Hyperledger Fabric: A Platform for Distributing Trust

Christian Cachin

IBM Research – Zurich

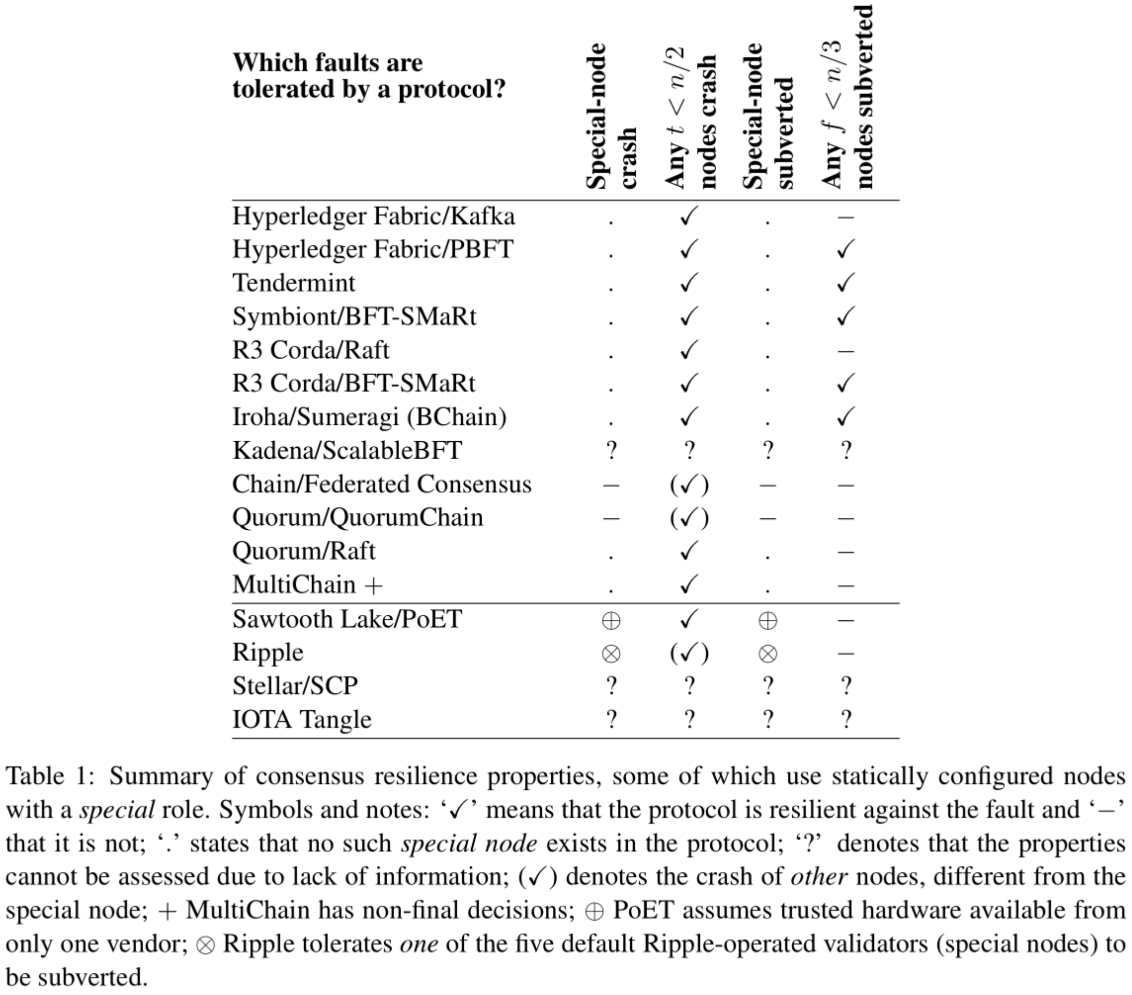
June 2018

How to design a blockchain?

- Blockchains are like cryptosystems
  - Must resist attacks
  - Resilient against unknown adversaries
- Impossible to demonstrate their security a priori, by demonstration
  - Only an attack shows how it fails
- Multiple ways to achieve security
  - Empirical validation
  - Mathematical proofs from broadly accepted assumptions
  - Public review, open discussion, standards

> The problem with bad security is that it looks just like good security. You can't tell the difference by looking at the finished product.
>
> - Both make the same security claims; both have the same functionality.
> - Both might use the same protocols, implement the same standards, and have been endorsed by the same industry groups.
> - Yet one is secure and the other is insecure.
>
> Bruce Schneier (1999)

Blockchain consensus protocols in the wild



Which faults are tolerated by a protocol?	Special-node crash	Any $t < n/2$ nodes crash	Special-node subverted	Any $f < n/3$ nodes subverted
Hyperledger Fabric/Kafka	.	✓	.	–
Hyperledger Fabric/PBFT	.	✓	.	✓
Tendermint	.	✓	.	✓
Symbiont/BFT-SMaRt	.	✓	.	✓
R3 Corda/Raft	.	✓	.	–
R3 Corda/BFT-SMaRt	.	✓	.	✓
Iroha/Sumeragi (BChain)	.	✓	.	✓
Kadena/ScalableBFT	?	?	?	?
Chain/Federated Consensus	–	(✓)	–	–
Quorum/QuorumChain	–	(✓)	–	–
Quorum/Raft	.	✓	.	–
MultiChain +	.	✓	.	–
Sawtooth Lake/PoET	⊕	✓	⊕	–
Ripple	⊗	(✓)	⊗	–
Stellar/SCP	?	?	?	?
IOTA Tangle	?	?	?	?

