BLOCKCHAIN & CONSENSUS

Ren Shan, MSCE, PMP

DEC, 2019

Linkgear Foundation © Copy Right 2019

About Me

- IT industry over 25 years
- IEEE Standard Association
  - Co-chair of SGs of WG – Clinic IoT Data with TIPPSS
  - Member of Blockchain WGs of Energy
- Leading Emerging & Innovative technologies in Federal practices

- Certified MS Azure Solution Expert
- HL Certified Architect
- PMP, CSM, MBA
- Living in DC area for 20+ years
- Raspberry Pi enthusiast

Ren Shan

Email: rens@linkgear.io

Blockchain The Trust & Time Machine

"A system in which a record of transactions made in bitcoin or another cryptocurrency are maintained across several computers that are linked in a peer-to-peer network."

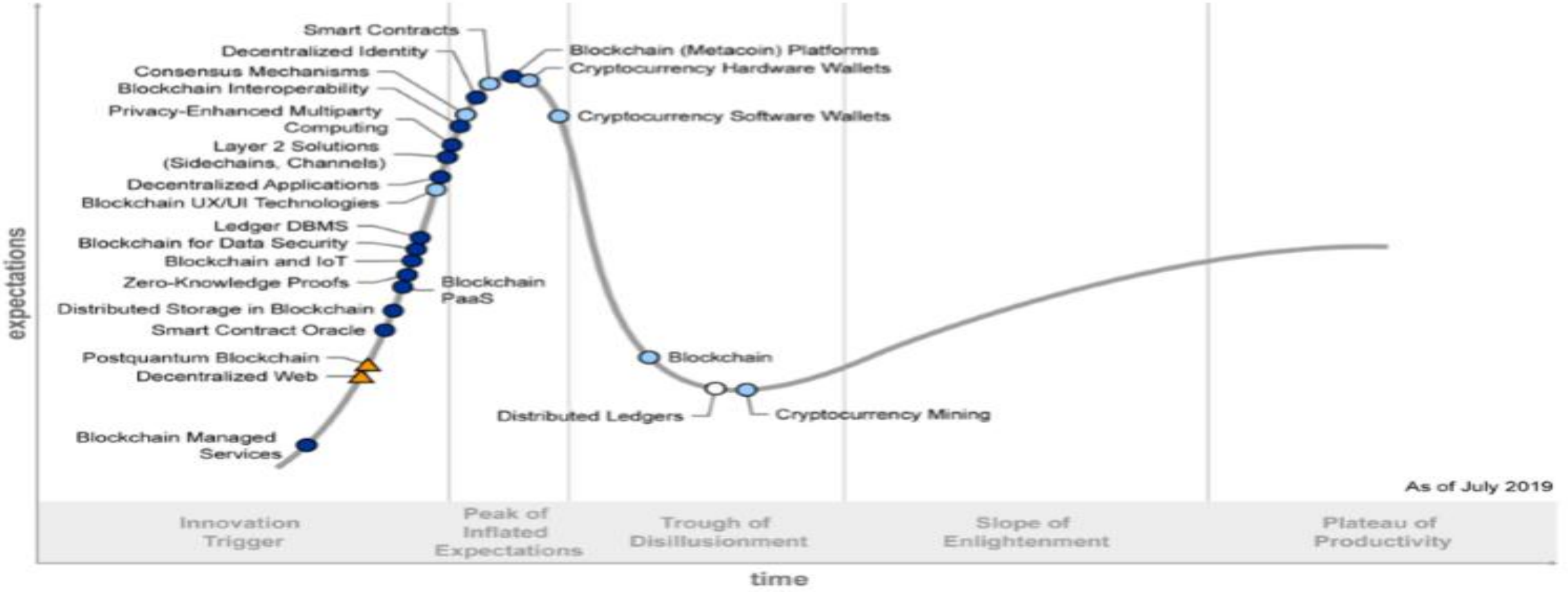
– Satoshi Nakamoto

- **Decentralized**
- **Consensus**
- **Public Ledger**

https://bitcoin.org/en/bitcoin-paper

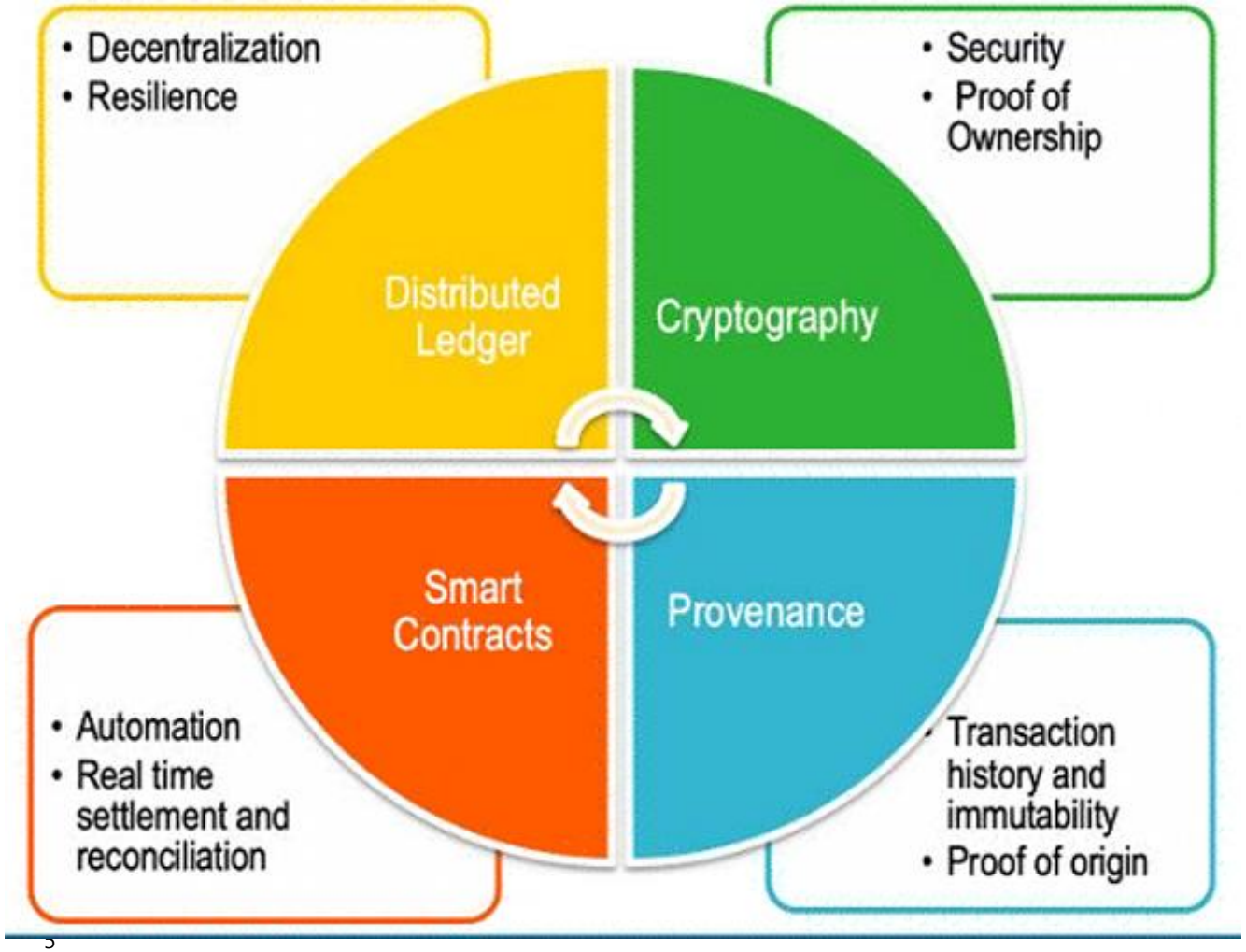
- Peer to Peer
- Cryptographic
- DAO (Decentralized Autonomous Org)
- Public & Private
- BC version 1.0, 2.0, 3.0
- Multi-ledger and Multi-cloud

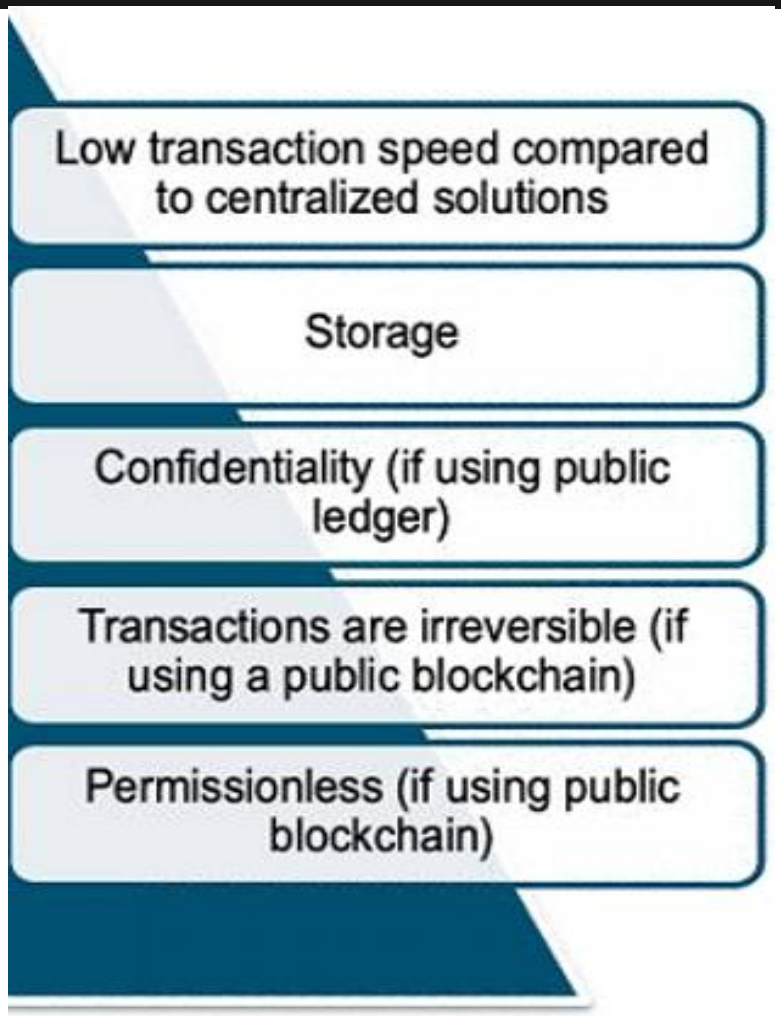
Hype Cycle for Blockchain Technologies, 2019



