Beyond Consensus in Permissioned Ledgers: Experiences in using BFT replication on DLTs

Alysson Bessani

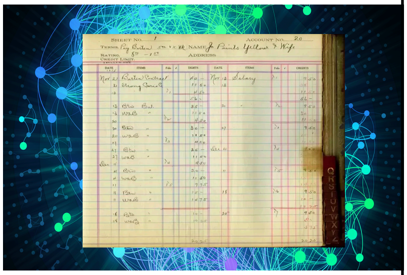
Associate Professor

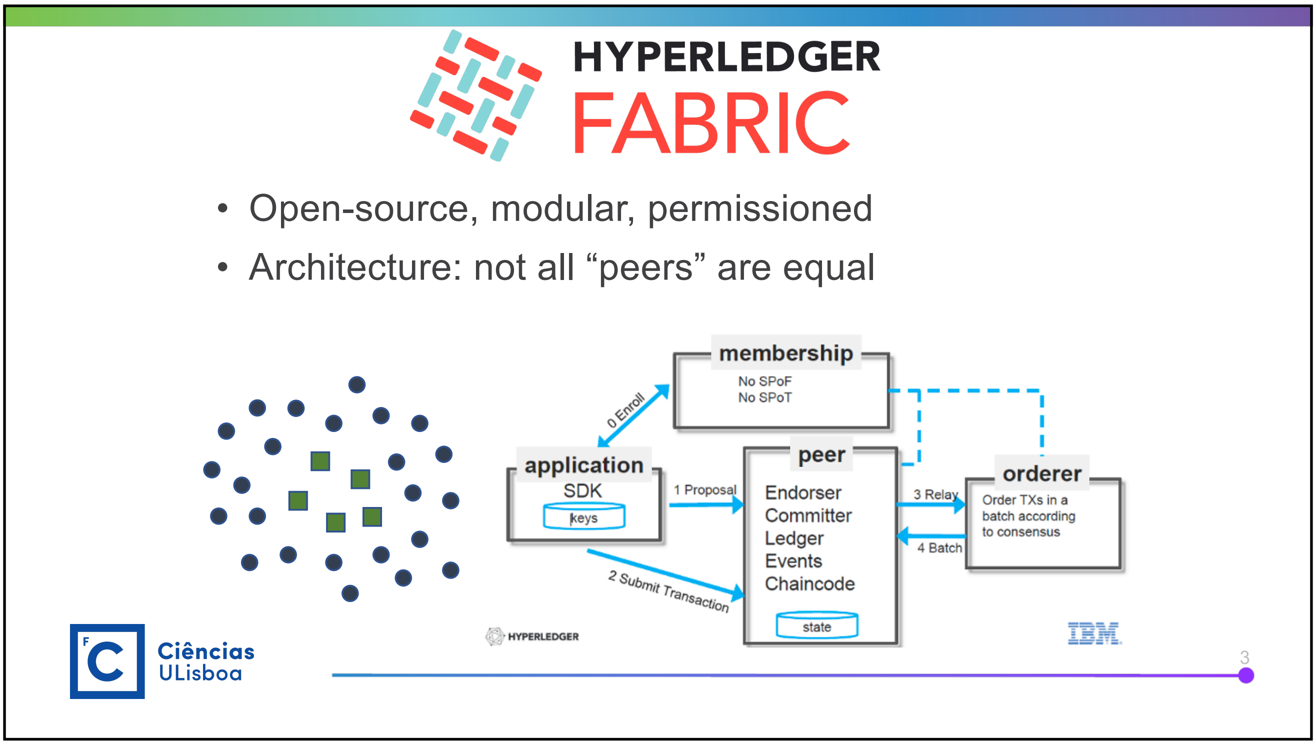
Ciências ULisboa

LASIGE

A view of permissioned blockchains

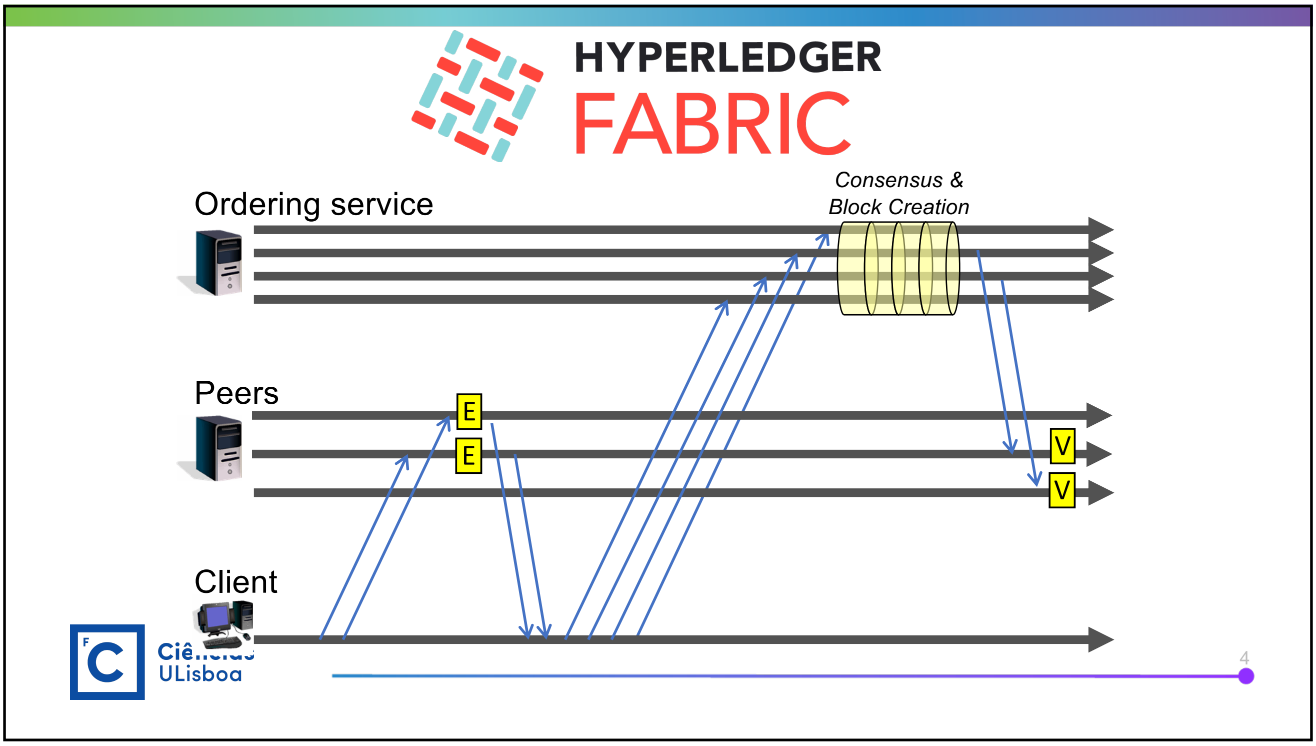
- **Decentralized trusted networked services**
  - Blockchains are instances of that…
- Distributed trust on the Internet (Cachin'01)
  - Systems that don't trust any single entity
- **Intrusion-tolerant systems** (Fraga & Powell'85)
- Requires **Byzantine Fault-Tolerant (BFT) consensus**





HYPERLEDGER FABRIC

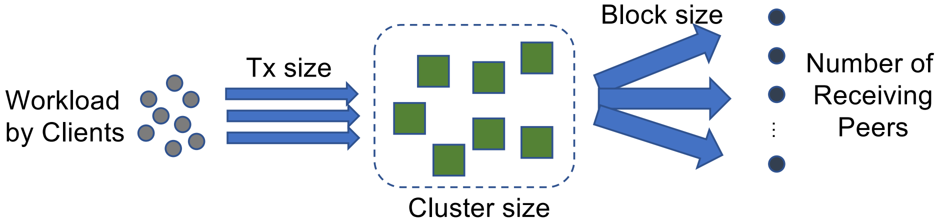
- Open-source, modular, permissioned
- Architecture: not all "peers" are equal



