, then P(v); additionally, v was proposed by some node.

  - Can be implemented with digital signatures on input tx

Public validation vs. private state

- So far everything on blockchain is public – where is privacy?
- Use cryptography – keep state "off-chain" and produce verifiable tx
  - In Bitcoin, verification is a digital signature by key that owns coin
  - In ZeroCash [BCG+14], blockchain holds committed coins and transfers use zero-knowledge proofs (zk-SNARKS) validated by P
  - Hawk [KMS+16] uses verifiable computation (VC)
    - Computation using VC performed off-chain by involved parties
    - P checks correctness of proof for VC
- Private computation requires additional assumption (MPC, trusted HW ...)

Security and privacy

- Transactional privacy
  - Anonymity or pseudonymity through cryptographic tools
  - Some is feasible today (e.g., anonymous credentials in IBM Identity Mixer)
- Contract privacy
  - Distributed secure cryptographic computation on encrypted data
- Accountability & non-repudiation
  - Identity and cryptographic signatures
- Auditability & transparency
  - Cryptographic hash chain
- Many of these need advanced cryptographic protocols

Hyperledger Fabric

Hyperledger

- A Linux Foundation project – www.hyperledger.org
  - Open-source collaboration, developing blockchain technologies for business
  - Started in 2016: Hyperledger unites industry leaders to advance blockchain technology
  - 135 members in Apr. '17

- Incubates and promotes blockchain technologies for business
- Today 4 frameworks and 4 tools, hundreds of contributors

- Hyperledger Fabric was originally contributed by IBM – github.com/hyperledger/fabric/
  - Architecture and consensus protocols originally contributed by IBM Research - Zurich

PREMIER MEMBERS

GENERAL MEMBERS


众享比特 PeerSafe
guardtime
33.复杂美.CN
HASHED HEALTH
HEALTHCARE BLOCKCHAIN INNOVATION
HUAWEI
HUNDSUN
趣链科技 Hyperchain
intellect EU
intuit
INVESHARE
IRDOTECH
KAISER PERMANENTE
KRX KOREA EXCHANGE
KSD Korea Securities Depository
Financial IT Partner koscom
.kubique fill the gap
LedgerDomain
Libra
loyyal
Lykke
MadHive
Milligan Partners
MIRACL
MONAX INDUSTRIES
MonetaGo
ML
MOSCOW EXCHANGE
MURPHY & McGONIGLE A Professional Corporation
NSE
Orchestrating a brighter world NEC
NETKI
NEX
NOKIA
norbloc
NTT DATA Global IT Innovator
onchain
橙色魔方 Orange Magic Cube
PAXOS
PDX 全息互信
RadarWin 雷盈
redhat
SAMSUNG

Hyperledger Fabric

- Enterprise-grade blockchain fabric and distributed ledger framework
  - One of multiple blockchain platforms in the Hyperledger Project
  - First "active" platform under the Hyperledger umbrella (since 3/2017)

- Developed open-source, by IBM and others (DAH, LSEG ...)
  - github.com/hyperledger/fabric
  - Initially called 'openblockchain' and contributed by IBM to Hyperledger project
  - Actively developed, IBM and IBM Zurich play key roles