Source: Gartner
ID: 383155

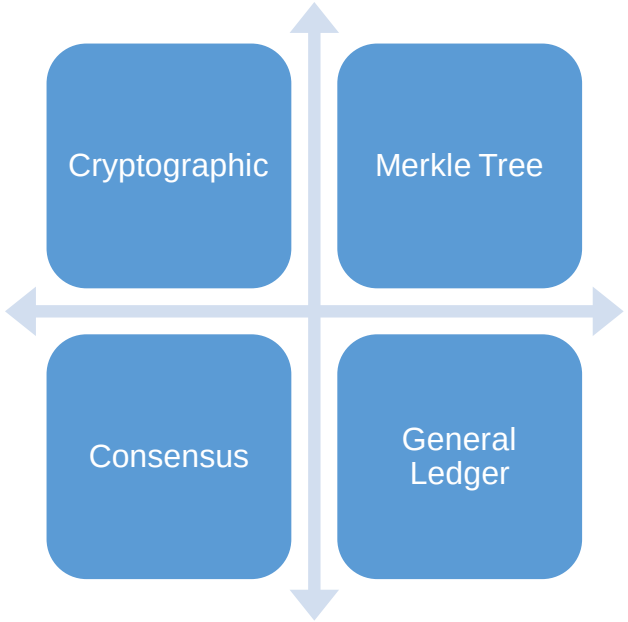
Blockchain Pros and Cons

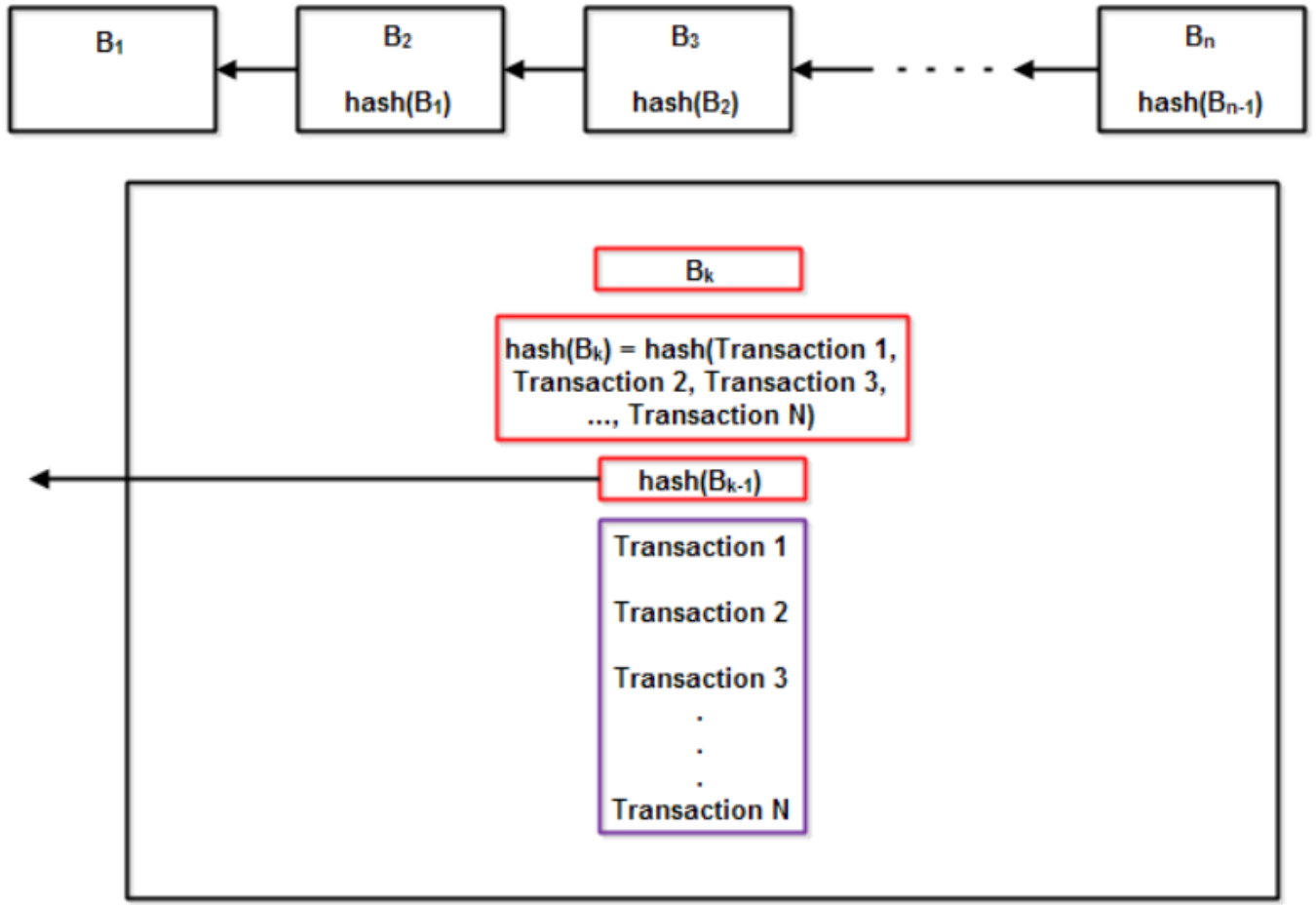




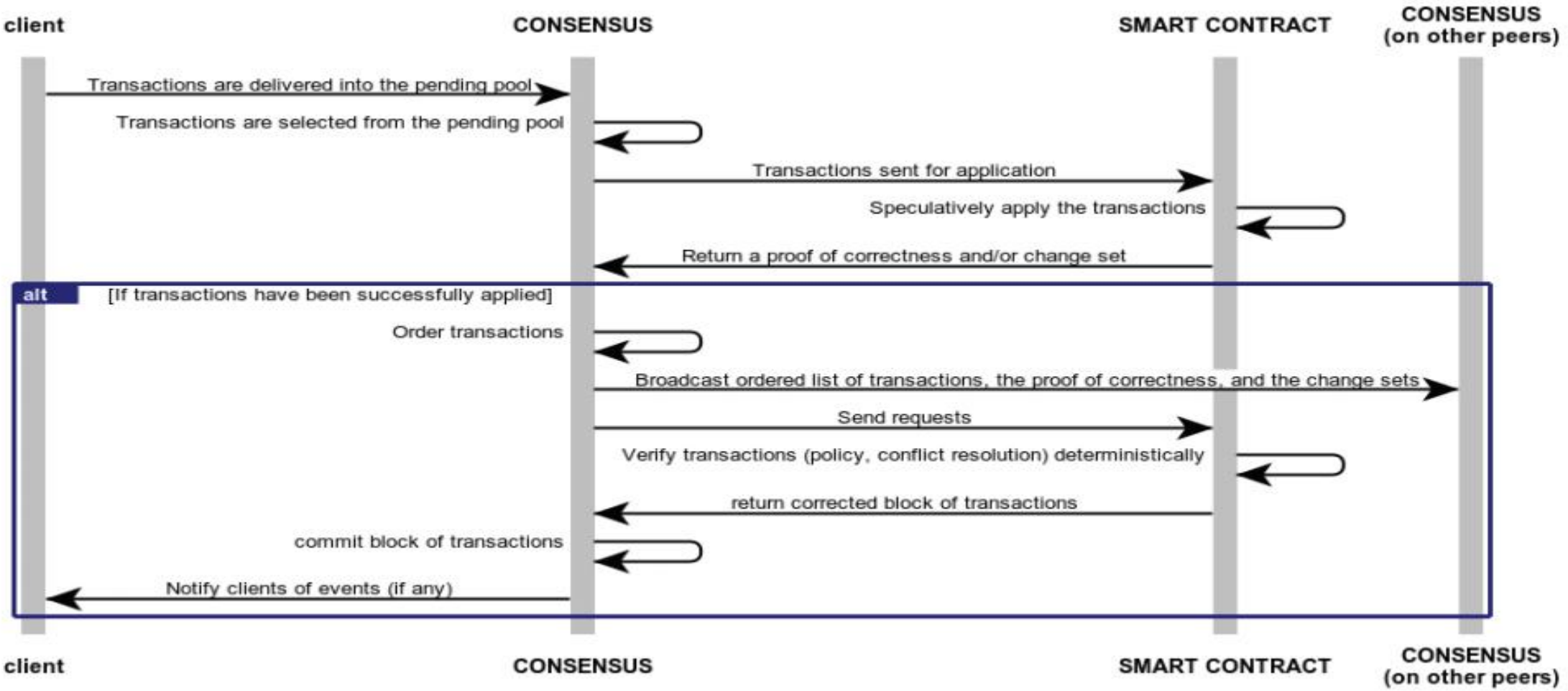
LINKGEAR FOUNDATION LLC

Blockchain How It Works



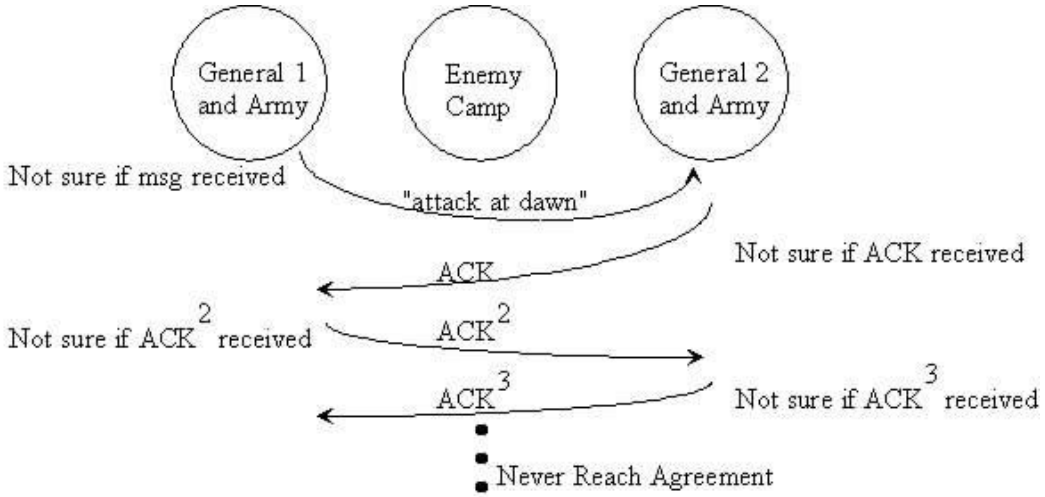


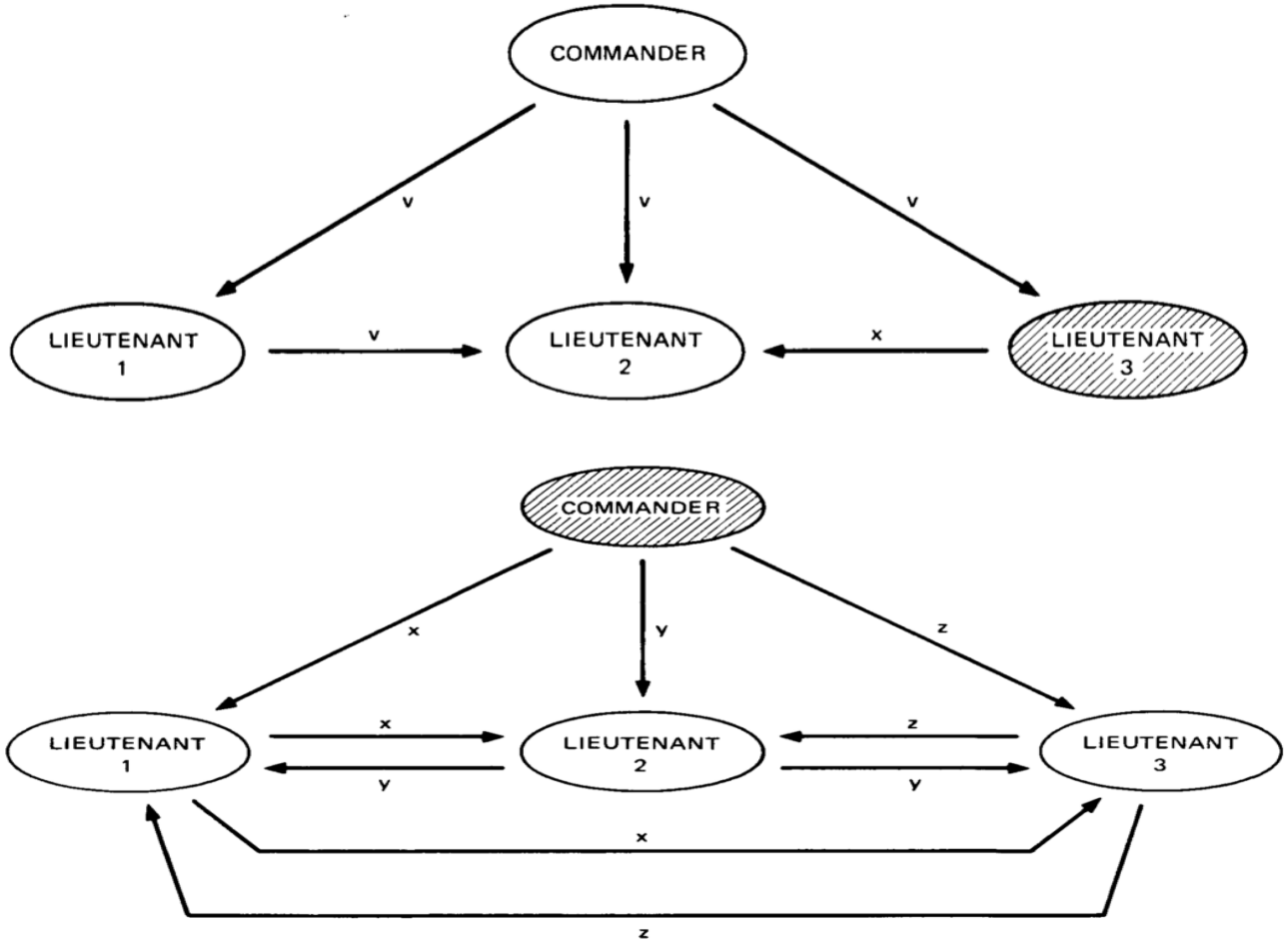
GENERALIZED CONSENSUS PROCESS FLOW STANDARD



www.websequencediagrams.com

Byzantine Fault Tolerance



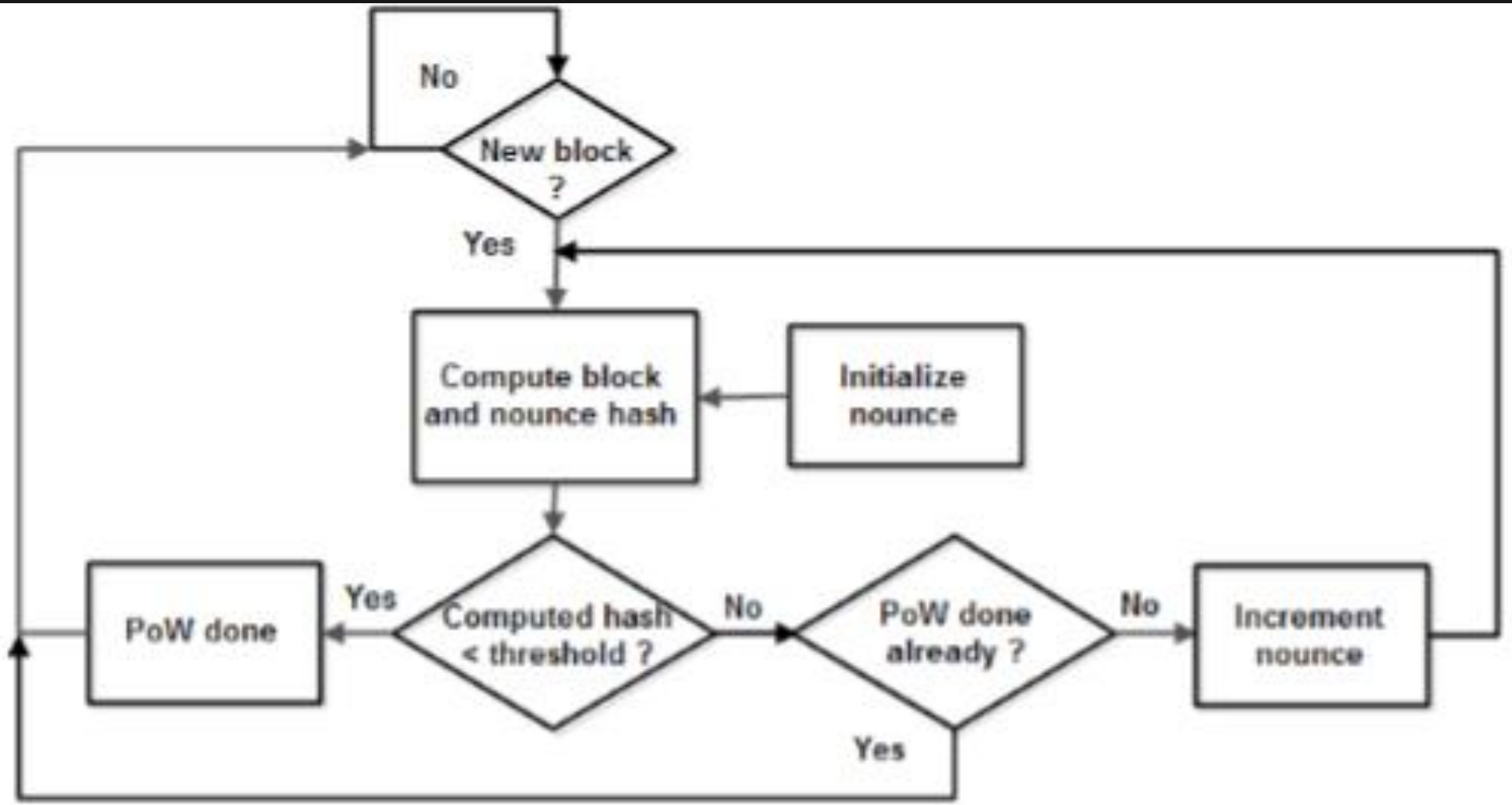


COMPARISON OF CONSENSUS ALGORITHM

Consensus Algorithm	Consensus Approach	Pros	Cons