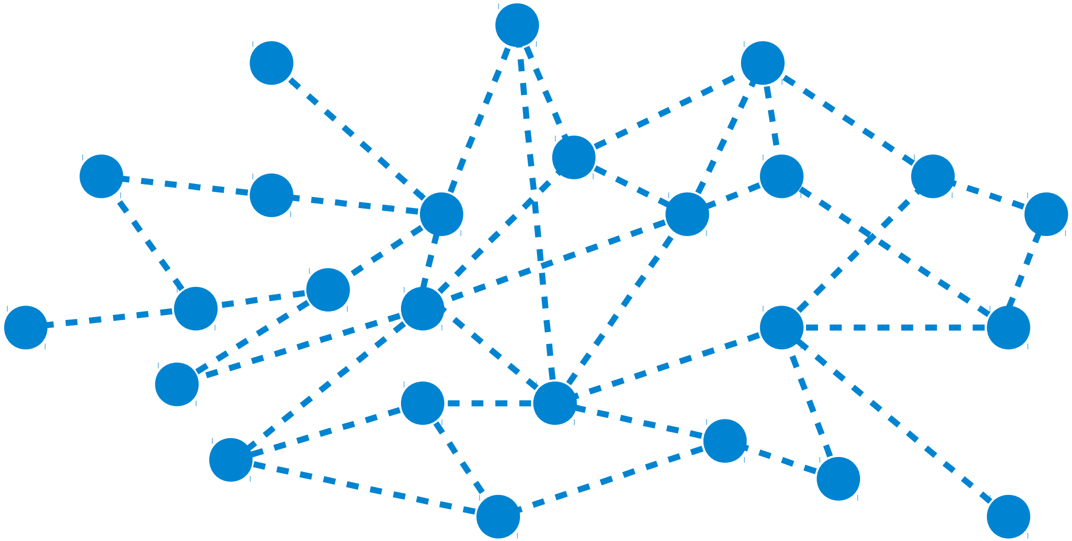
Table 1: Summary of consensus resilience properties, some of which use statically configured nodes with a *special* role. Symbols and notes: '✓' means that the protocol is resilient against the fault and '–' that it is not; '.' states that no such *special node* exists in the protocol; '?' denotes that the properties cannot be assessed due to lack of information; (✓) denotes the crash of *other* nodes, different from the special node; + MultiChain has non-final decisions; ⊕ PoET assumes trusted hardware available from only one vendor; ⊗ Ripple tolerates *one* of the five default Ripple-operated validators (special nodes) to be subverted.

C. Cachin, M. Vukolic: Blockchain consensus protocols in the wild, DISC 2017.

https://arxiv.org/abs/1707.01873

Blockchain consensus

Permissionless or decentralized blockchains



- Anyone can join
- Sybil attacks
- No traditional votes

- Bitcoin's idea: One CPU = One vote

- "Vote" by investing and proving work

Features of decentralized consensus

- Survives censorship and suppression **(+ / —)**
  - No identities, no counting of nodes
  - Give incentive to participate with mining reward

- Scales to 1000s of nodes **(+)**

- High latency (minutes or more), and decisions are never final **(—)**

- Requires proof-of-work (PoW) **(—)**
  - Majority of hashing power controls the network

- PoW = waste-of-work: Consensus proctocol consumes huge amounts of power
  - Bitcoin consumes 20% more electricity than Switzerland
    (bitcoinenergyconsumption.com // Bundesamt für Energie (BFE), Stromverbrauch 2017)

Consortium or permissioned blockchains

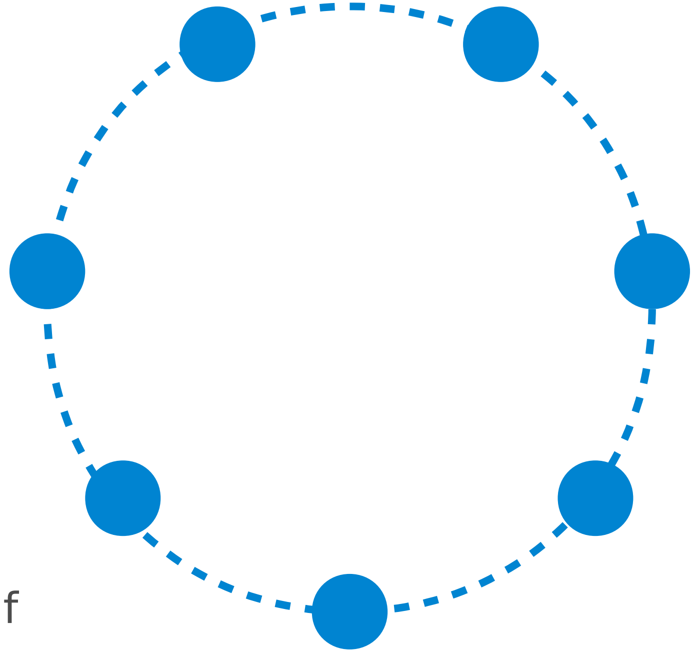
- Traditional BFT consensus based on voting

- Defined group of validator nodes

- Has been studied for decades
  - Byzantine Fault Tolerance (BFT)
  - Elaborate mathematical theory (quorums)
  - Clear assumptions and top-down design