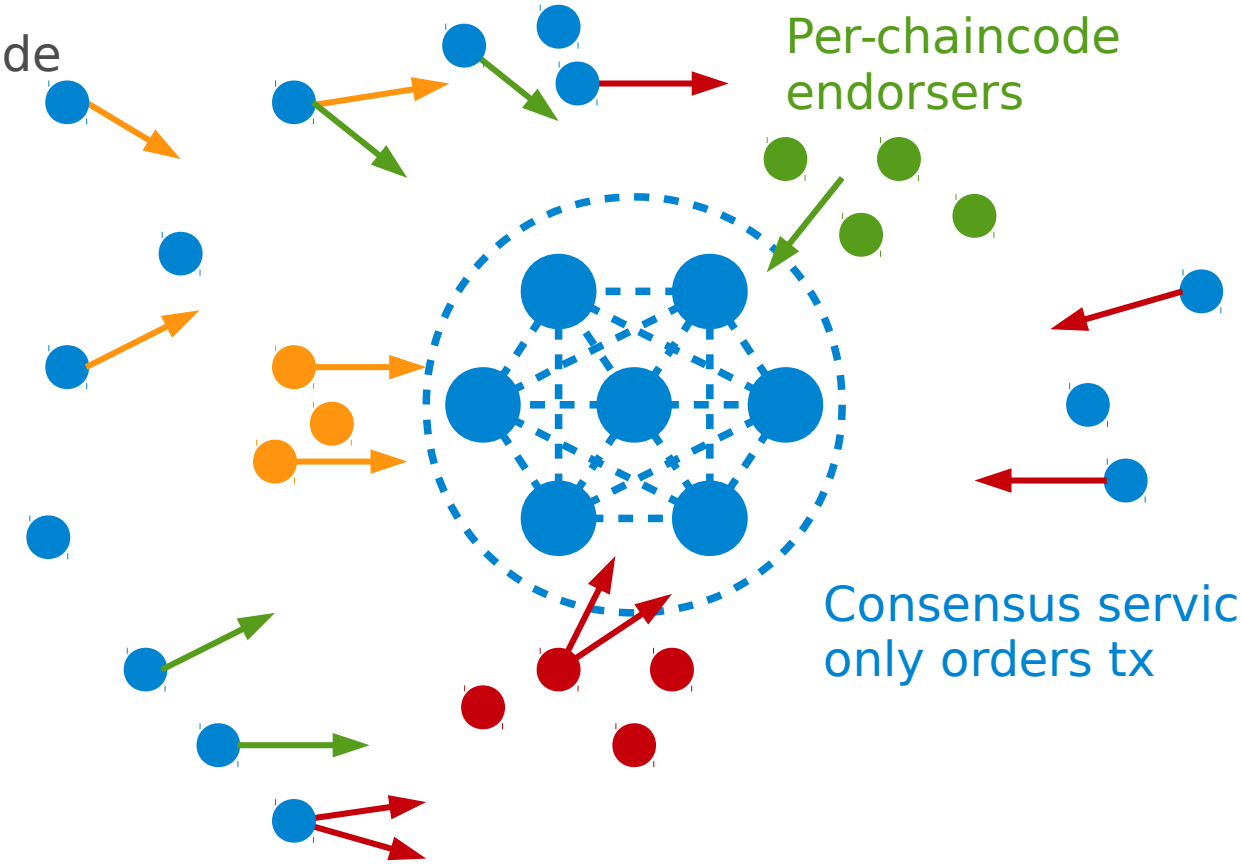
- Technical details
  - Implemented in GO
  - Runs smart contracts or "chaincode" within Docker containers
  - Transactions Deploy new chaincode / Invoke an operation / Read state
  - Implements consortium blockchain using traditional consensus (BFT, ZooKeeper)

Hyperledger Fabric V1

- Separate the functions of nodes into endorsers and consensus nodes
  - Every chaincode may have different endorsers
  - Endorsers have state, run tx, and validate tx for their chaincode
  - Chaincode specifies endorsement policy
  - Consensus nodes order endorsed and already-validated tx
  - All peers apply all state changes in order, only for properly endorsed tx
- Functions as replicated database maintained by peers [PWSKA00, KJP10]
  - Replication via (BFT) atomic broadcast in consensus
  - Endorsement protects against unauthorized updates
- Scales better – only few nodes execute, independent computations in parallel
- Permits some confidential data on blockchain via partitioning state
  - Data seen only by endorsers assigned to run that chaincode