Kafka/C(crash)BFT in Hyperledger Fabric Ordering Service	Permissioned voting based. **Leader does ordering**. Only in-sync replicas can be **voted as leader**. ("Kafka," 2017).	Provides crash fault tolerance. Finality happens in a matter of seconds.	While Kafka is crash fault tolerant, it is not Byzantine fault tolerant, which prevents the system from reaching agreement in the case of malicious or faulty nodes.
R(Redundant)BFT in Hyperledger Indy	Pluggable election strategy set to a permissioned, voting based strategy by default ("Plenum," 2016). All instances do ordering, but only the requests ordered by the **master instance** are actually executed. (Aublin, Mokhtar & Quéma, 2013)	Provides Byzantine fault tolerance. Finality happens in a matter of seconds.	The more nodes that exist on the network, the more time it takes to reach consensus. The nodes in the network are known and must be totally connected.
Sumeragi in Hyperledger Iroha	Permissioned server **reputation** system.	Provides Byzantine fault tolerance. Finality happens in a matter of seconds. Scale to petabytes of data, distributed across many clusters (Struckhoff, 2016).	The more nodes that exist on the network, the more time it takes to reach consensus. The nodes in the network are known and must be totally connected.
PoET(for instance, Hyperledger Sawtooth)	Pluggable election strategy set to a permissioned, lottery based strategy by default, in Trusted Excution Environment (TEE) environment	Provides scalability and Byzantine fault tolerance.	Finality can be delayed due to forks that must be resolved.
PoW (Proof of Work)	Public, mining and 51% (majority)	Provides scalability and Byzantine fault tolerance.	Finality happens in a longer time and based on computing power. Requires energy and storage.
PoS/DPoS (Proof of Stake)	Public, no mining needed based on Stake	Provides limited scalability and Byzantine fault tolerance.	Finality happens in a shorter time. Requires stakes/different types of power.

