In Trusted Components we Trust!

Suyash Gupta

SkyLab
University of California, Berkeley
https://gupta-suyash.github.io/

Awesome Collaborators

Natacha Crooks

University of California, Berkeley

Sajjad Rahnama

Mohammad Sadoghi

University of California, Davis


BFT
ME
imgflip.com

Byzantine Failures Do Not Exist!

Subscribe to receive notifications of new posts:

Email Address | Subscribe

CLOUDFLARE The Cloudflare Blog

Product News | Speed & Reliability | Security | Serverless | Zero Trust | Developers | Deep Dive | Life @Cloudflare

A Byzantine failure in the real world

11/27/2020

Tom Lianza Chris Snook

An analysis of the Cloudflare API availability incident on 2020-11-02

Cores that don't count

Peter H. Hochschild
Paul Turner
Jeffrey C. Mogul
Google
Sunnyvale, CA, US

Rama Govindaraju
Parthasarathy
Ranganathan
Google
Sunnyvale, CA, US

David E. Culler
Amin Vahdat
Google
Sunnyvale, CA, US

Byzantine Fault Tolerance, from Theory to Reality

Kevin Driscoll[1], Brendan Hall[1], Håkan Sivencrona[2], Phil Zumsteg[1]

[1]Honeywell International
3660 Technology Drive, Minneapolis, MN 55418
{brendan.hall,kevin.driscoll,phil.j.zumsteg}@Honeywell.com
[2]Chalmers University of Technology
Department of Computer Engineering, SE-412 96 Göteborg, Sweden
sivis@computer.org

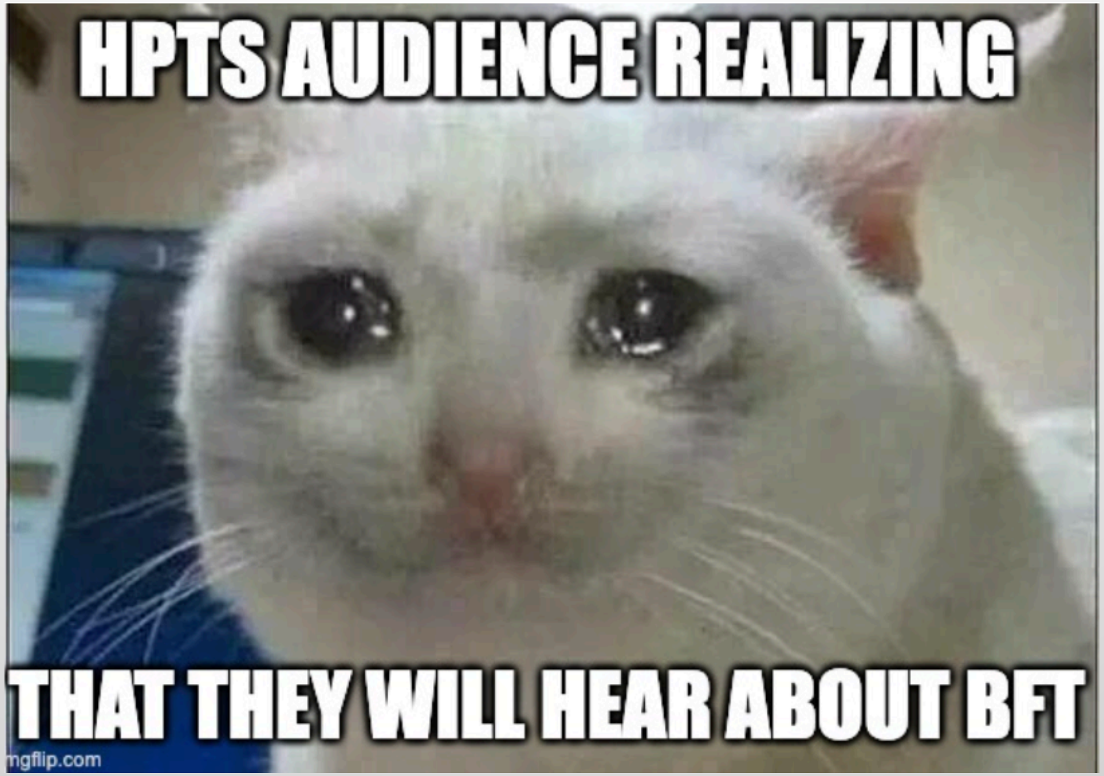
Real World Examples of Byzantine Failure

On NASA's DASHlink, there is a collection of real system failures that they encountered. These web pages also describe some phenomenology that can cause Byzantine faults.

