ITU Workshop on "Security Aspects of Blockchain" (Geneva, Switzerland, 21 March 2017)

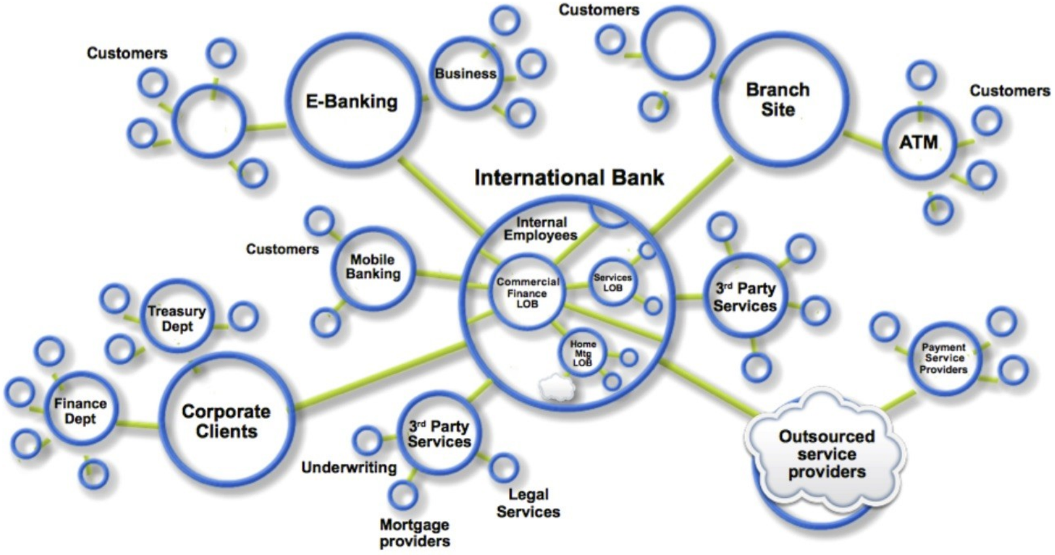
Blockchain, cryptography, and consensus

Dr. Christian Cachin
IBM Research - Zurich
www.zurich.ibm.com/~cca/

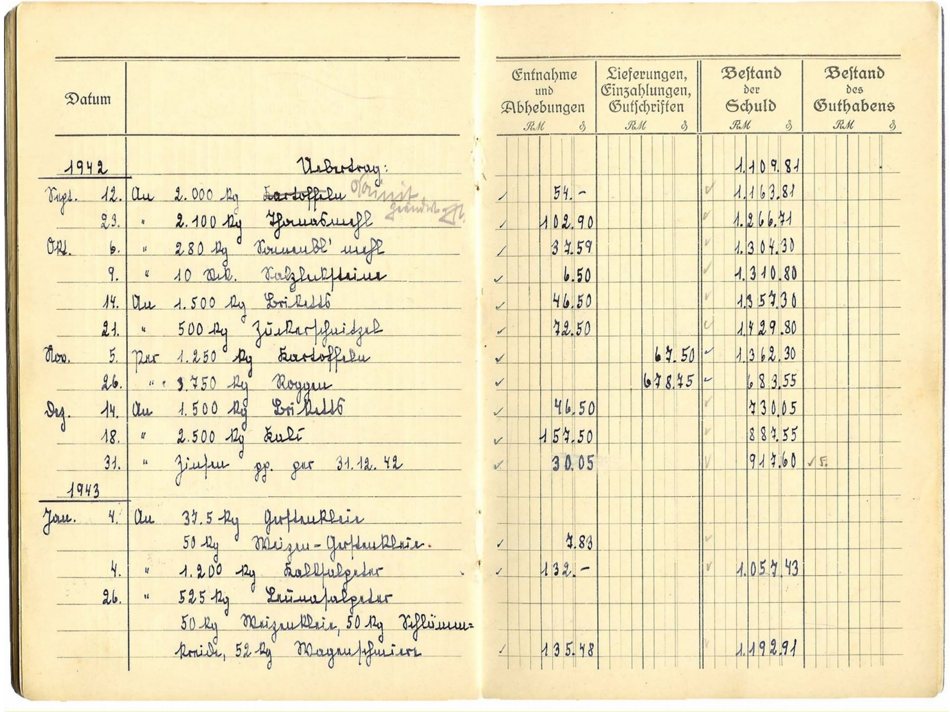
Geneva, Switzerland, 21 March 2017

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

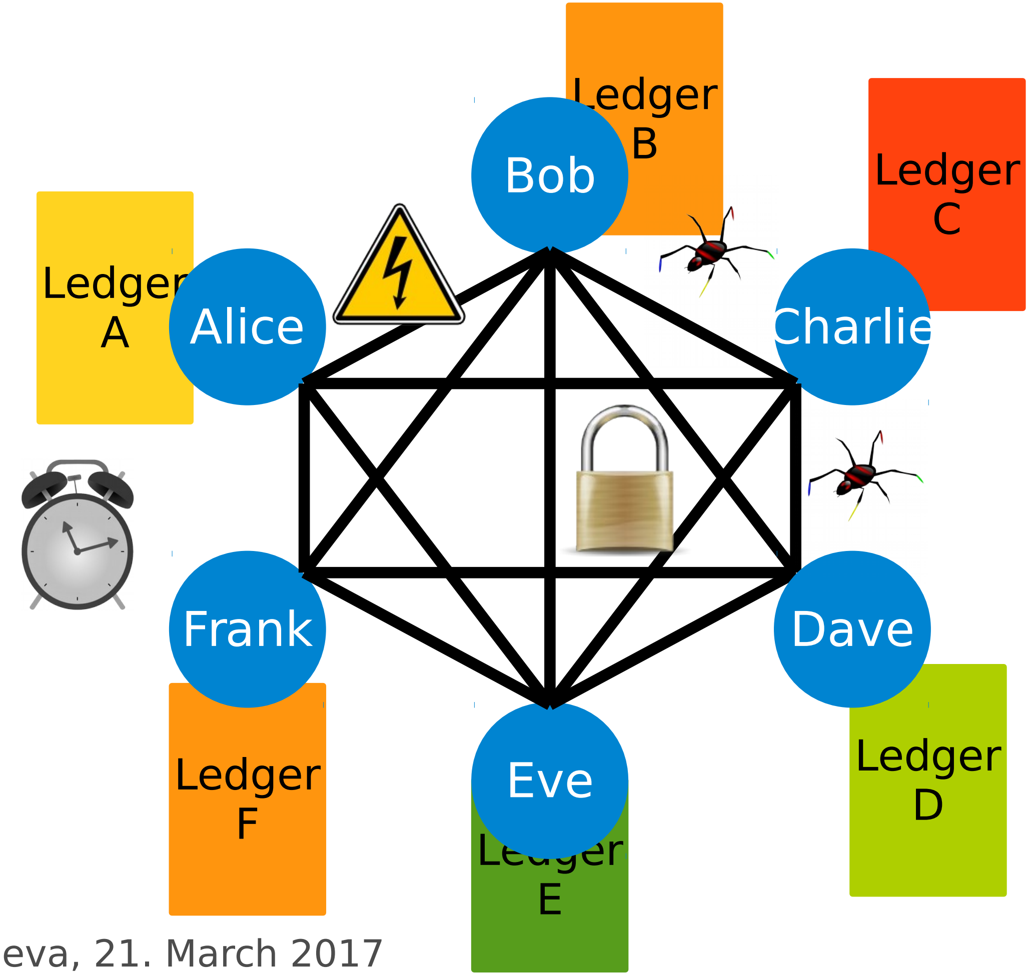


Ledger