Separation of endorsement from consensus

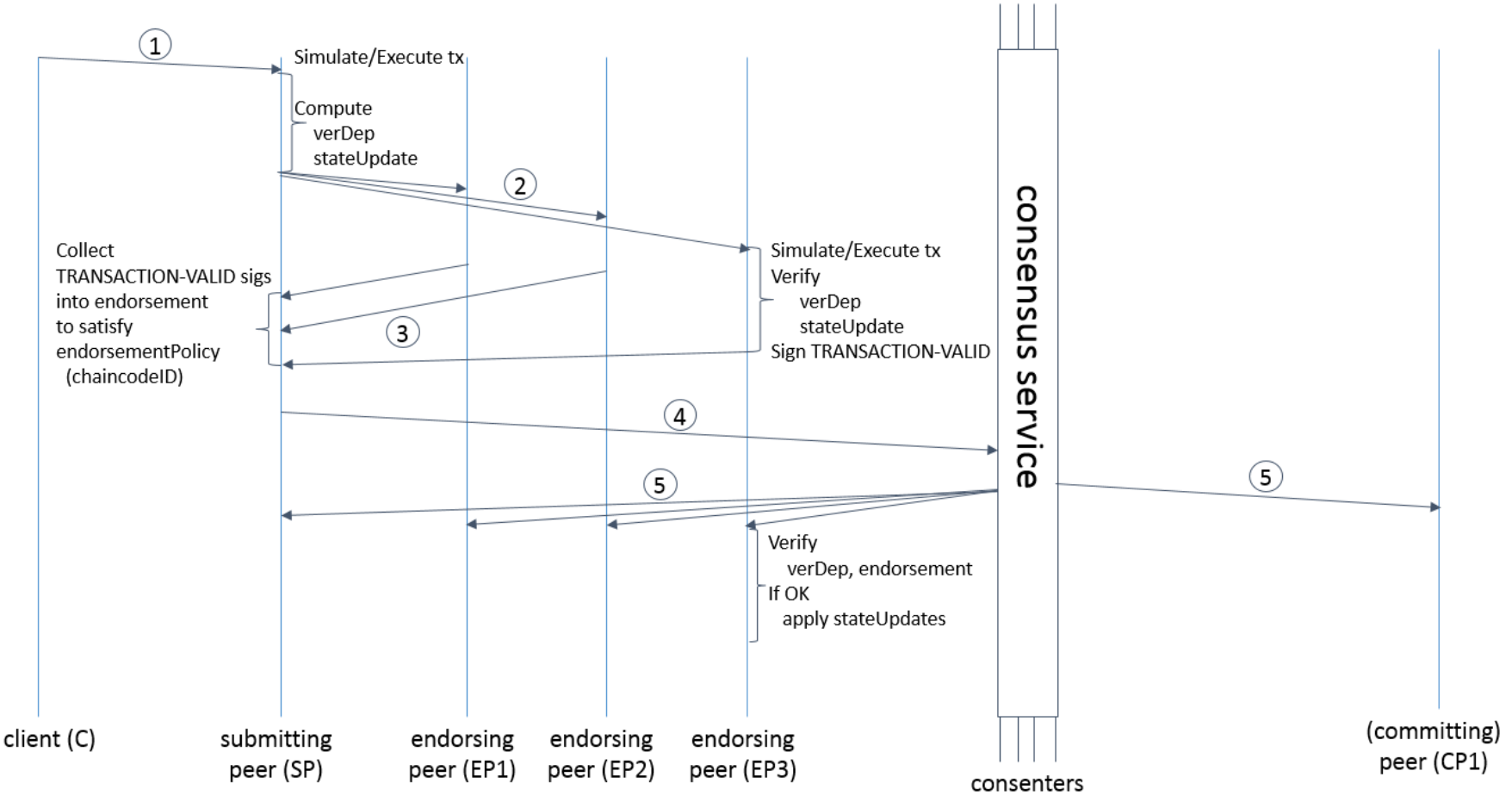
- Validation is by chaincode
- Dedicated endorsers per chaincode
- Consensus service
  - Only communication
  - Pub/sub messaging
  - Ordering for endorsed tx
- State and hash chain are common
  - State may be encrypted