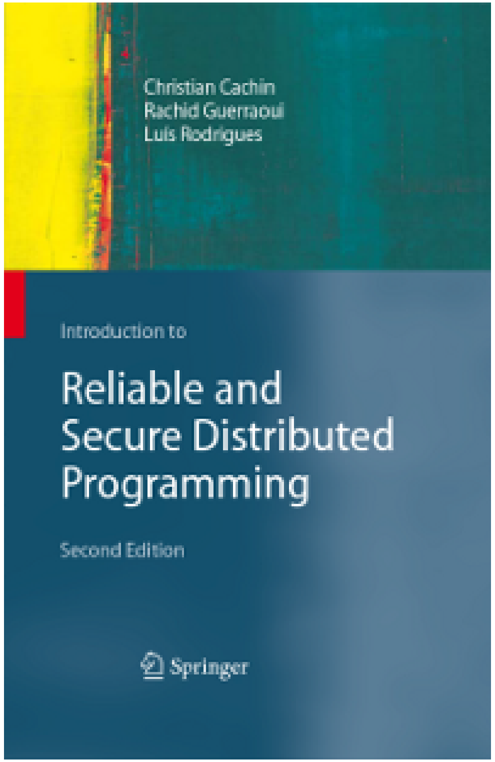
- Many variations possible
  - Change group membership through protocol itself
  - Votes weighted by stake

- Implementations available, some open source



History of BFT consensus

- Helped develop the field of distributed computing
  - The mathematical consensus abstraction plays a key role
  - Rich body of literature, textbooks …
- Computer-science theory research
  - Very active topic ca. 1985–2000
  - Many theorems, no systems (cf. Paxos …)
- Computer systems research
  - Very active topic ca. 1999–2010
  - Many systems, no deployment (cf. ZooKeeper, Raft/etcd …)
- Blockchain research and development
  - Revived interest, starting ca. 2015
  - Deployment in practice



Christian Cachin
Rachid Guerraoui
Luís Rodrigues

Introduction to

Reliable and Secure Distributed Programming

Second Edition

Springer

Features of BFT consensus

- Well-understood **(+)**
  - Many protocols, manny research papers, textbooks
  - Security proofs and open-source implementations

- Fast **(+)**
  - 1000s or 10'000s of tx/s
  - Latency of seconds

- Decisions are final **(+)**

- Usually requires all-to-all, $\Omega(N^2)$, communication **(—)**
  - Does not scale to 1000s of nodes