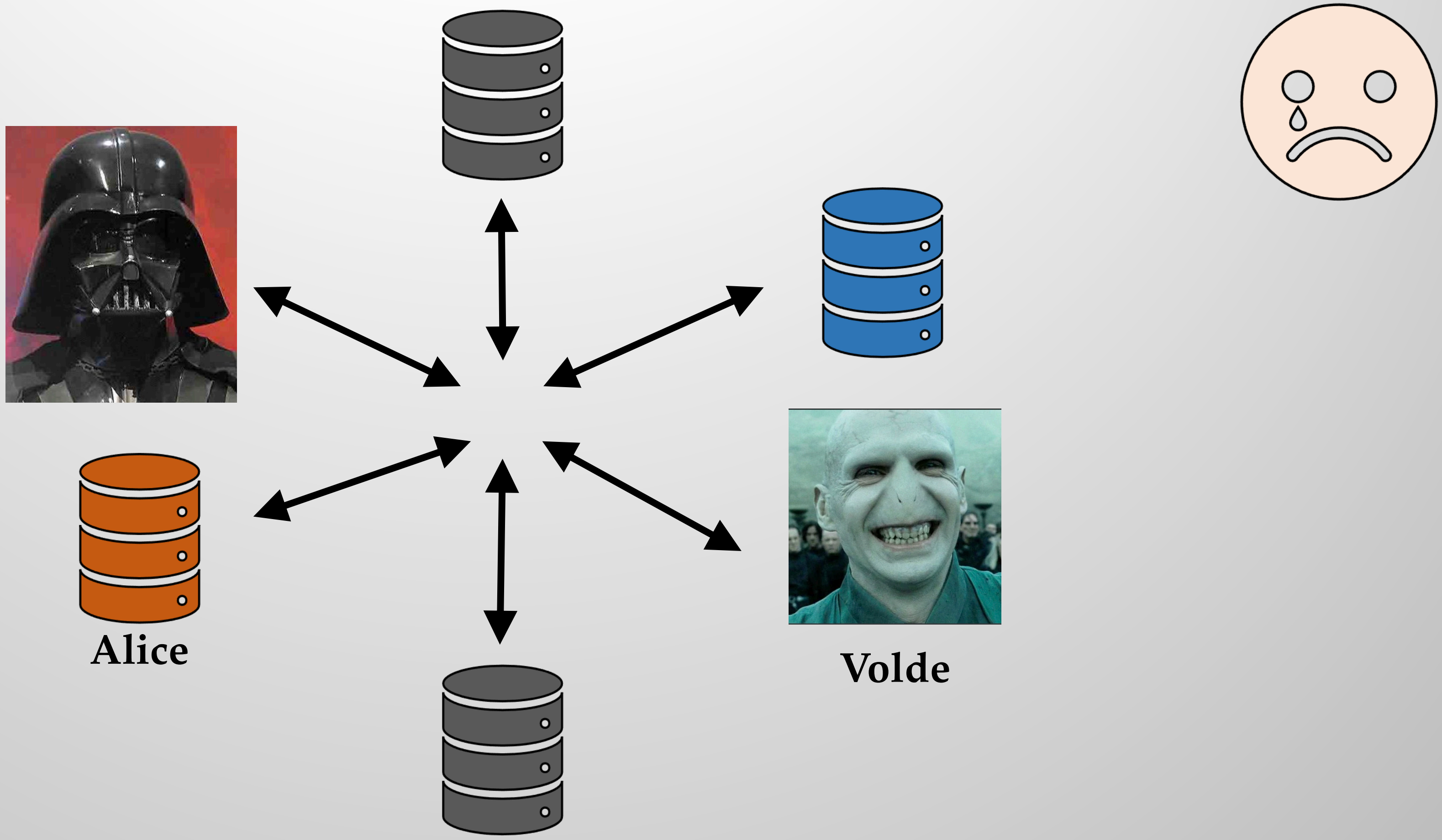
List of Scenarios Outline (1)