Transactions in Fabric V1

- Client
  - Produces a tx (operation) for some chaincode (smart contract)
- Submitter peer
  - Execute/simulates tx with chaincode
  - Records state values accessed, but does not change state → readset/writeset
- Endorsing peer
  - Re-executes tx with chaincode and verifies readset/writeset
  - Endorses tx with a signature on readset/writeset
- Consensus service
  - Receives endorsed tx, orders them, and outputs stream of "raw" tx (=atomic broadcast)
- All peers
  - Disseminate tx stream from consensus service with p2p communication (gossip)
  - Filter out the not properly endorsed tx, according to chaincode endorsement policy
  - Execute state changes from readset/writeset of valid tx, in order

Transaction flow



Modular consensus in Fabric V1

- "Solo orderer"
  - One host only, acting as specification during development (ideal functionality)
- Apache Kafka, a distributed pub/sub streaming platform
  - Tolerates crashes among member nodes, resilience from Apache Zookeeper inside
  - Focus on high throughput
- SBFT - Simple implementation of PBFT (6/2017 - under development)
  - Tolerates f < n/3 Byzantine faulty nodes among n
  - Focus on resilience

Conclusion

Conclusion

- Blockchain enables new trust models
- Many interesting technologies
  - Distributed computing for consensus
  - Cryptography for integrity, privacy, anonymity
- We are only at the beginning
- Blockchain = Distributing trust over the Internet
  - www.hyperledger.org
  - www.ibm.com/blockchain/
  - www.research.ibm.com/blockchain/

References

[AGK+15] P.-L. Aublin, R. Guerraoui, N. Knezevic, V. Quéma, M. Vukolic: The Next 700 BFT Protocols. ACM TOCS, 32(4), 2015.

[BCG+14] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, M. Virza: Zerocash: Decentralized Anonymous Payments from Bitcoin. IEEE S&P 2014.

[CKPS01] C. Cachin, K. Kursawe, F. Petzold, V. Shoup: Secure and Efficient Asynchronous Broadcast Protocols. CRYPTO 2001.

[CGR11] C. Cachin, R. Guerraoui, L. Rodrigues: Introduction to Reliable and Secure Distributed Programming (2. ed.). Springer, 2011.

[CSV16] C. Cachin, S. Schubert, M. Vukolic: Non-determinism in Byzantine Fault-Tolerant Replication. OPODIS 2016.

[CL02] M. Castro, B. Liskov: Practical Byzantine fault tolerance and proactive recovery. ACM TOCS, 20(4), 2002.

[DSW16] C. Decker, J. Seidel, R. Wattenhofer: Bitcoin meets strong consistency. ICDCN 2016.

[EGS+16] I. Eyal, A. Gencer, E.G. Sirer, R. van Renesse: Bitcoin-NG: A Scalable Blockchain Protocol. NSDI 2016.

References

[KMS+16] A. Kosba, A. Miller, E. Shi, Z. Wen, C. Papamanthou: Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts. IEEE S&P 2016.

[LNB+15] L. Luu, V. Narayanan, K. Baweja, C. Zheng, S. Gilbert, P. Saxena: A Secure Sharding Protocol For Open Blockchains. ACM CCS 2016.

[MR98] D. Malkhi, M. Reiter: Byzantine Quorum Systems. Distributed Computing, 1998.

[MXC+16] A. Miller, Y. Xia, K. Croman, E. Shi, D. Song: The Honey Badger of BFT Protocols. ACM CCS 2016.

[V16] M. Vukolic: The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication. LNCS 9591, Proc. iNetSeC 2015.

Hyperledger Fabric references

- www.hyperledger.org
- Docs – hyperledger-fabric.readthedocs.io/en/latest/
- Chat – chat.hyperledger.org, all channels like #fabric-*
- Designs – wiki.hyperledger.org/community/fabric-design-docs
- Architecture of V1 – github.com/hyperledger/fabric/blob/master/proposals/r1/Next-Consensus-Architecture-Proposal.md
- Code – github.com/hyperledger/fabric

A shorter version of this paper appears in *Proc. Intl. Conference on Dependable Systems and Networks (DSN-2001)*, Gothenborg, Sweden, IEEE, 2001.