- Needs identities of nodes **(+ / —)**

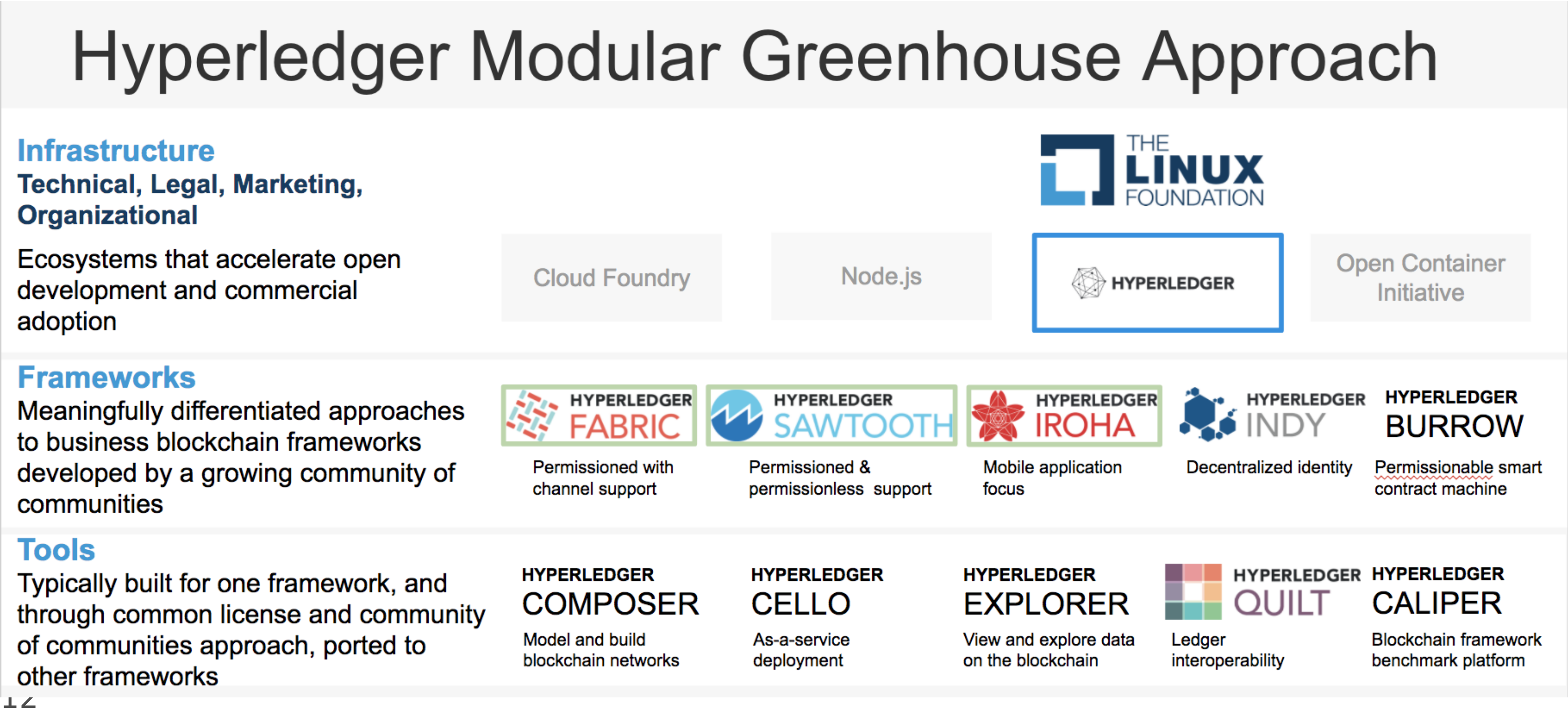
Hyperledger

Hyperledger

- Hyperledger – www.hyperledger.org
- Global collaboration hosted by the Linux Foundation
  - Advances blockchain technologies for business, neutral, community-driven
  - Started in 2016: Hyperledger unites industry leaders to advance blockchain technology
  - ca. 230 members in May '18
- Develops and promotes blockchain technologies for business
- Today 5 frameworks and 5 tools, hundreds of contributors

- Hyperledger Fabric – github.com/hyperledger/fabric/
  - One blockchain framework of Hyperledger

Hyperledger overview



Hyperledger Modular Greenhouse Approach

Infrastructure

Technical, Legal, Marketing, Organizational

Ecosystems that accelerate open development and commercial adoption

THE LINUX FOUNDATION

Cloud Foundry

Node.js

HYPERLEDGER

Open Container Initiative

Frameworks

Meaningfully differentiated approaches to business blockchain frameworks developed by a growing community of communities

HYPERLEDGER FABRIC
Permissioned with channel support

HYPERLEDGER SAWTOOTH
Permissioned & permissionless support

HYPERLEDGER IROHA
Mobile application focus

HYPERLEDGER INDY
Decentralized identity

HYPERLEDGER BURROW
Permissionable smart contract machine

Tools

Typically built for one framework, and through common license and community of communities approach, ported to other frameworks

HYPERLEDGER COMPOSER
Model and build blockchain networks

HYPERLEDGER CELLO
As-a-service deployment

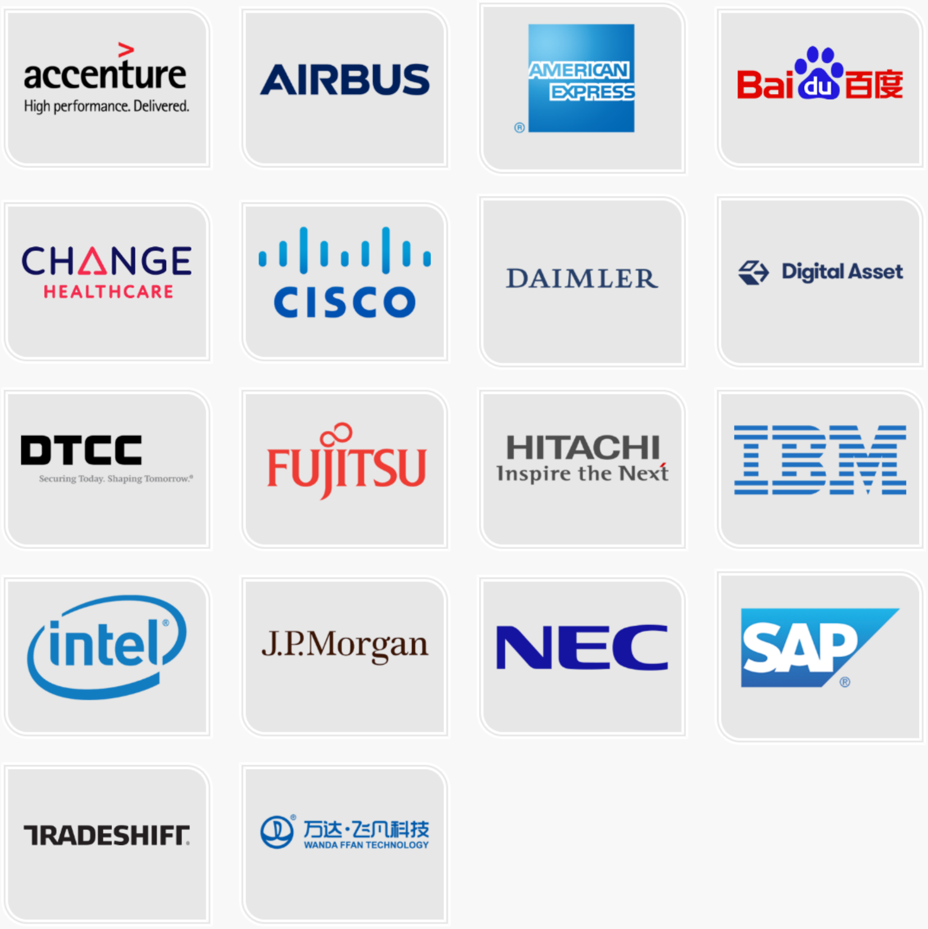
HYPERLEDGER EXPLORER
View and explore data on the blockchain