HYPERLEDGER FABRIC

HYPERLEDGER FABRIC

Ordering Service

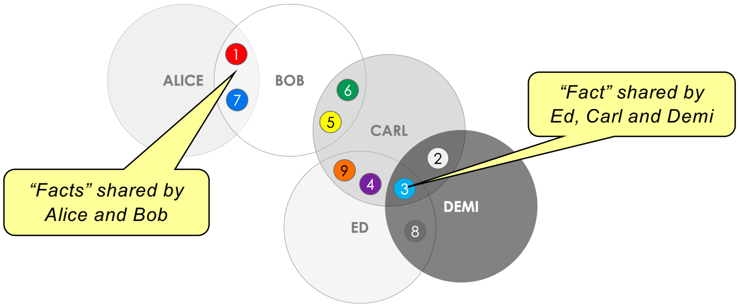


- Ordering node state:
  - the ordered transactions not yet in a block,
  - header of the last generated block, and
  - latest configuration block

Ciências ULisboa

corda

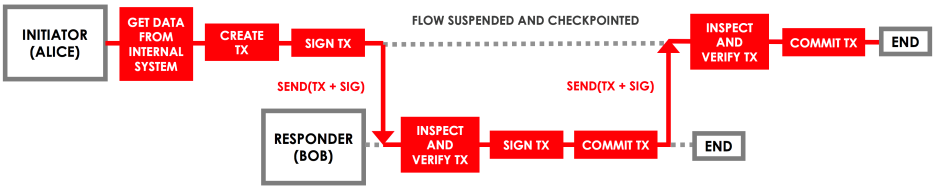
- Open-source blockchain project targeting (at least initially) the financial market
- Key idea: **there is no shared global ledger**
  - Instead, **there are many distributed ledgers**


ALICE
BOB
CARL
ED
DEMI
1
7
6
5
2
9
4
3
8
"Facts" shared by Alice and Bob
"Fact" shared by Ed, Carl and Demi

Ciências ULisboa

corda

- Only participants of a transaction have to *execute* and *validate* it
- A transaction is *committed* only if it achieve
  - **Validity consensus**: all involved participants need to validate and sign the transaction
  - **Uniqueness consensus**: requires a notary service

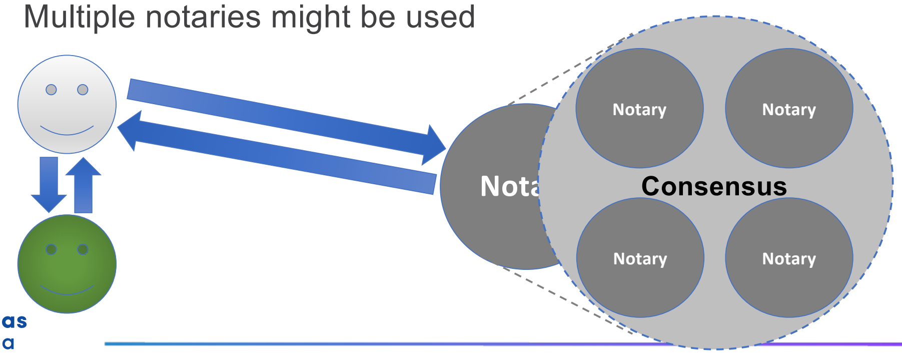


Ciências ULisboa

7

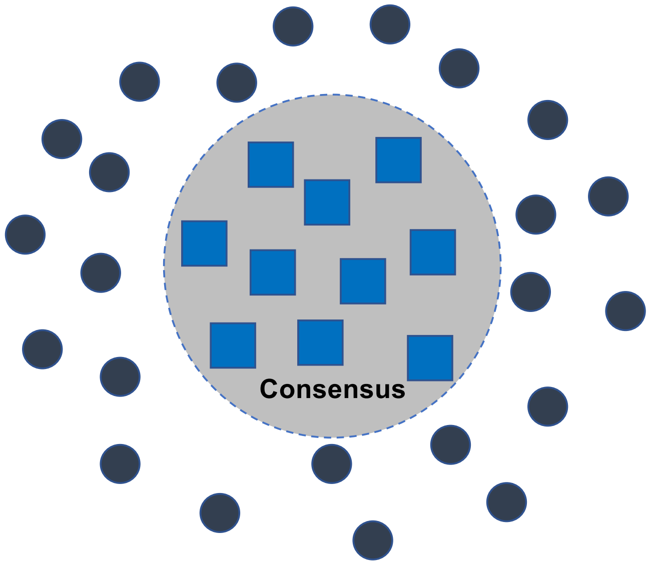
corda

- Notary implements an key-value store that register all state “consumptions”
- Some specific transaction validation might be executed
- Multiple notaries might be used



Ciências ULisboa

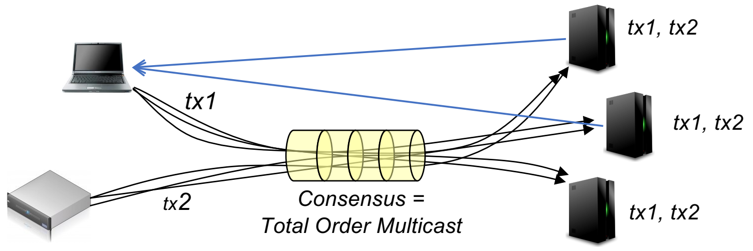
8



■ Validator

● Client

State Machine Replication

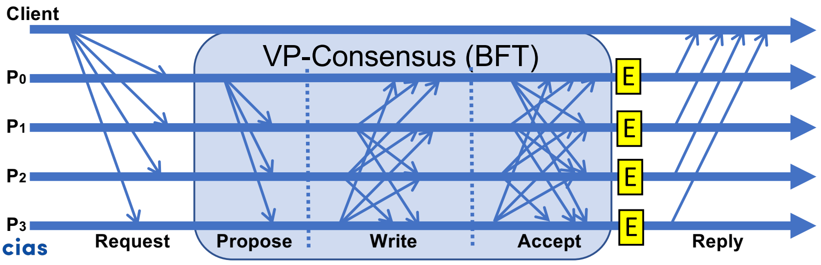


Safety: all replicas execute the same sequence of transactions

Liveness: transactions issued by correct clients are answered

BFT-SMaRt [DSN’14] (http://bft-smart.github.io/library/)

- State machine replication middleware written in Java (“seriously” developed and maintained since 2010)
- Can be configured to tolerate only crashes
- Available under Apache license
- Similar to PBFT in normal case, but it isn’t PBFT