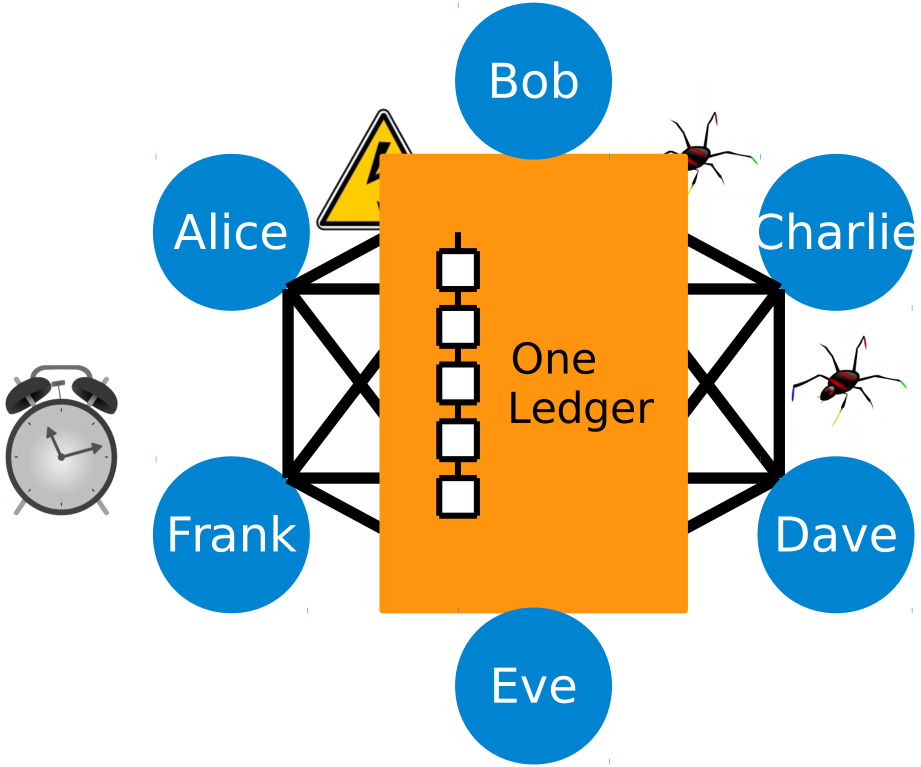
- Ledger records all business activity as transactions
  - Databases
- Every market and network defines a ledger
- Ledger records asset transfers between participants
- Problem — (Too) many ledgers
  - Every market has its ledger
  - Every organization has its own ledger

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

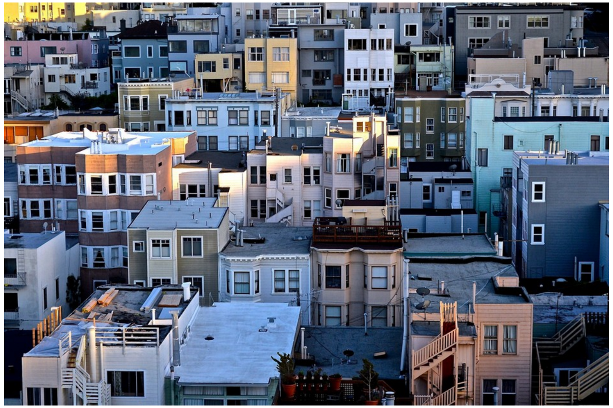
Business logic

- 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

Why blockchain now?

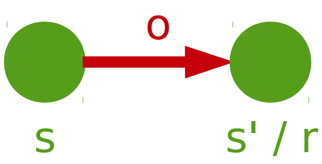
- Cryptography has been a key technology in the financial world for decades
  - Payment networks, ATM security, smart cards, online banking ...
- Trust model of (financial) business has not changed
  - Trusted intermediary needed for exchange among non-trusting partners
  - Today cryptography mostly 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?