HYPERLEDGER QUILT
Ledger interoperability

HYPERLEDGER CALIPER
Blockchain framework benchmark platform

Hyperledger members

PREMIER MEMBERS



Hyperledger Fabric

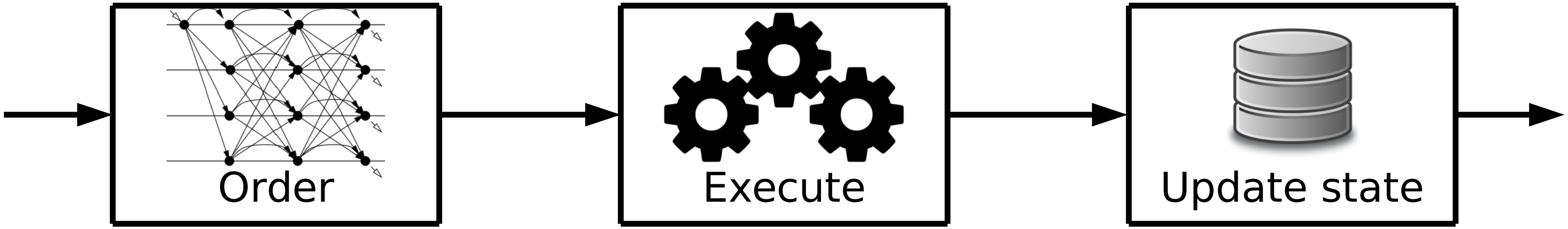
Hyperledger Fabric –
An enterprise blockchain platform

- Fabric is a distributed ledger framework for consortium blockchains
  - One of multiple blockchain platforms in the Hyperledger Project (V0.6 in Oct. '16)
  - First active platform in Hyperledger project and production-ready (V1.0 in Jul. '17)

- Developed open-source
  - github.com/hyperledger/fabric
  - Initially developed as *openblockchain* and contributed by IBM
  - Driven IBM, State Street, Digital Asset Holdings, HACERA and others
    - IBM Research – Zurich (Rüschlikon) produced important designs and key components
  - Key technology for IBM's blockchain strategy

- Technical details [Androulaki et al., Eurosys 2018, doi.org/10.1145/3190508.3190538]
  - Modular architecture (e.g., pluggable consensus, cryptography, languages, trust model)
  - Programmable consortium blockchain, implemented in GO
  - Runs smart contracts called "chaincode" within Docker containers

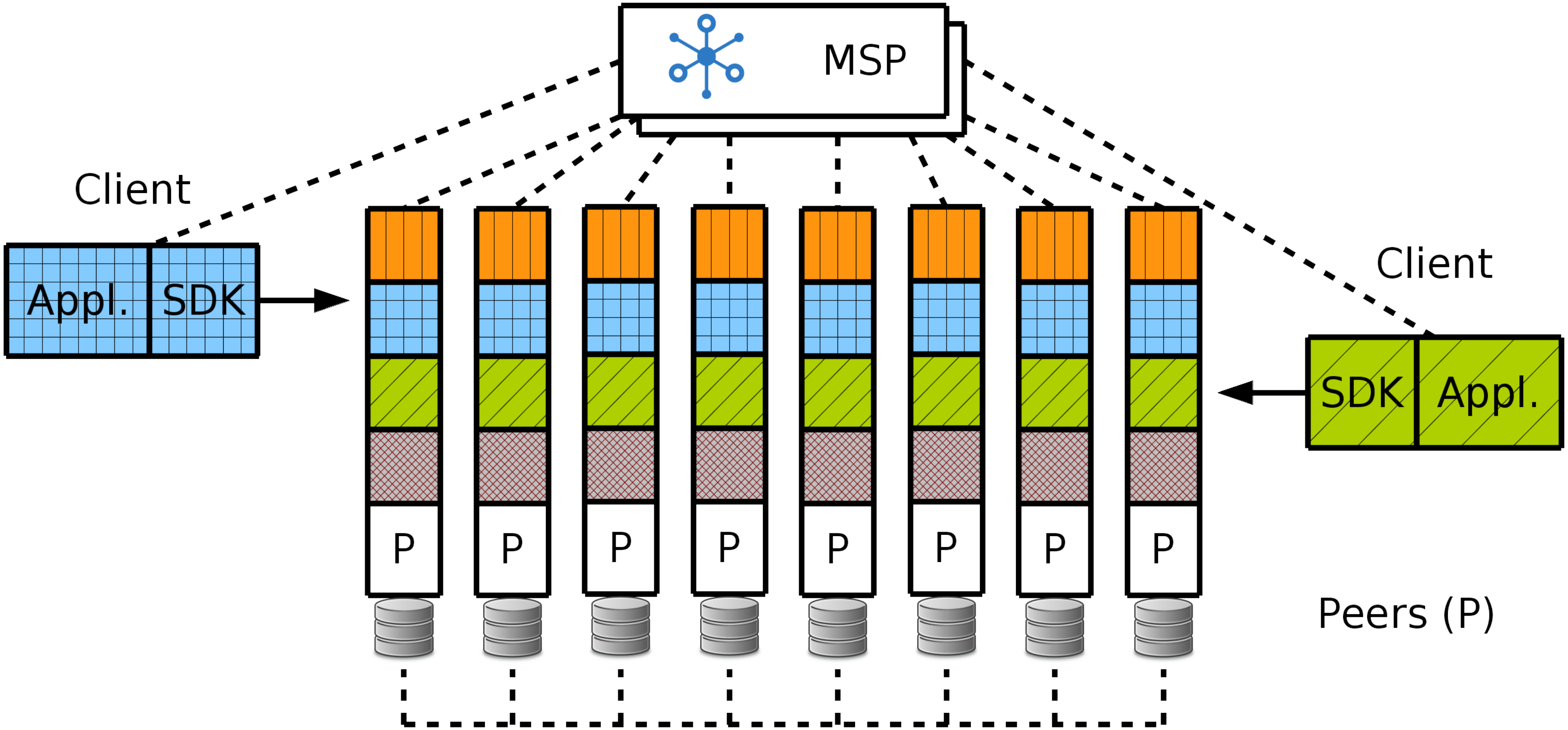
Traditional architecture – Replicated service