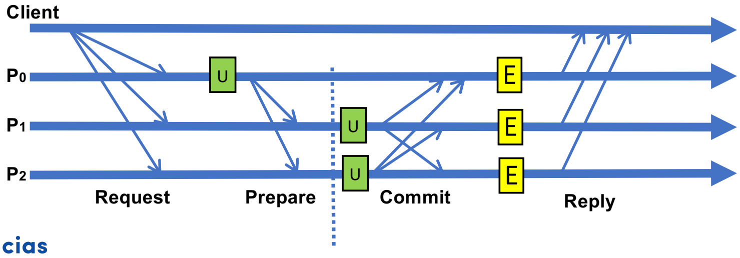


3f+1 replicas
O(n^2) messages
5 comm. steps

Ciências ULisboa

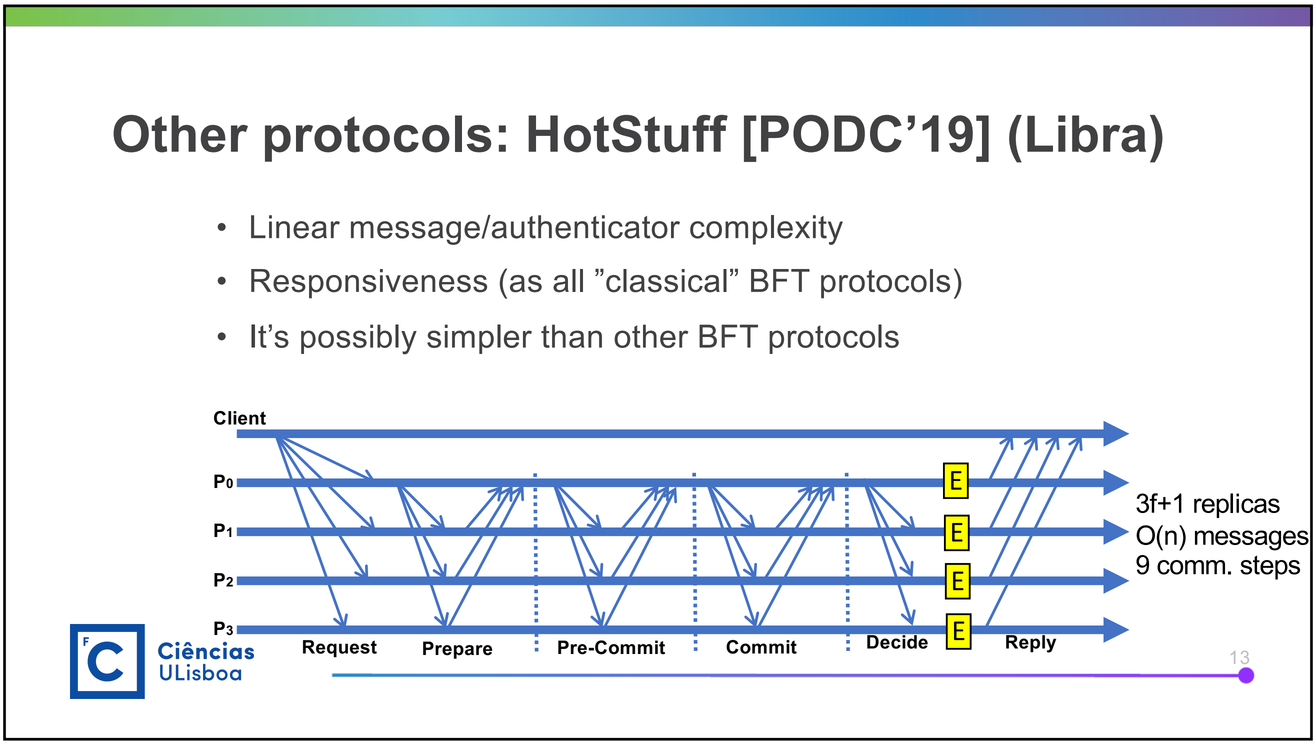
Other protocols: MinBFT [IEEE TC’13] (https://github.com/hyperledger-labs/minbft)

- Leverages trusted computing to constraint adversarial behaviour (i.e., requires TPM or SGX)
- Requires the same number of replicas, comm. steps and message complexity than crash protocols (e.g., Paxos, Raft)



2f+1 replicas
O(n^2) messages
4 comm. steps
Trusted signed counter

Ciências ULisboa



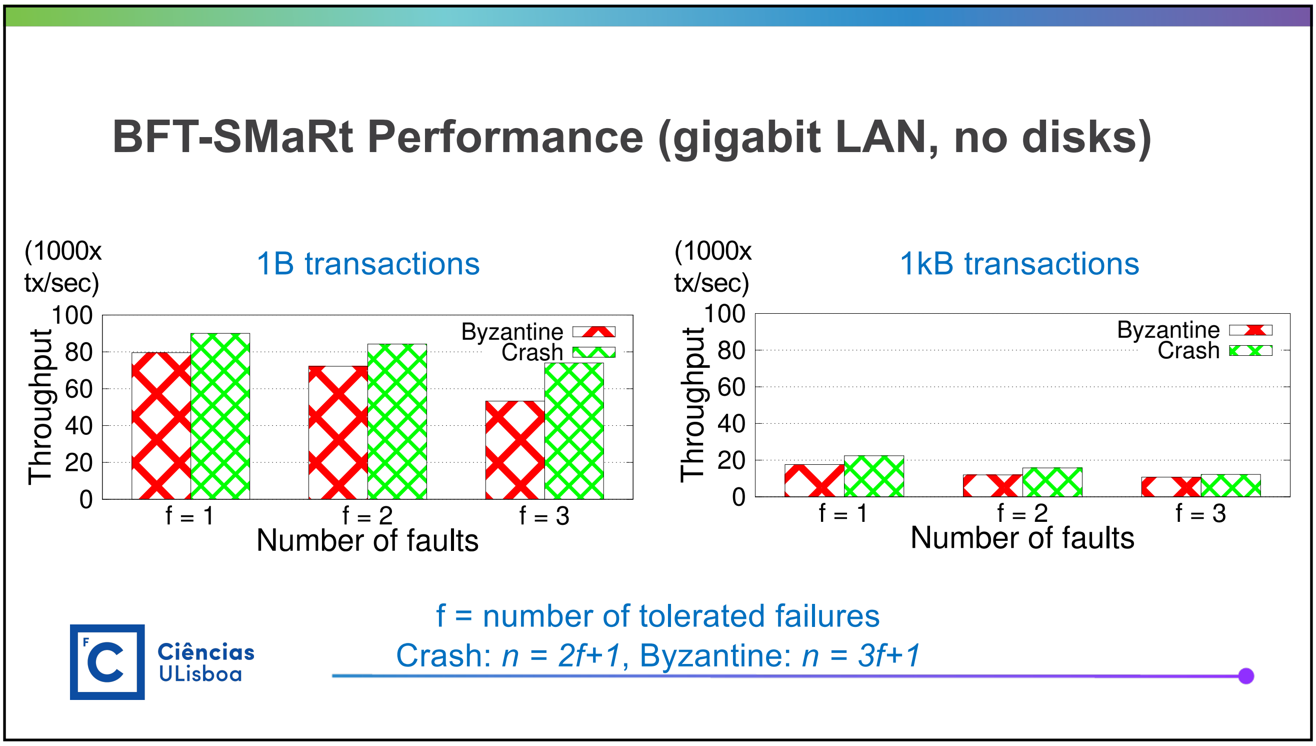
Other protocols: HotStuff [PODC'19] (Libra)

- Linear message/authenticator complexity
- Responsiveness (as all ”classical” BFT protocols)
- It's possibly simpler than other BFT protocols

3f+1 replicas
O(n) messages
9 comm. steps

Ciências ULisboa

13



BFT-SMaRt Performance (gigabit LAN, no disks)

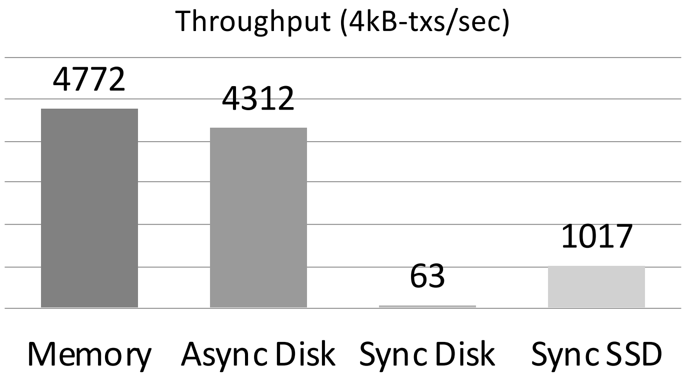
f = number of tolerated failures
Crash: *n = 2f+1*, Byzantine: *n = 3f+1*