Distributing Trust on the Internet

Christian Cachin

IBM Research
Zurich Research Laboratory
CH-8803 Rüschlikon, Switzerland
cca@zurich.ibm.com

March 8, 2001

Abstract

This paper describes an architecture for secure and fault-tolerant service replication in an asynchronous network such as the Internet, where a malicious adversary may corrupt some servers and control the network. It relies on recent protocols for randomized Byzantine agreement and for atomic broadcast, which exploit concepts from threshold cryptography. The model and its assumptions are discussed in detail and compared to related work from the last decade in the first part of this work, and an overview of the broadcast protocols in the architecture is provided. The standard approach in fault-tolerant distributed systems is to assume that at most a certain fraction of servers fails. In the second part, novel general failure patterns and corresponding protocols are introduced. They allow for realistic modeling of real-world trust assumptions, beyond (weighted) threshold models. Finally, the application of our architecture to trusted services is discussed.

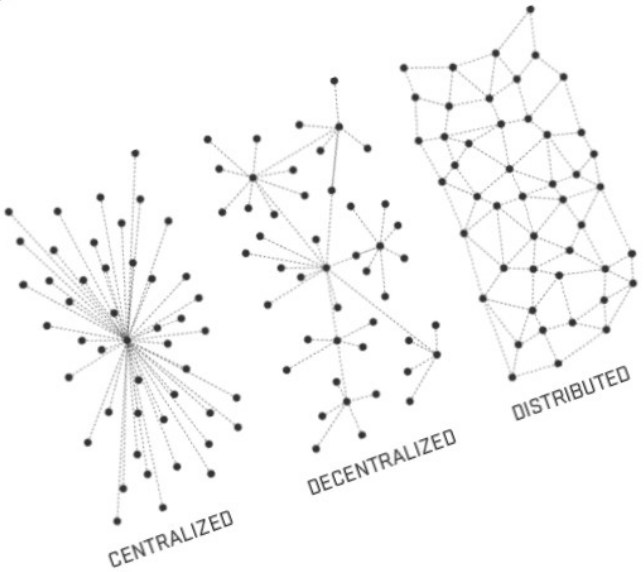
Consensus?

Consensus?

Consensus-as-a-service: a brief report on the emergence of permissioned, distributed ledger systems

By Tim Swanson

Published: April 6, 2015



Consensus?

Consensus-as-a-service: a brief report on the emergence of permissioned, distributed ledger systems

By Tim Swanson

Published: April 6, 2015

Consensus?

Consensus-as-a-service: a brief repo
systems

By Tim Swanson

Published: April 6, 2015

Consensus?

guardtime

Black Lantern Solutions Industries About Blog

A Distributed Consensus Engine for Digital Transactions

A Ledger without a Currency

KSI Ledger is a KSI Blockchain backed, distributed, permissioned ledger, leveraging KSI's proprietary, scalable consensus model and rock solid trust anchor.

KSI Ledger properties:

- No currency
- Provable audit trail
- Scalable to global transaction volumes

Assets: can be defined by any permissioned user, they are textual Ricardian Contracts and are represented by the hash value.

Accounts: represents an entities' balance of an Asset. An Account relates to one specific asset – multiple Accounts are needed for multiple Assets.

Issues: the primary bridge for assets from the traditional banking system to KSI Ledger and back. Implemented as a vostro/nostro account at a settlement house, who responds to issues with a

Consensus?

guardtime

Black Lantern Solution

A Distributed Consens
for Digital Transac

A Ledger without a

KSI Ledger is a KSI Blockchain backed, distributed, permissioned ledger, leveraging KSI's proprietary, scalable consensus model and rock solid trust anchor.

KSI Ledger properties:

- No currency
- Provable audit trail
- Scalable to global transaction volumes

News World Business Fintech Politics Technology Science Sport Entertainment

International Business Times

Business Fintech Blockchain Bitcoin

Tendermint talks about consensus and the 'internet of blockchains'

Tendermint founder Jae Kwon and CTO Ethan Buchman recently launched Cosmos, a hub to connect blockchains.