- Consensus or atomic broadcast
- Deterministic (!) tx execution
- Persist state on all peers

- All prior BFT systems operate as a replicated state machine [Schneider, ACM Comp. Surv. 1990]
- All other (permissioned) blockchains operate like this
  - Including Hyperledger Fabric until V0.6

Traditional architecture (including Fabric 0.6)



Issues with the traditional replication design

- Sequential execution
  – Increased latency – or – complex schemes for parallelism

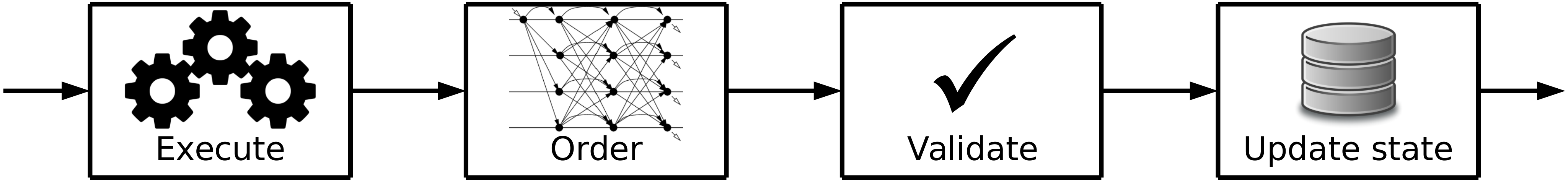
- Operations must be deterministic
  – Difficult to enforce with generic programming language (difficult per se!)
  – Modular filtering of non-deterministic operations is costly [Cachin et al., OPODIS 2016]

- Trust model is fixed for all applications (smart contracts)
  – Typically some (F+1) validator nodes must agree to result (at least one correct)
  – Fixed to be the same as in consensus protocol

- Privacy is difficult, as data spreads to all nodes
  – All nodes execute all applications

- All these are lessons learned from Hyperledger Fabric, before V0.6

Fabric V1 architecture



- Simulate tx and endorse
- Create rw-set
- Collect endorsements

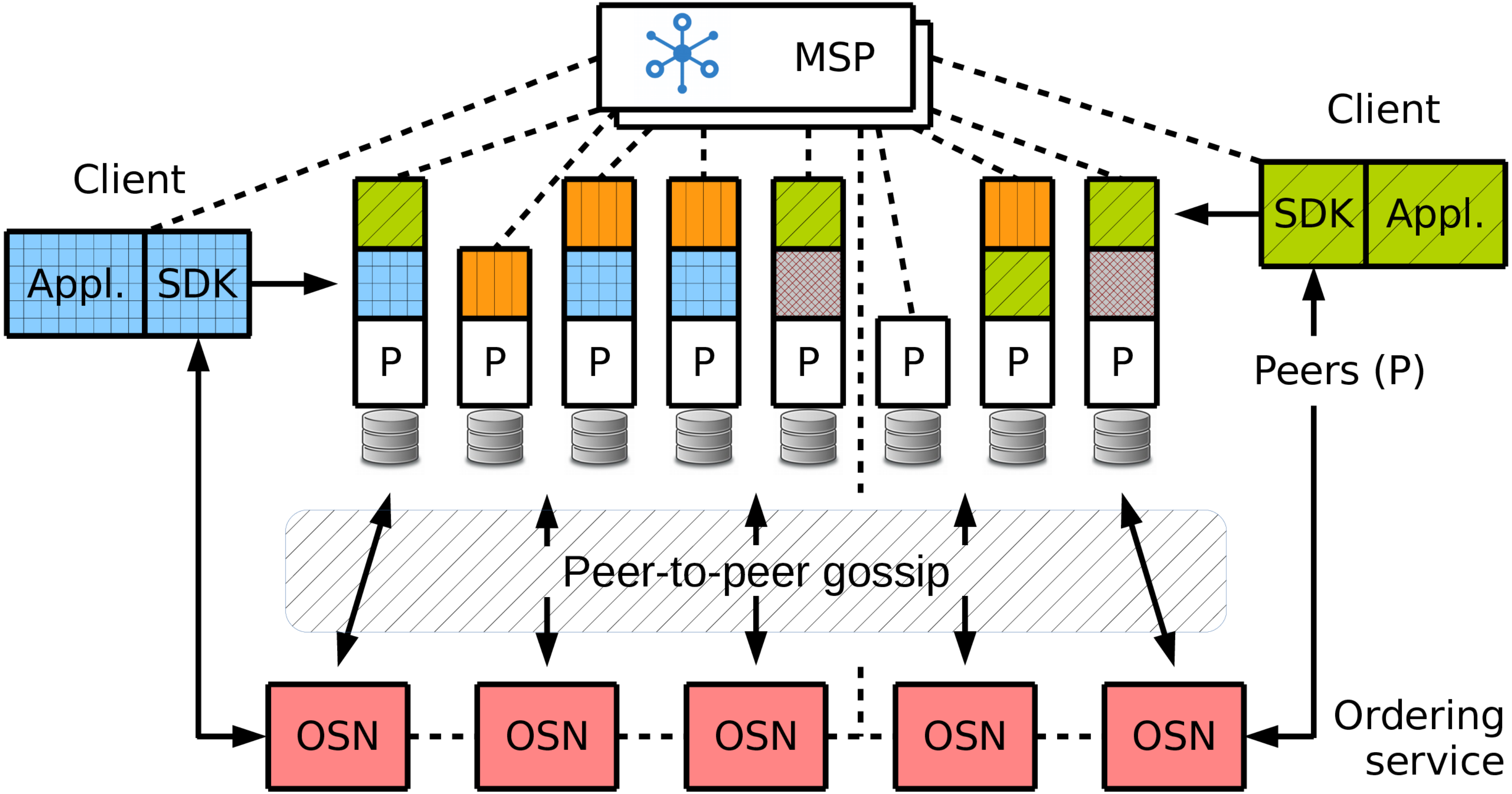
- Order rw-sets
- Atomic broadcast (consensus)
- Stateless ordering service