Ciências ULisboa

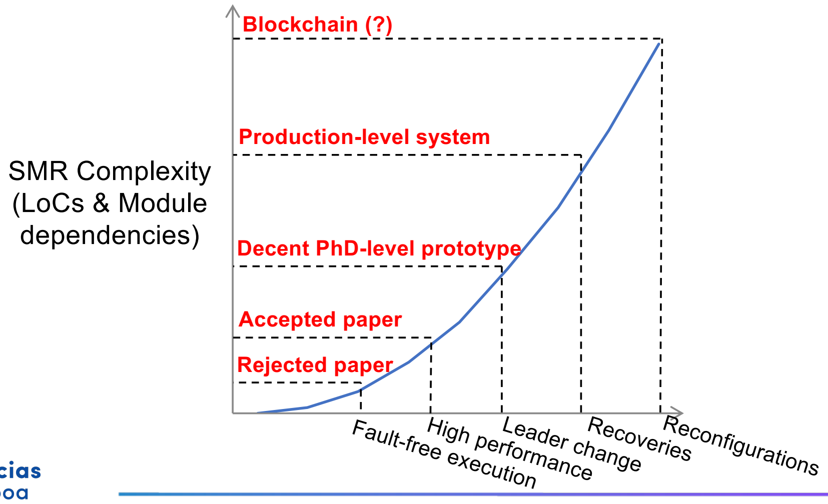
Consensus is not enough

- A consensus engine also needs:
  - **Durability**: any request completed at a client is reflected in the service after a recovery (more than *f* replicas can be faulty, but not Byzantine)
  - **Crash recovery**: recovered replicas need to be synched
  - **Reconfiguration**: replica group changes

Durability = Stable Logging



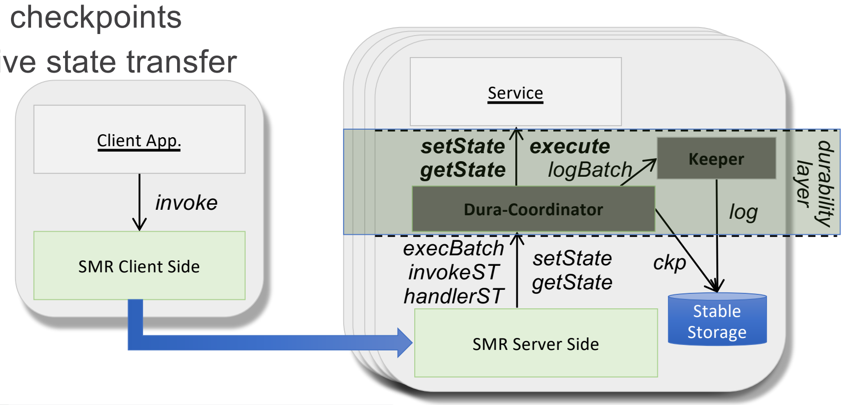
More features = More Complexity



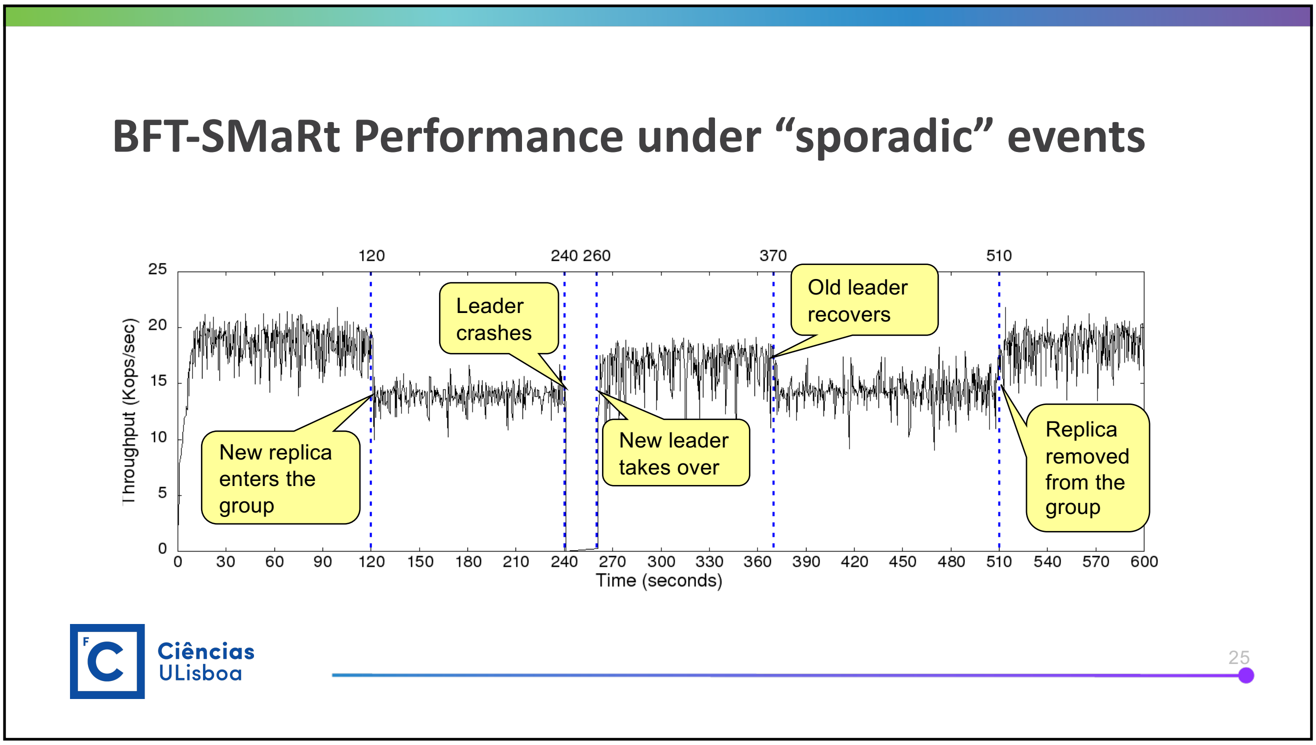
Ciências ULisboa

BFT-SMaRt

- Techniques for efficient durability
  - Parallel Logging
  - Sequential checkpoints
  - Collaborative state transfer