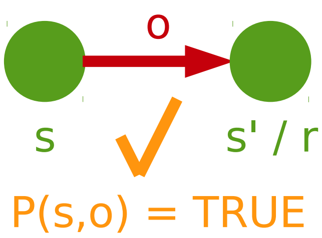
A state machine

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

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

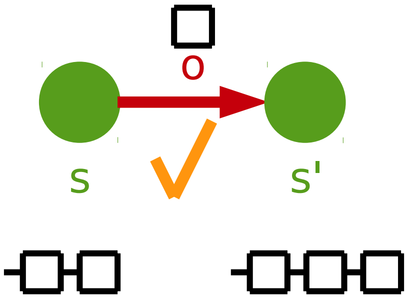


- Validation condition
  - Operation needs to be valid, in current state, according to a predicate P()



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

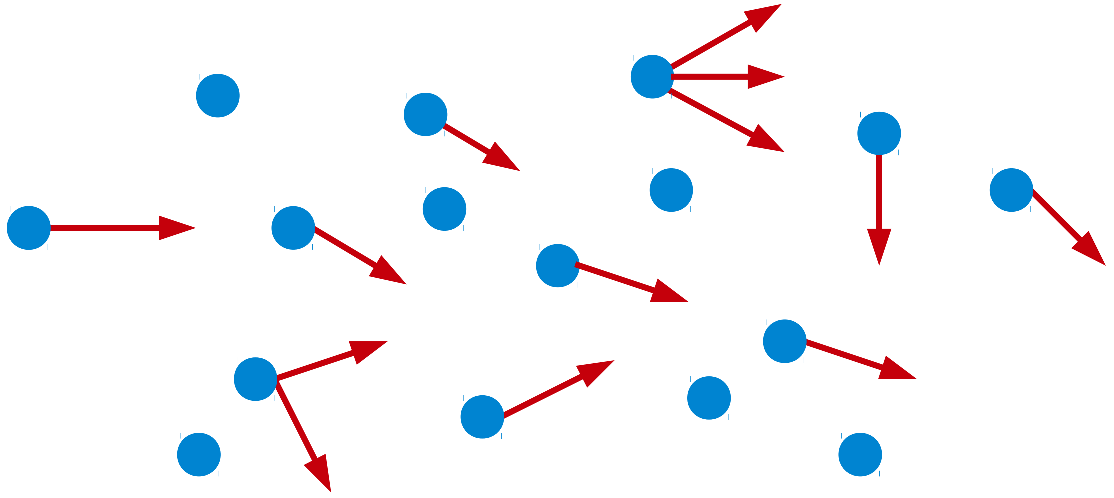
- Log entries form a hash chain

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

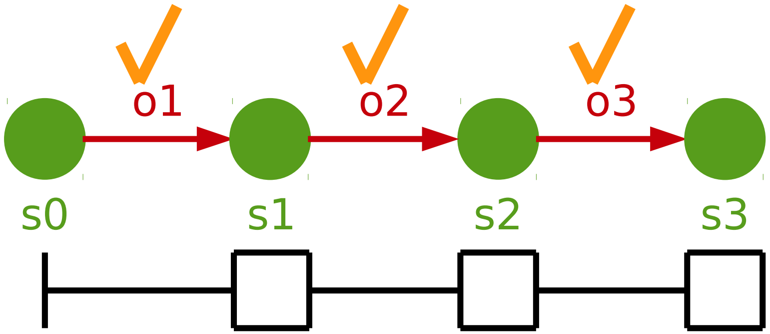
Example – The Bitcoin state machine

- 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"