Honeywell

- Leading the list are examples of Byzantine failures because of the very widespread "that can't happen" disbelief in these failures and the fact that there are known solutions
- Introduction to the Byzantine Generals Problem
  - Definitions of Fault, Failure, and Error
  - Byzantine failure definitions and background
- Examples of actual occurrences
  - Space shuttle mission STS-124
  - Space shuttle data bus standing wave
  - Mid-value select
  - Command / Monitor wrap-back
  - Time-Triggered Protocol (TTP/C) heavy ion fault injection
  - Multi-Microprocessor Flight Control System (M^2FCS)
  - Potential grounding of an entire aircraft fleet
  - A pushbutton input to the command and monitor lanes of an airplane brake system caused the system to fail (see section 1.6.7 of www.fss.aero/accident-reports/dvdfiles/ES/1998-05-21-ES.pdf)


HPTS AUDIENCE REALIZING
THAT THEY WILL HEAR ABOUT BFT
mgflip.com

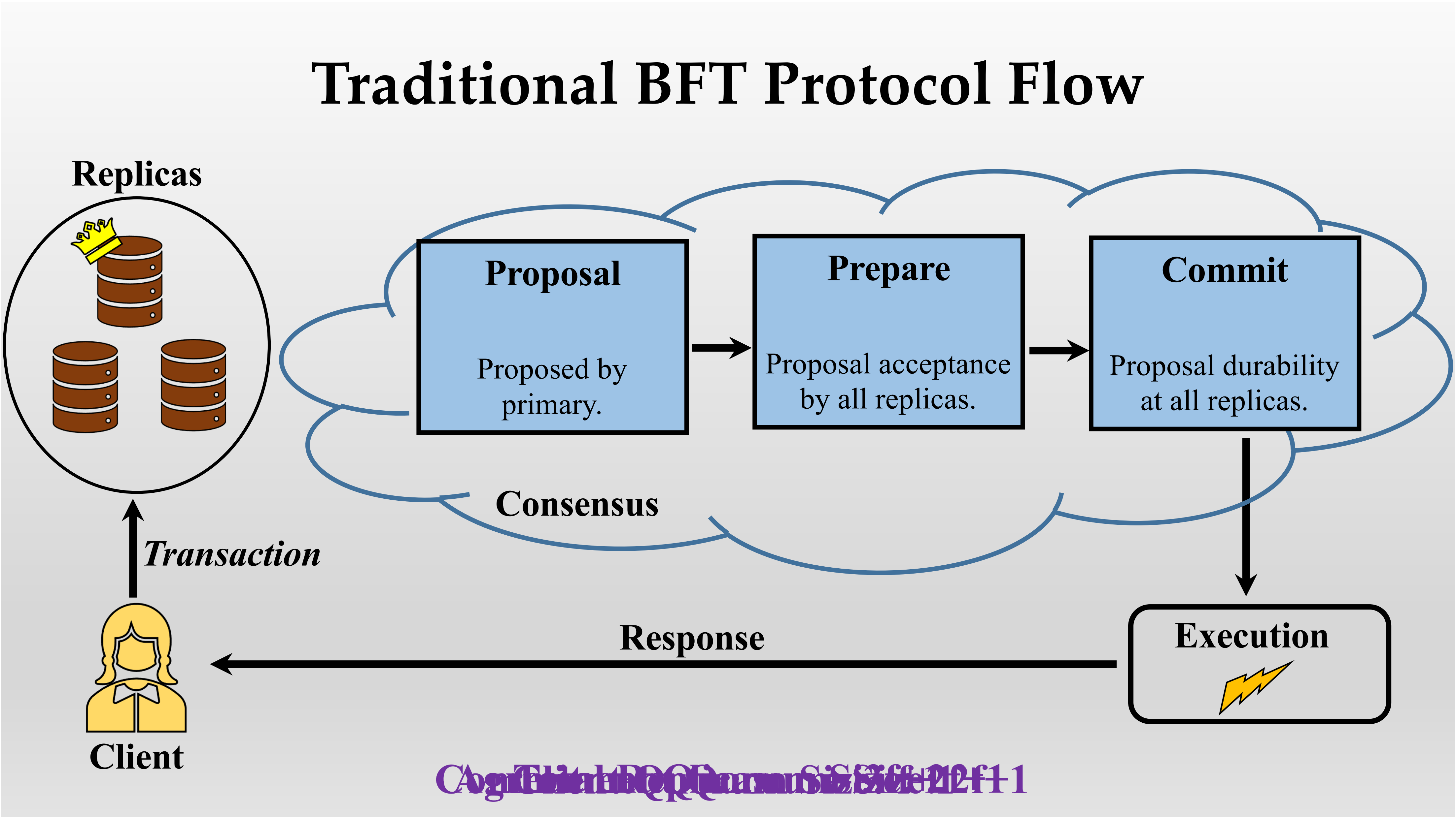
Everything is fine until there are Adversaries!