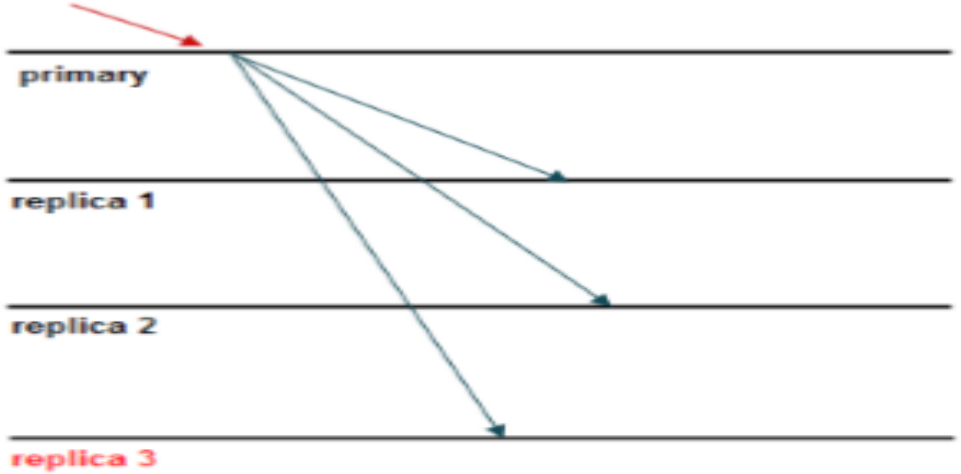
	PoW	PoS	PoET	BFT and variants	Federated BFT
Blockchain type	Permissionless	Both	Both	Permissioned	Permissionless
Transaction finality	Probabilistic	Probabilistic	Probabilistic	Immediate	Immediate
Transaction rate	Low	High	Medium	High	High
Token needed?	Yes	Yes	No	No	No
Cost of participation	Yes	Yes	No	No	No
Scalability of peer network	High	High	High	Low	High
Trust model	Untrusted	Untrusted	Untrusted	Semi-trusted	Semi-trusted
Adversary Tolerance	<=25%	Depends on specific algorithm used	Unknown	<=33%	<=33%

Consensus - POW



primary node and 3f replicas – 3f+1 nodes
maximum f nodes are faulty

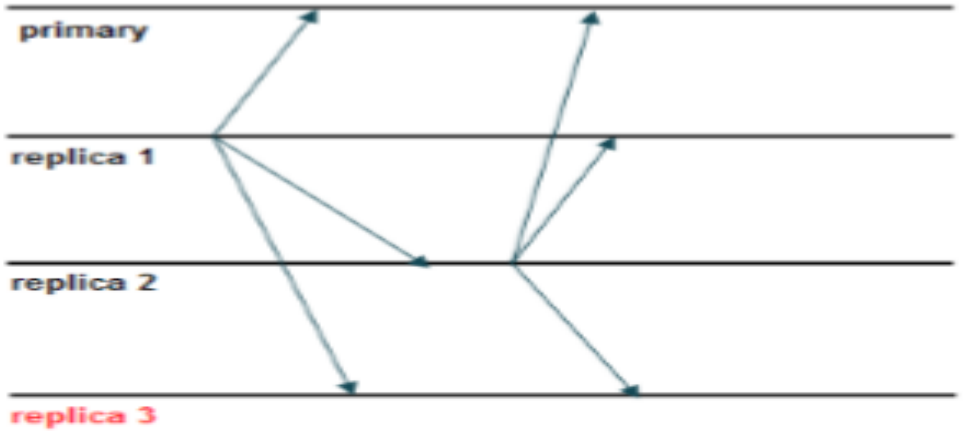
REQUEST f=1 replica 3 - faulty node



Pre-prepare: 1 message (multicast)