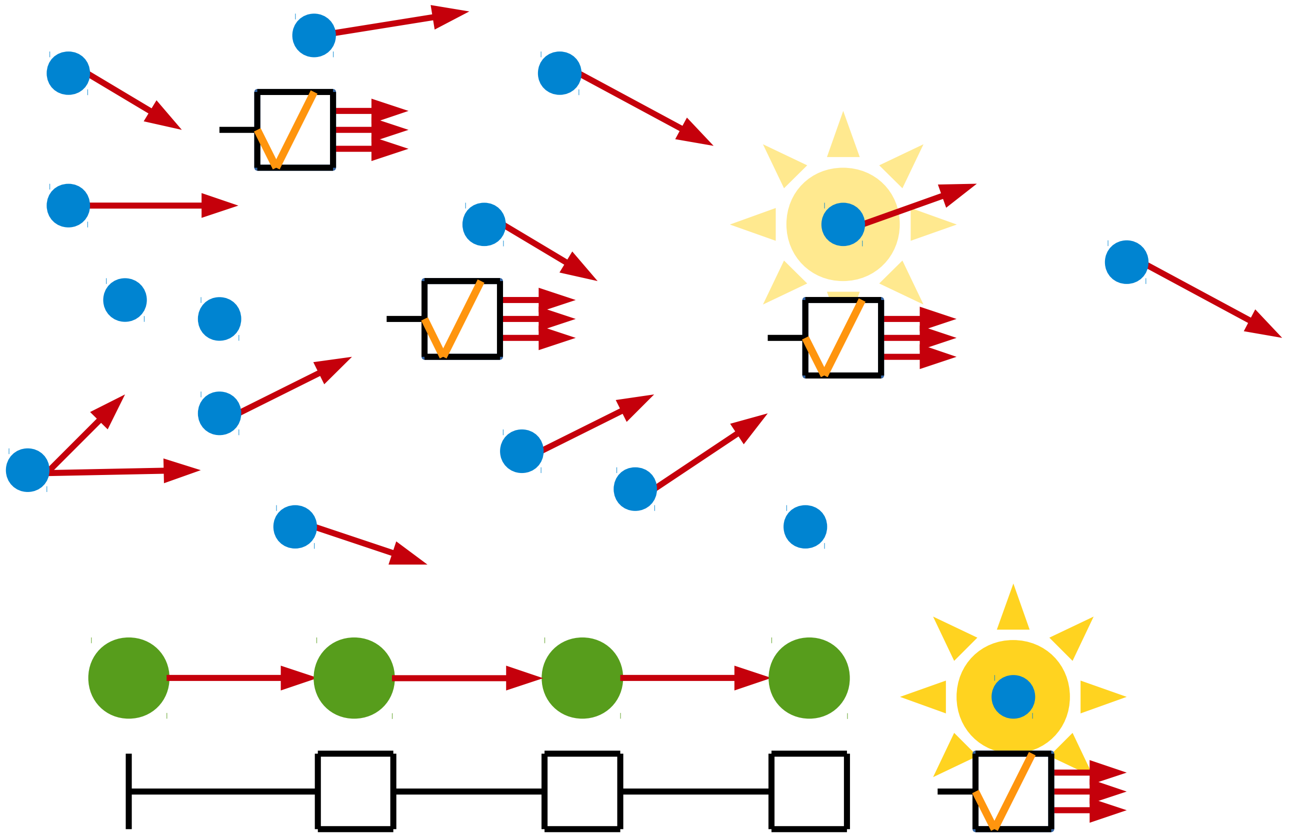
- Transactions can be simple
  - Bitcoin tx are statement of ownership for coins, digitally signed
    "This bitcoin now belongs to K2" signed by K1