- Validate endorsements & rw-sets
- Eliminate invalid and conflicting tx

- Persist state on all peers

- Includes techniques from databases
- Extends a middleware-replicated database to BFT model

Fabric V1 – Separating endorsement and consensus



Fabric V1 details

- 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 [Kemme et al., 2010]
  - Replication via (BFT) atomic broadcast in consensus
  - Endorsement protects against unauthorized updates

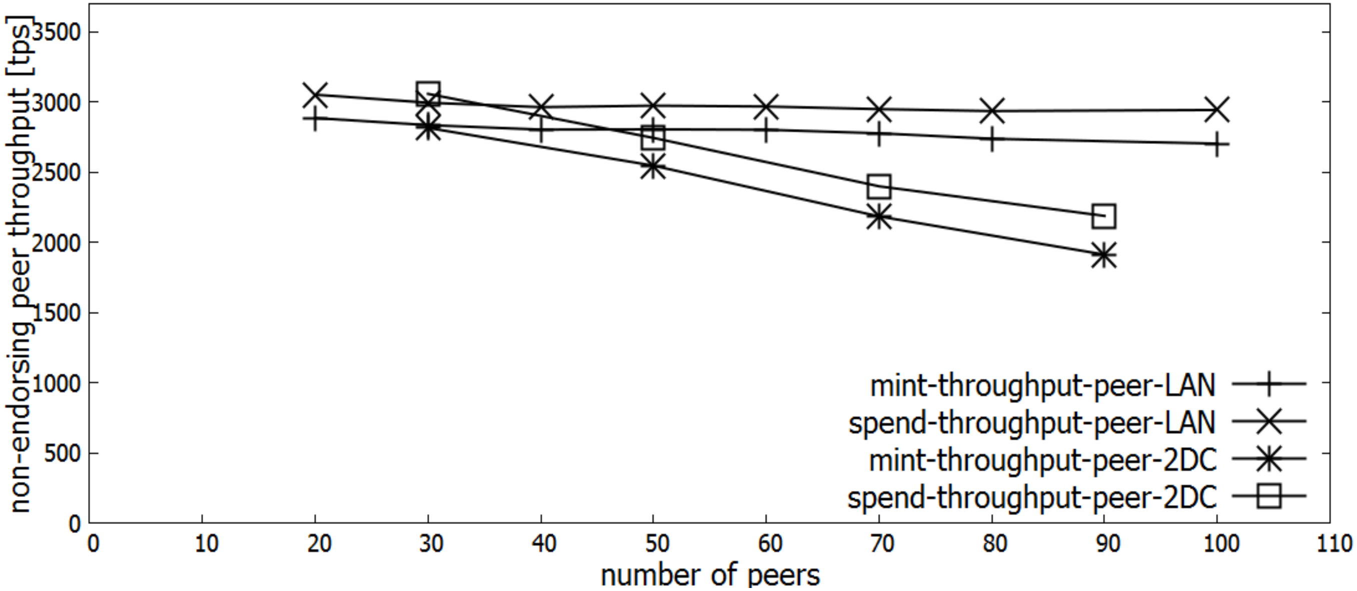
- Scales better – only few nodes execute, independent computations in parallel

- Permits some confidential data on blockchain via partitioning state
  - Data seen only by endorsers assigned to run that chaincode

Modular consensus in Fabric V1

- "Solo orderer"
  - One host only, for testing
- Apache Kafka, a distributed pub/sub streaming platform
  - Tolerates crashes among member nodes, resilience from Apache Zookeeper inside
  - Focus on high throughput
- BFT-SMaRt – Research prototype
  - Tolerates $F < N/3$ Byzantine faulty nodes among N
  - Demonstration of functionality [Sousa et al., A BFT Ordering Service for Hyperledger Fabric ..., DSN 2018]
- SBFT – Simple implementation of PBFT (currently under development)
  - Tolerates $F < N/3$ Byzantine faulty nodes among N
  - Focus on resilience

Fabric V1 – Performance of 'Fabric Coin'