Ciências ULisboa



BFT-SMaRt Performance under “sporadic” events

New replica enters the group

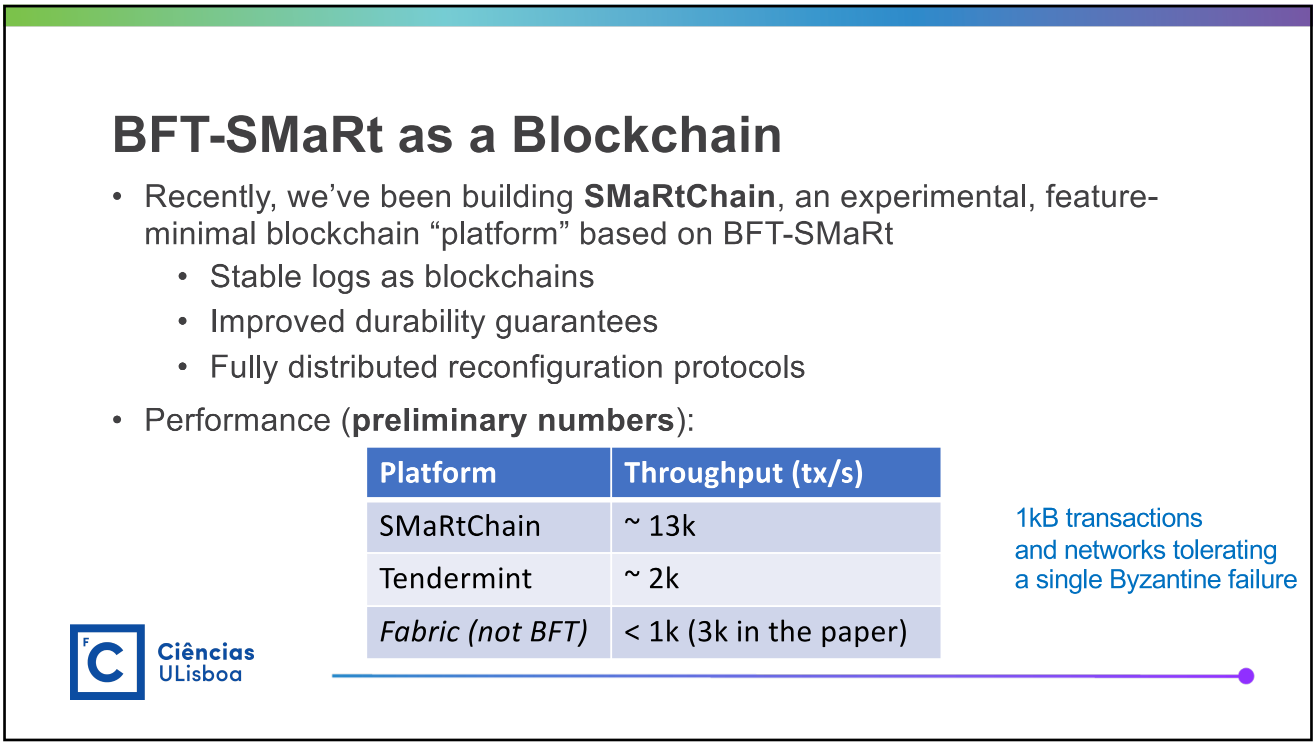
Leader crashes

New leader takes over

Old leader recovers

Replica removed from the group

25



BFT-SMaRt as a Blockchain

- Recently, we’ve been building **SMaRtChain**, an experimental, feature-minimal blockchain “platform” based on BFT-SMaRt
  - Stable logs as blockchains
  - Improved durability guarantees
  - Fully distributed reconfiguration protocols
- Performance (**preliminary numbers**):

Platform	Throughput (tx/s)
SMaRtChain	~ 13k
Tendermint	~ 2k
Fabric (not BFT)	< 1k (3k in the paper)

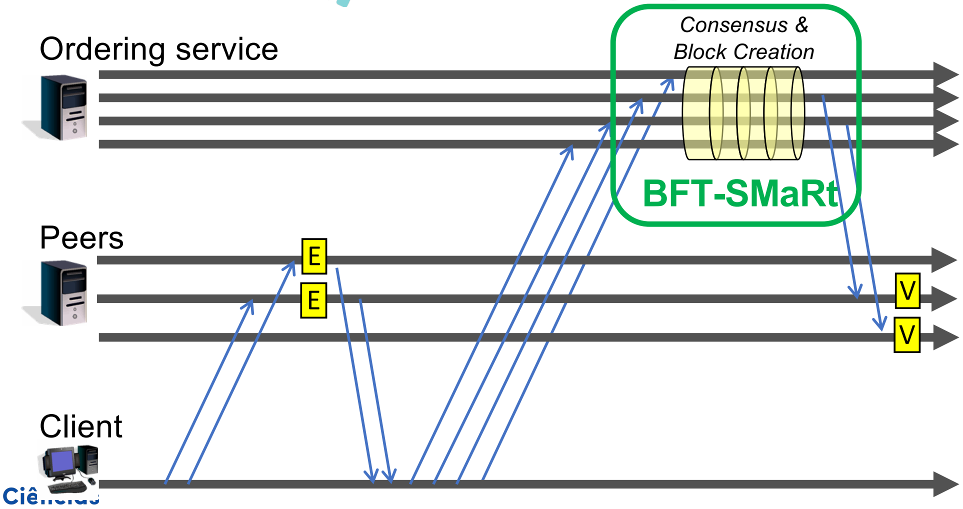
1kB transactions and networks tolerating a single Byzantine failure

BFT-SMaRt on other Blockchains

- Symbiont Assembly (rewrote BFT-SMaRt in Go)
- Experimental Corda BFT notary
- BFT orderer for Hyperledger Fabric [DSN'18]