- 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

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

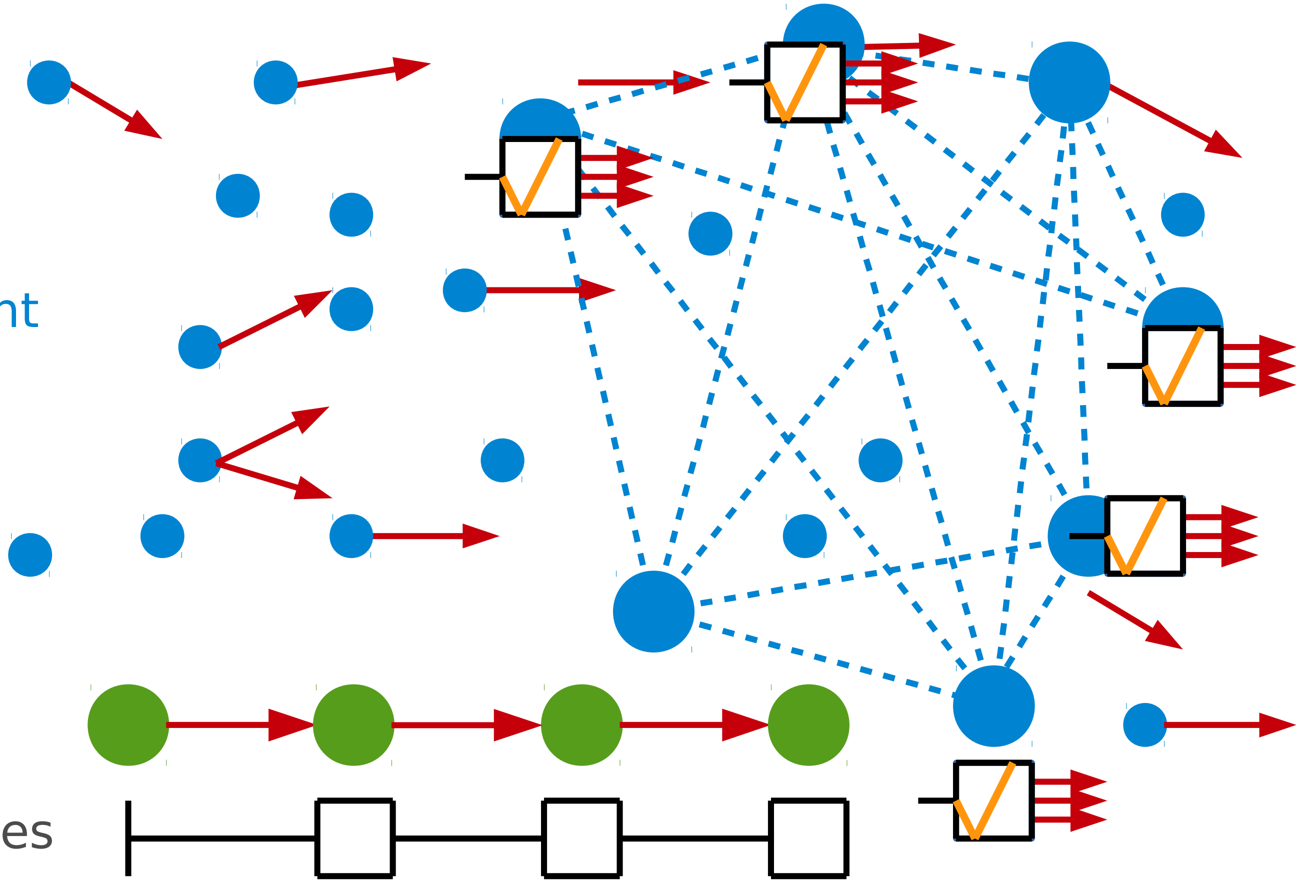
- Survives censorship and suppression
  - No central entity
- Nakamoto consensus requires proof-of-work (PoW)
  - Original intent: one CPU, one vote
  - Majority of hashing power controls network
  - Gives economic incentive to participate (solution to PoW is a newly "mined" Bitcoin)
- Today, total hashing 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

Consortium consensus (BFT, Hyperledger)