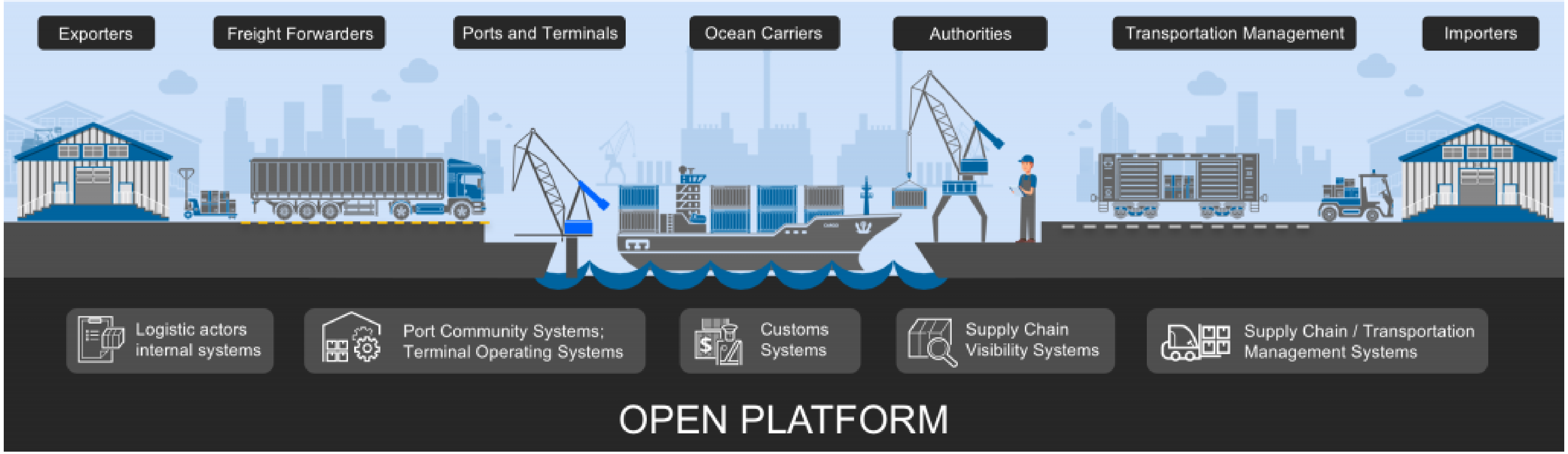
- Scalability with number of non-endorsing peers
- Bitcoin-like transactions (UTXO): mint and spend
- Cloud deployment on a LAN and in two data centers (2DC)

[Androulaki et al., Eurosys 2018, doi.org/10.1145/3190508.3190538]

Hyperledger Fabric deployment

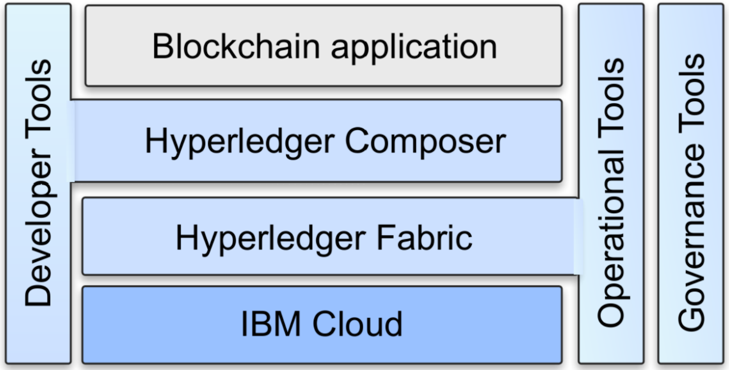
- Fabric is the most prominent and widely used blockchain platform for business
  - Cloud deployment (BaaS) by: IBM, Amazon, Azure, Oracle, Fujitsu, SAP ...
  - Hundreds of prototypes and in-production systems built by IBM alone

- At the core of many new businesses
  - Example: IBM-Maersk joint venture, building a blockchain platform for global trade


Exporters
Freight Forwarders
Ports and Terminals
Ocean Carriers
Authorities
Transportation Management
Importers
Logistic actors internal systems
Port Community Systems; Terminal Operating Systems
Customs Systems
Supply Chain Visibility Systems
Supply Chain / Transportation Management Systems
OPEN PLATFORM

IBM Blockchain Platform

- Fully integrated blockchain service platform
  - Developer tools like Hyperledger Composer
  - Hyperledger Fabric distributed ledger technology
  - Governance tools
  - Deployed on IBM Cloud environment



- Provides enterprise-grade security
  - Keys managed by hardware security modules (HSM), certified by NIST at highest level
  - Secure service container (SSC) technology, protecting code and data from admins (such as available with IBM LinuxONE)

Current research directions

- Private transactions in Fabric
  - Privacy-preserving state-based endorsement (Side DB)
  - Share data selectively with channel-private data, ledger stores only hashes

- Zero-knowledge proofs (ZKP)
  - Anonymous authentication with IBM Identity Mixer, anonymity with attribute-based access control
  - Zero-Knowledge Asset Transfer (ZKAT), for privacy-preserving exchange of assets

- Secure smart-contract execution with Intel SGX technology
  - Hardware-based secure enclaves
  - Data and application logic protected from malicious peers [Brandenburger et al., arxiv.org/abs/1805.08541]

Conclusion

Conclusion

- Blockchain = Distributing trust over the Internet

- Go beyond the hype and turn to established science and engineering

- Hyperledger Fabric is the most advanced enterprise blockchain platform
  - Driven by innovations from IBM Research

- Some links

  www.hyperledger.org
  www.ibm.com/blockchain/
  www.zurich.ibm.com/blockchain/
  ibm.ent.box.com/v/BlockFiles/
  cachin.com/cc

PRIV LEDGE

Privacy-Enhancing Cryptography in Distributed Ledgers (EU Horizon 2020; 2018-2020)

priviledge-project.eu

Hyperledger Fabric references

- www.hyperledger.org/projects/fabric
- Architecture paper – doi.org/10.1145/3190508.3190538
  ACM Eurosys 2018 conference
- Designs – wiki.hyperledger.org/projects/fabric/design-docs
- Docs – hyperledger-fabric.readthedocs.io/en/latest/
- Code – github.com/hyperledger/fabric
- Chat – chat.hyperledger.org, all channels like #fabric-*