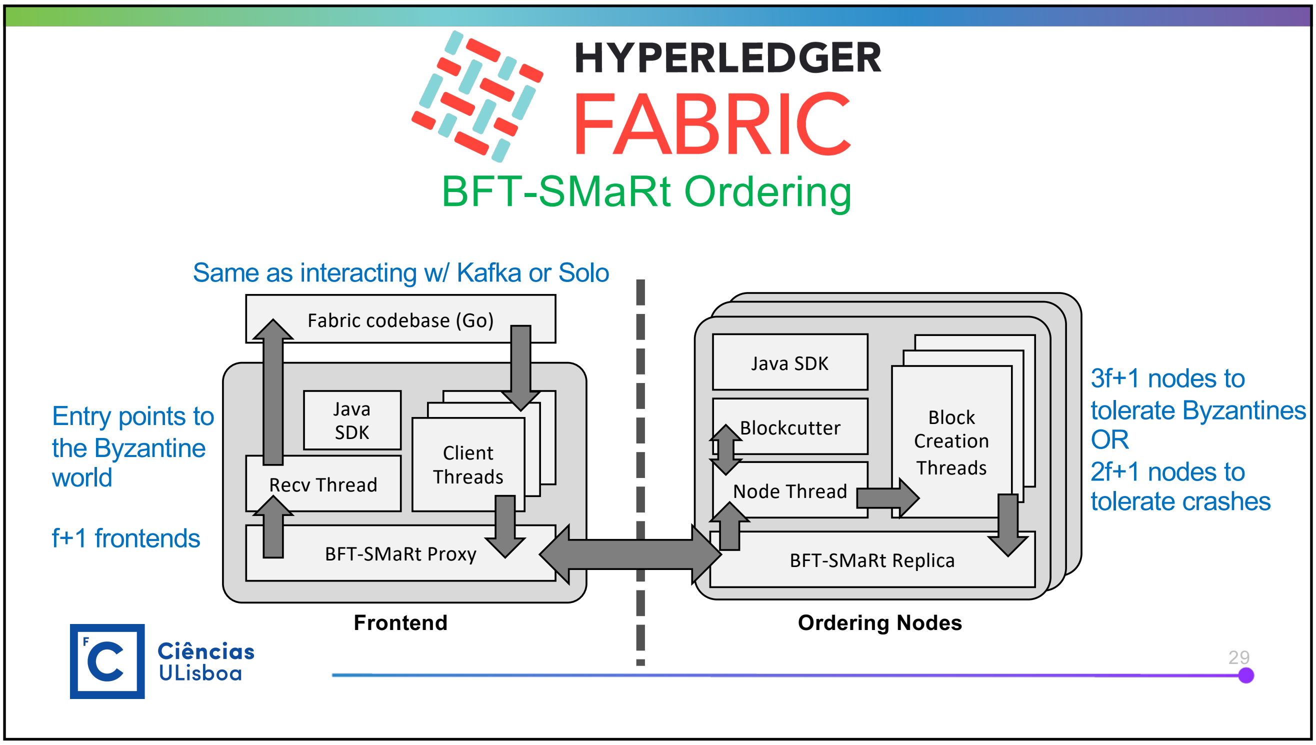
Ciências ULisboa

HYPERLEDGER FABRIC



Ciências ULisboa

28



HYPERLEDGER FABRIC

BFT-SMaRt Ordering

Same as interacting w/ Kafka or Solo

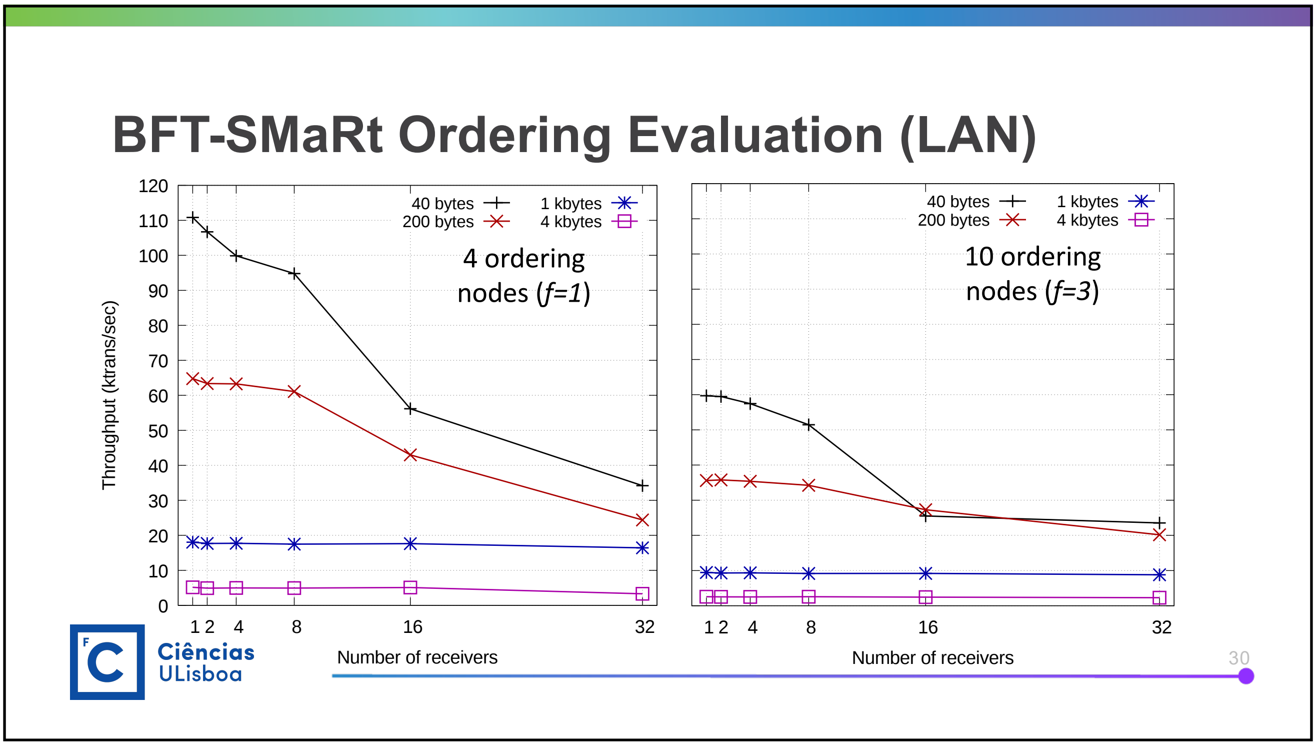
Entry points to the Byzantine world

f+1 frontends

3f+1 nodes to tolerate Byzantines OR 2f+1 nodes to tolerate crashes

Ciências ULisboa

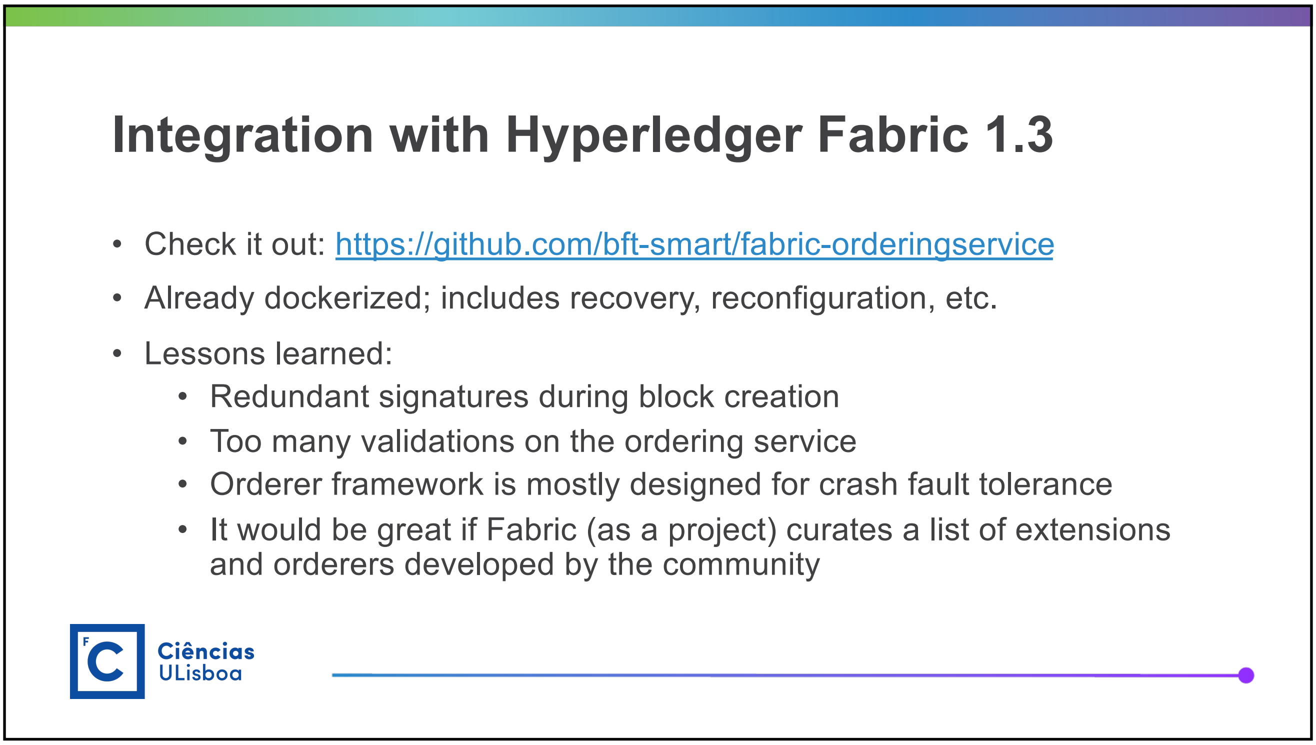
29



BFT-SMaRt Ordering Evaluation (LAN)