- 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

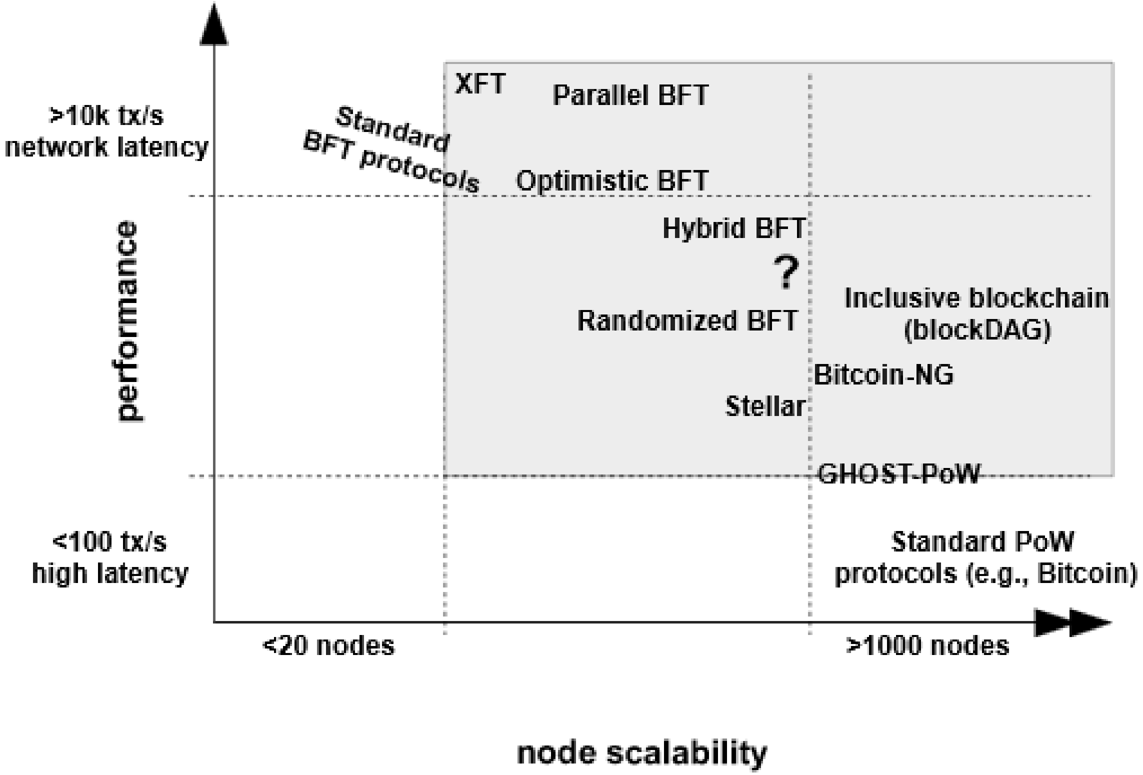
- Hyperledger fabric (IBM's contribution to Hyperledger)
  – Includes PBFT protocol [CL02]

- 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)

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 – 97GB as of 1/2017
    (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

Geneva, 21. March 2017

Hyperledger project

- Open-source collaboration under Linux Foundation
  - www.hyperledger.org
  - Hyperledger unites industry leaders to advance blockchain technology (Dec. '15)
  - 100 members in Jan. '17

HYPERLEDGER PROJECT

- Develops enterprise-grade, open-source distributed ledger technology

- Code contributions from several members

- Fabric is the IBM-started contribution – github.com/hyperledger/fabric/
  - Security architecture and consensus protocols from IBM Research - Zurich

PREMIER

accenture High performance. Delivered.
CME Group
DEUTSCHE BÖRSE GROUP
Digital Asset
DTCC
FUJITSU
HITACHI Inspire the Next
IBM
intel
J.P.Morgan
R3

GENERAL

ABN·AMRO
ANZ
belink
bitse
BLOCKCHAIN
Blockstream
bloq
BNY MELLON
Broadridge
Calastone
cisco
cloudsoft
CLS Fundamental to FX
coinplug
consensys
CREDITS
Cuscal The complete payments partner
evue digital labs
Gem
guardtime
intellect
INVeSHARE
itBit
Milligan Partners
MonetaGo
ML
MOSCOW EXCHANGE
Orchestrating a brighter world NEC
norbloc
NTT DATA
onchain
redhat
Ribbit.me
Skry
SORAMITSU
STATE STREET
SWIFT
symbiont
tequa creek
THOMSON REUTERS
vmware
WELLS FARGO

Geneva,

Hyperledger Fabric

- Enterprise-grade consortium blockchain and distributed ledger framework
  - A blockchain implementation in the Hyperledger Project
- Developed open-source, by IBM and others (DAH, LSEG ...)
  - github.com/hyperledger/fabric
  - Initially called 'openblockchain' and donated by IBM to Hyperledger project
  - Actively developed, IBM and IBM Zurich play key roles
- Technical details
  - Implemented in GO
  - Runs smart contracts ("chaincode") within Docker containers
  - Implements consortium blockchain using traditional consensus (BFT, Paxos)

Hyperledger Fabric details (V0.6)