Solution →

Byzantine-Fault Tolerant Consensus

- ✓ Safety
- ✓ Liveness


Traditional BFT Protocol Flow
Replicas
Proposal
Proposed by primary.
Prepare
Proposal acceptance by all replicas.
Commit
Proposal durability at all replicas.
Consensus
Transaction
Execution
Response
Client

BFT Consensus is Expensive!

- Paxos requires $n = 2f + 1$ replicas.
- BFT protocols require $n=3f+1$ replicas.
  - Why 3f+1?
  - Byzantine replicas can equivocate!
  - Equivocation leads to massive communication.

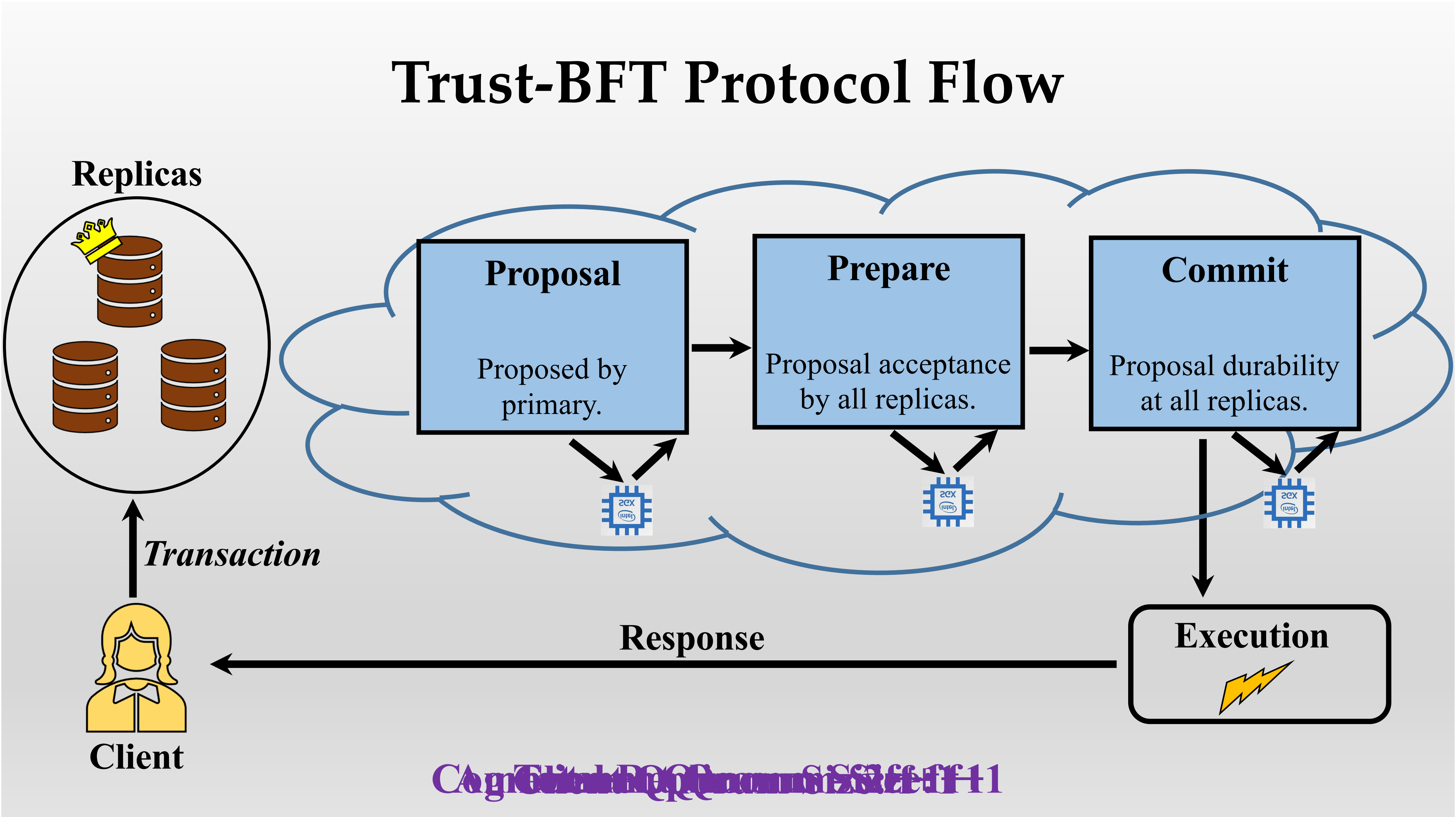
Need for Trust

Keystone

intel SGX

Trusted BFT Protocols

- Require each replica to have a co-located trusted component.
- **Assumption**: Trusted component cannot be compromised.
- Trusted component attests order for each client request.
- **Fault-Tolerance**: In a system of n replicas at most f byzantine replicas, n >= 2f+1.


Trust-BFT Protocol Flow
Replicas
Proposal
Proposed by primary.
Prepare
Proposal acceptance by all replicas.
Commit
Proposal durability at all replicas.
Transaction
Response
Execution
Client

So Are We Done?

Unfortunately No!

Challenges for Trust-BFT Protocols!

- We found three challenges with the design of Trust-BFT protocols.
- We show that Trust-BFT protocols target wrong metric.
- **Correct metric → Throughput per Hardware**

Observation 1: Loss of Responsiveness

- Delayed messages from one honest replica → Clients stuck!
- Why? Consensus only need weak quorums → f+1
- Execution still needs quorum of f+1
- Byzantine replicas can always avoid sending messages.