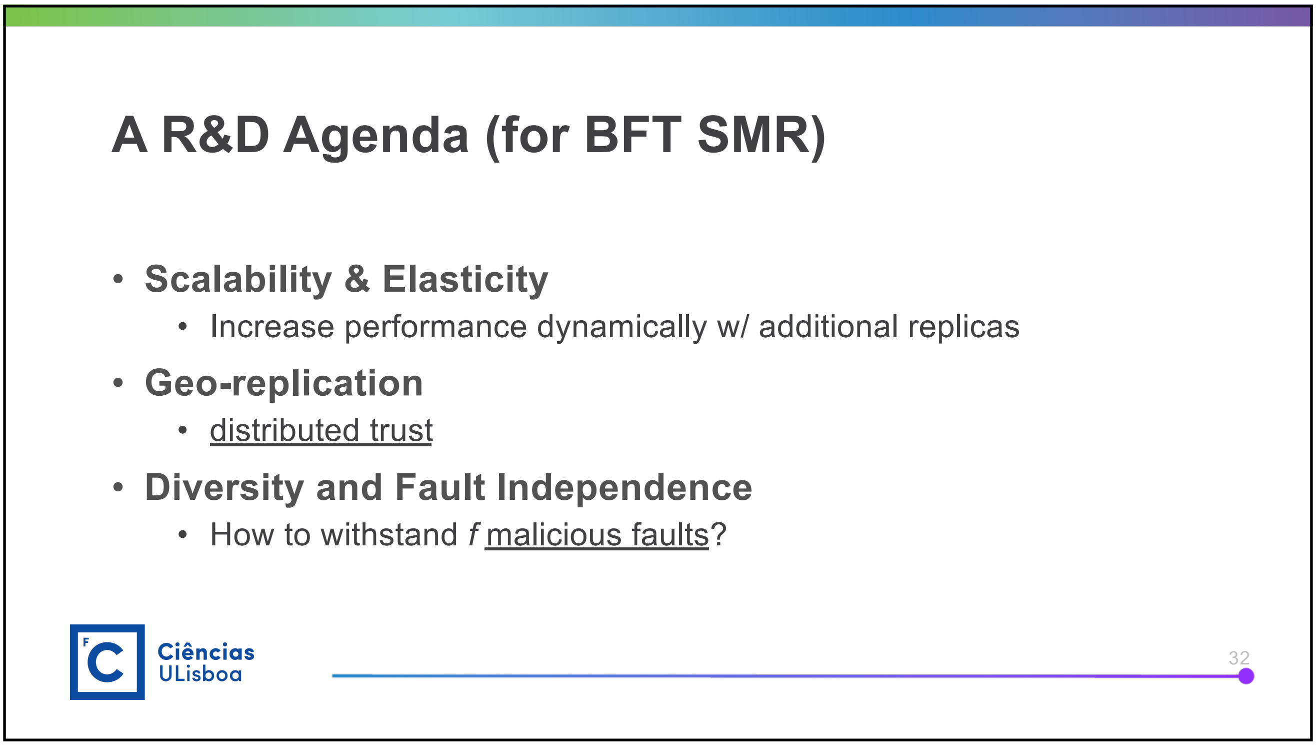
Ciências ULisboa

30



Integration with Hyperledger Fabric 1.3

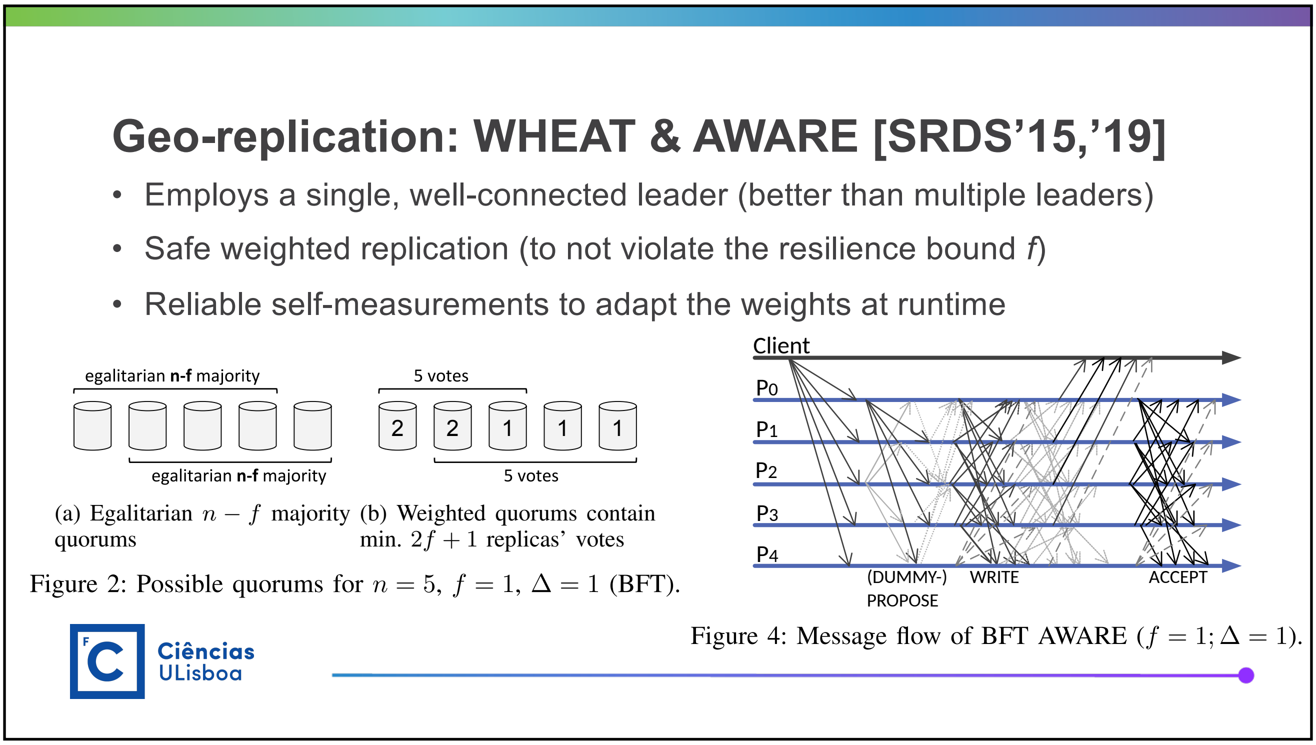
- Check it out: https://github.com/bft-smart/fabric-orderingservice
- Already dockerized; includes recovery, reconfiguration, etc.
- Lessons learned:
  - Redundant signatures during block creation
  - Too many validations on the ordering service
  - Orderer framework is mostly designed for crash fault tolerance
  - It would be great if Fabric (as a project) curates a list of extensions and orderers developed by the community



A R&D Agenda (for BFT SMR)

- **Scalability & Elasticity**
  - Increase performance dynamically w/ additional replicas
- **Geo-replication**
  - distributed trust
- **Diversity and Fault Independence**
  - How to withstand *f* malicious faults?

32



Geo-replication: WHEAT & AWARE [SRDS'15,'19]

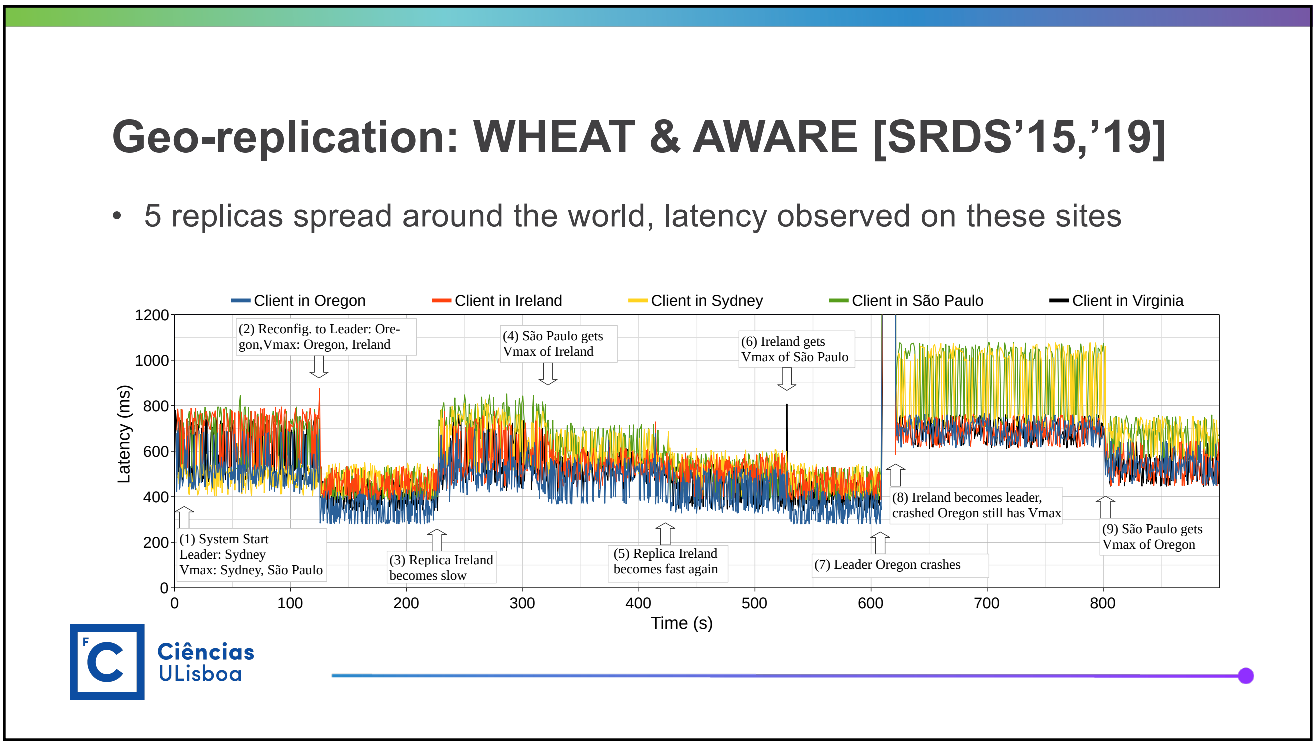
- Employs a single, well-connected leader (better than multiple leaders)
- Safe weighted replication (to not violate the resilience bound *f*)
- Reliable self-measurements to adapt the weights at runtime