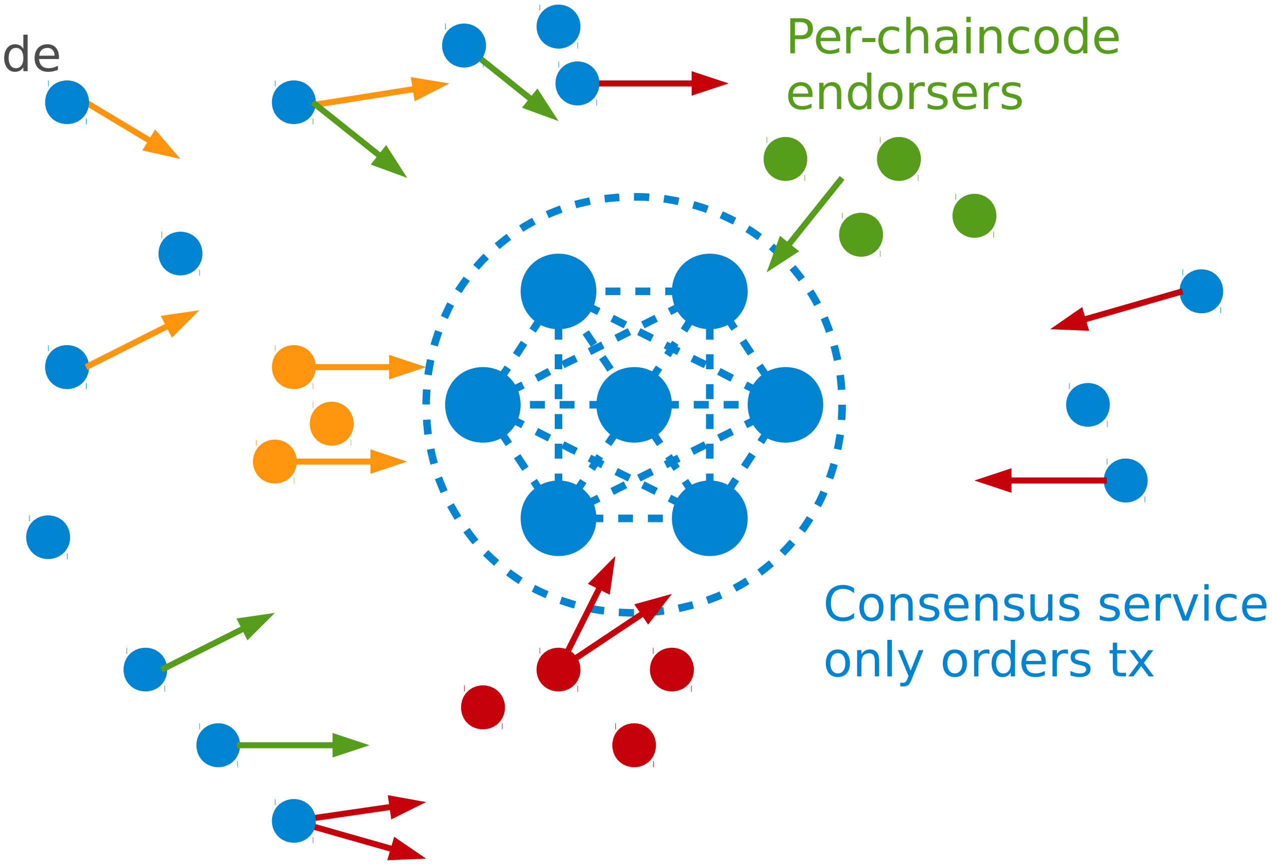
- Peers (validating peers and non-validating peers)
  - GO and other languages, gRPC over HTTP/2
  - Validating peers (all running consensus) and non-validating peers
- Membership service issues identity-certificates and transaction-certificates
- Transactions
  - Deploy new chaincode / Invoke an operation / 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)

Towards 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
  - Orders the endorsed tx, produces ordered stream of tx
  - Filters out the not properly endorsed tx, according to chaincode endorsement policy
- All peers
  - Disseminate tx from consensus service with p2p communication (gossip)
  - Execute state changes from readset/writeset of valid tx, in order

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, has Apache Zookeeper
  - Focus on high throughput
- SBFT - A simple implementation of Practical Byzantine Fault Tolerance (PBFT)
  - 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/