replica 3 does not transmit messages

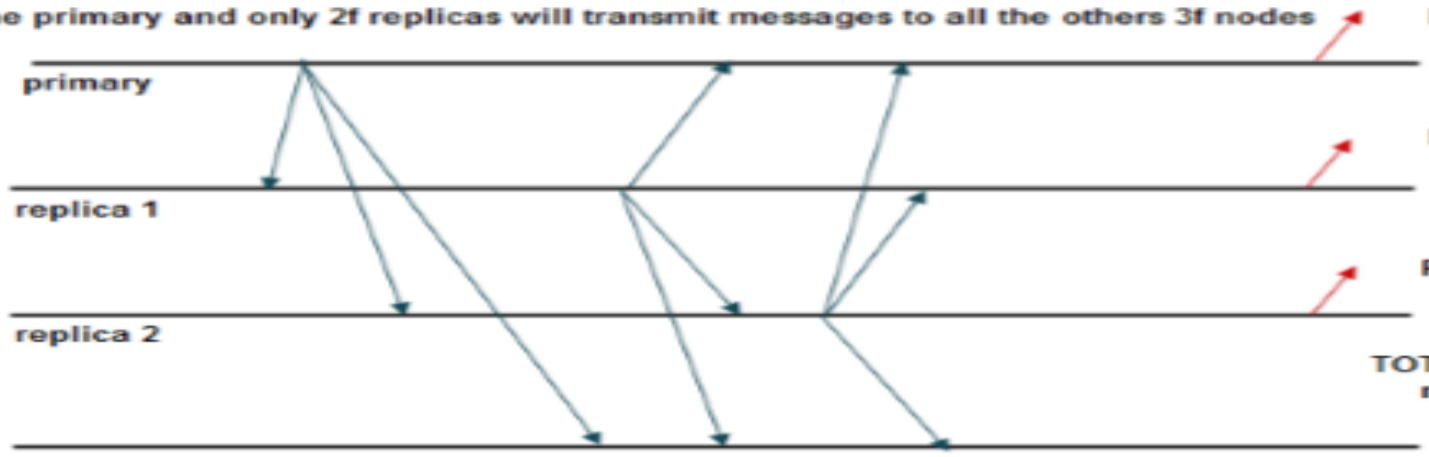
only 2f replicas will transmit messages to all the others 3f nodes



Prepare: 2f messages (multicast)

replica 3 does not transmit messages

the primary and only 2f replicas will transmit messages to all the others 3f nodes

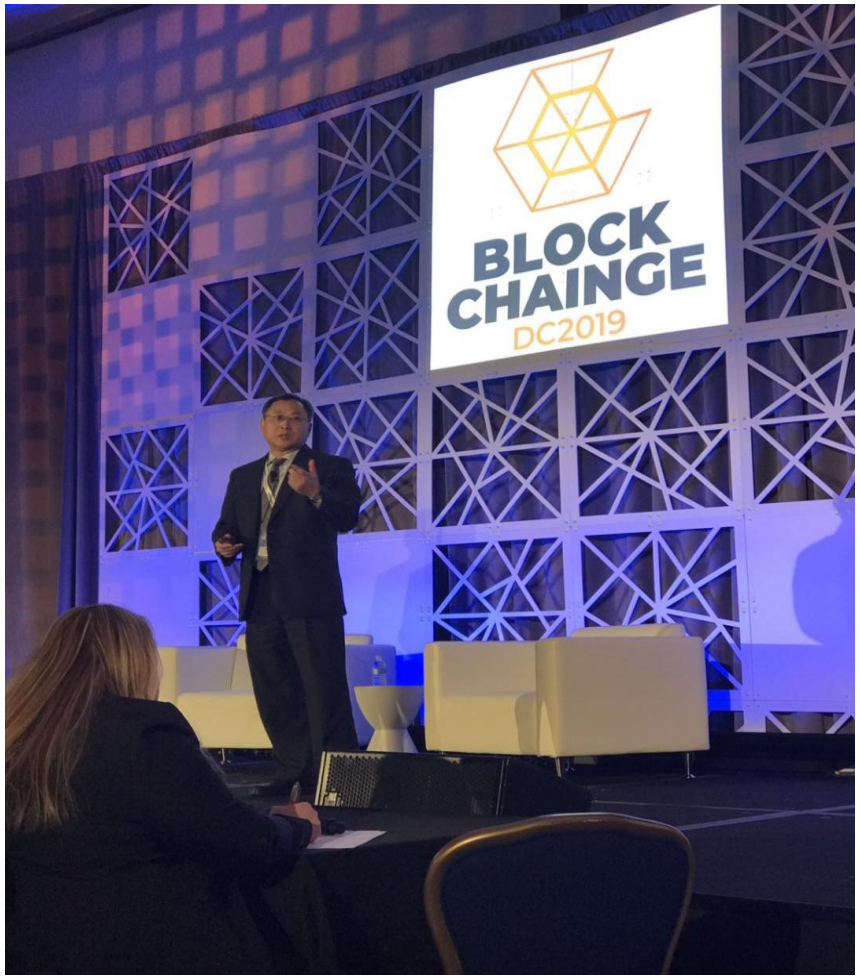


TOTAL = 6f + 1 messages

Commit: (2f+1) messages (multicast)

The primary and the 2f non faulty replicas replay to the client (2f+1 messages)

LINKGEAR: winners of BCDC 2019



GET IN TOUCH

Official Web Site:	http://linkgear.org
Email:	rens@linkgear.io
GitHub:	https://github.com/linkcryptocoin
Facebook:	https://www.facebook.com/linkgear.foundation/
LinkedIn:	https://www.linkedin.com/company/linkgear
Telegram:	https://t.me/LinkGear
Crunchbase:	https://www.crunchbase.com/organization/linkgear-foundation
Gust:	https://gust.com/companies/linkgear-foundation-llc/dashboard
Angel:	https://angel.co/linkgear-foundation

Class Offering

- Python (10 weeks)
- Starting in Dec 15 2019