(a) Egalitarian $n - f$ majority quorums

(b) Weighted quorums contain min. $2f + 1$ replicas' votes

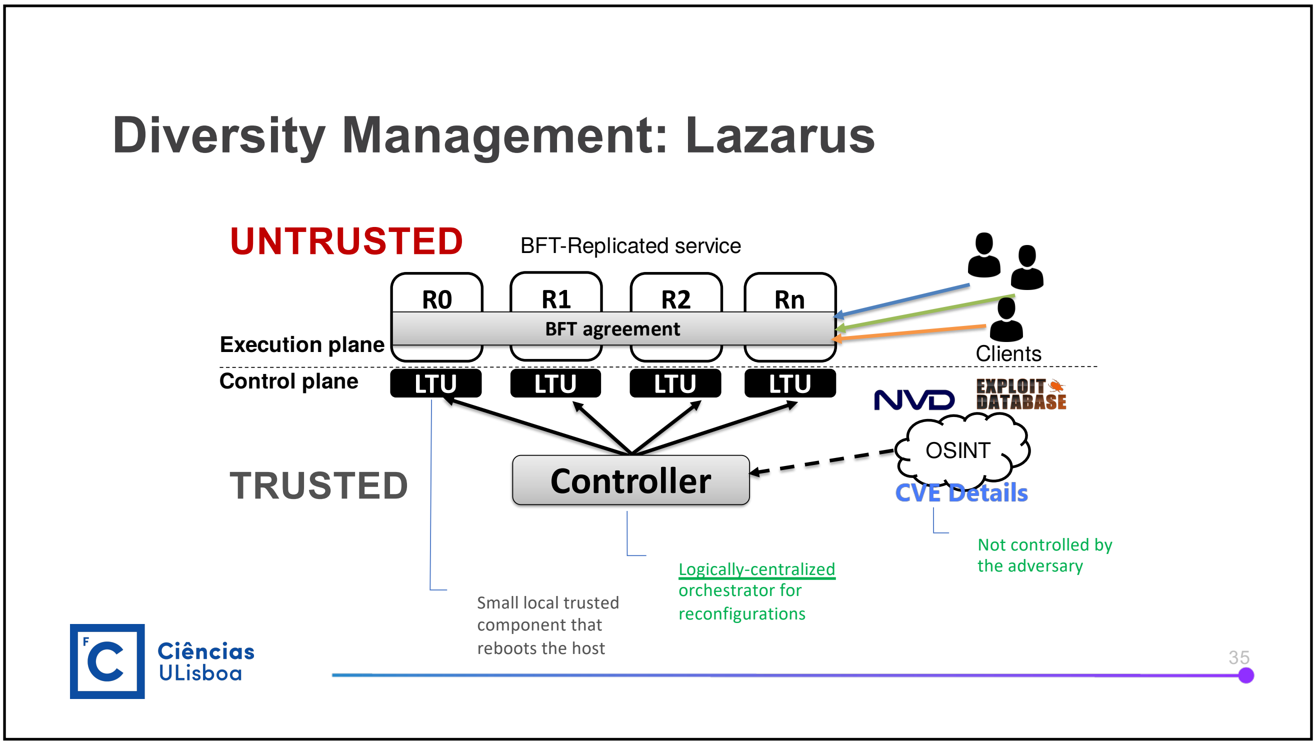
Figure 2: Possible quorums for $n = 5$, $f = 1$, $\Delta = 1$ (BFT).

Figure 4: Message flow of BFT AWARE ($f = 1; \Delta = 1$).



Geo-replication: WHEAT & AWARE [SRDS'15,'19]

- 5 replicas spread around the world, latency observed on these sites



Diversity Management: Lazarus

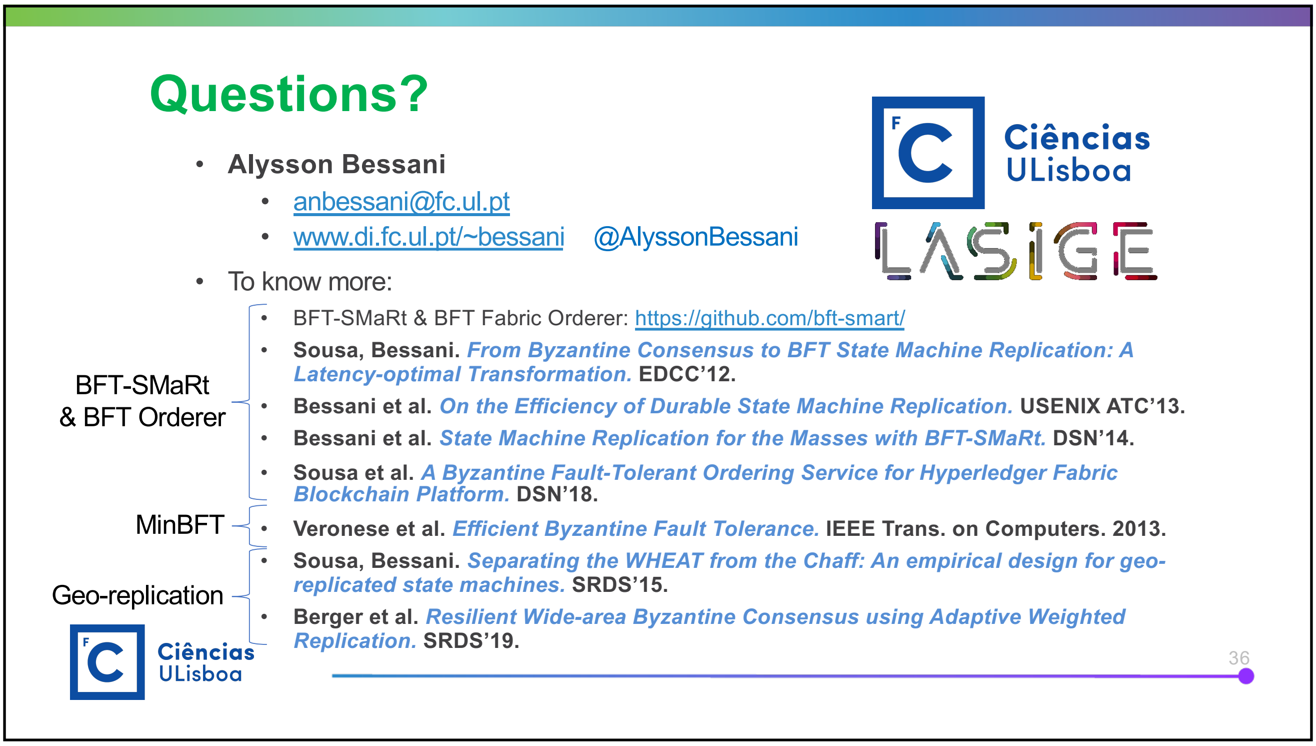
Small local trusted component that reboots the host

Logically-centralized orchestrator for reconfigurations

Not controlled by the adversary

Ciências ULisboa

35



Questions?

- **Alysson Bessani**
  - anbessani@fc.ul.pt
  - www.di.fc.ul.pt/~bessani @AlyssonBessani
- To know more:

BFT-SMaRt & BFT Orderer

- BFT-SMaRt & BFT Fabric Orderer: https://github.com/bft-smart/
- **Sousa, Bessani.** ***From Byzantine Consensus to BFT State Machine Replication: A Latency-optimal Transformation.*** **EDCC'12.**
- **Bessani et al.** ***On the Efficiency of Durable State Machine Replication.*** **USENIX ATC'13.**
- **Bessani et al.** ***State Machine Replication for the Masses with BFT-SMaRt.*** **DSN'14.**
- **Sousa et al.** ***A Byzantine Fault-Tolerant Ordering Service for Hyperledger Fabric Blockchain Platform.*** **DSN'18.**

MinBFT

- **Veronese et al.** ***Efficient Byzantine Fault Tolerance.*** **IEEE Trans. on Computers. 2013.**

Geo-replication

- **Sousa, Bessani.** ***Separating the WHEAT from the Chaff: An empirical design for geo-replicated state machines.*** **SRDS'15.**
- **Berger et al.** ***Resilient Wide-area Byzantine Consensus using Adaptive Weighted Replication.*** **SRDS'19.**

Ciências ULisboa

LASIGE

36