Observation 2: Loss of Safety under Rollbacks

- Trusted Enclaves can be rollbacked!
- Impossibly hard to rollback are TPMs or persistent counters!
- TPMs are too slow → 180ms per access.

Observation 3: Lack Parallel Invocations

- ➢ Every message sent → Requires attestation.
- ➢ A replica cannot run consensus on two transactions in parallel!
- ➢ **We show** that despite 2f+1 replicas → Trusted-BFT slower than BFT.

Solution →

FlexiTrust Protocols

- Novel Suite of Protocols.
- Guarantee both liveness and responsiveness.
- Only one access to trusted component per transaction.
- Minimal Memory consumption → No logging at trusted component!

Magical Ingredient behind FlexiTrust Protocols

- ➢ Switch back to replication factor 3f+1.
- ➢ Trusted hardware still useful to reduce phases and communication.
- ➢ Only primary accesses trusted hardware before sending proposal!
- ➢ Throughput per hardware more than Trusted-BFT.

➢ One Line Conclusion:

- Simply reducing replication will not yield better throughput.
- FlexiTrust protocols advocate meaningful application of BFT consensus.

➢ Check out:

- ***ResilientDB – VLDB'20*** (https://github.com/resilientdb/resilientdb)
- ***Basil – SOSP'21*** (https://medium.com/initc3org/decentralizing-databases-with-basil-604827608ff8)
- ***S. Gupta***, J. Hellings and M. Sadoghi, ***Fault-tolerant Distributed Transactions on Blockchain***, Morgan & Claypool Synthesis Lectures on Data Management and Springer, 2021.

➢ Reach me:

- **Twitter:** suyash_sg
- **Email:** suyash.gupta@berkeley.edu