By Ian Allison
November 3, 2016 08:38 GMT

Jae Kwon, founder and CEO, Tendermint, has been studying the properties of consensus systems since he first became interested in Bitcoin. In a 1988 paper entitled, Consensus in the Presence of Partial Synchrony, Kwon saw the blueprint for the Tendermint proof of stake algorithm he would go on to build.

"I dropped everything and started working on it, and that was March of 2014," says Kwon. "The algorithm has evolved and gone through several iterations and now it turns out that what we have created is very similar to Practical Byzantine Fault Tolerance."

Issues: the primary bridge for
system to KSI Ledger and back. Implement
account at a settlement house, who responds to issues

NEW 11 JULY - 25 SEPTEMBER 2016 9:30 AM - 8:00 PM

IBT VIDEO

Which companies shou
the shrinking pound?

COLUMNISTS

Consen


NEWS
PRICE & DATA
GUIDES
EVENTS
RESEARCH
PRESS RELEASES
consensus 2017
MAKING BLOCKCHAIN REAL
MAY 22-24, 2017
NYC
Only 45 more tickets available at $999.
About
Register
Agenda
Sponsors
Hotel
CONSENSUS 2017
CoinDesk is proud to present our 3rd annual blockchain technology summit, May 22-24, 2017 at the New York Marriott Marquis. Consensus 2017 will feature 100+ speakers and 2,000+ attendees from the leading industry startups, investors, financial institutions, enterprise tech leaders, and academic and policy groups who are building the foundations of the blockchain and digital currency economy.
We expect to shatter our 2016 attendance record, so reserve your spot today!
REGISTER NOW
National Business Times
Technology
Science
Sport
Entertainment
IBT VIDEO
What
shrinki
COLUMNISTS

Hyperledger Fabric details (v0.6-preview) / 1

- Platform-agnostic
  - GO, gRPC over HTTP/2
- Peers
  - Validating peers (all running consensus) and non-validating peers
- Transactions
  - Deploy new chaincode / Invoke an operation / Read state
  - Chaincode is arbitrary GO program running in a Docker container
- State is a key-value store (RocksDB)
  - Put, get ... no other state must be held in chaincode
  - Non-validating peers store state and execute transactions

Hyperledger Fabric details / 2

- Consensus in BFT model
  - Modular architecture supports other consensus protocols
  - Currently, Practical Byzantine Fault Tolerance (PBFT) [CL02]
  - Non-determinism addressed by Sieve protocol [CSV16]
  - Static membership in consensus group
- Hash chain computed over state and transactions

Hyperledger Fabric details / 3

- Membership service issues certificates to peers
  - Enrollment certificates (E-Cert, issued by E-CA)
    - Assign identity to peer, gives permission to join and issue transactions
  - Transaction certificates (T-Cert, issued by T-CA)
    - Capability to issue one transaction (or more)
    - Unlinkable to enrollment certificate, for anyone except for transaction CA
- Pseudonymous transaction authorization
  - Controlled by peer, how many Transaction-Signatures with same T-Cert

Hyperledger Fabric details (V0.6)

- Peers
  - Validating peers (all running consensus) and non-validating peers
- Membership service issues identity-certificates and transaction-certificates
- Transactions
  - Deploy new chaincode / Invoke an operation / Read state
  - Chaincode is arbitrary GO program running in a Docker container
- State is a key-value store (RocksDB)
  - Put, get ... no other state must be held in chaincode
  - Non-validating peers store state and execute transactions
- Hash chain computed over state (and possibly transactions)

Non-determinism in BFT replication [CSV16]

- Service-replication paradigm needs deterministic state machines
  - Agree on order of operations, then every node executes
- What if application is given as black-box? Deterministic? Undecidable!
- Our approach – filter out inadvertent non-determinism
  - Execute operation, compare results, and revert it if "too much" divergence is evident
  - When "enough" nodes arrive at the same result, accept it
- If application is randomized
  - For algorithmic purpose (Monte Carlo): use master-slave approach
  - For cryptography and security functions: cryptographic verifiable random functions (VRF)