

# Efficient Middleware for Byzantine Fault Tolerant Database Replication

Rui Garcia<sup>1</sup>

Rodrigo Rodrigues<sup>2</sup>

Nuno Preguiça<sup>1</sup>

<sup>1</sup> CITI / DI – FCT – Universidade Nova de Lisboa

<sup>2</sup> MPI-SWS

## 1. Motivation

Databases are central in computing infrastructures  
Byzantine faults occur in practice:

- Software bugs
- Hardware errors
- Intrusions

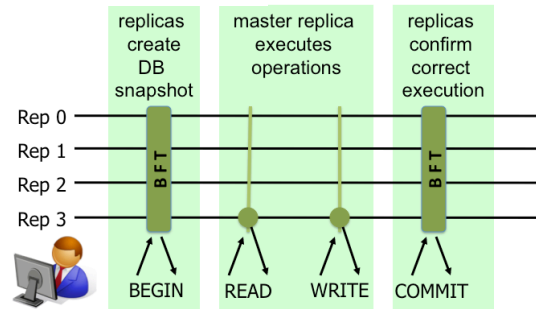
### Goal:

Efficient database BFT replication

### Challenges:

Avoid serializing every operation through BFT  
Exploit weaker consistency (snapshot isolation)

## 2. Basic solution



- Only run begin/commit as BFT operations
- Replicas must confirm tentative execution

## 3. Limitations

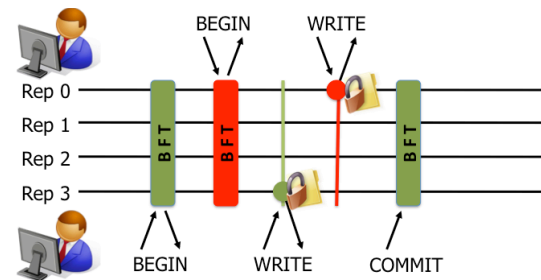
(1) Database systems use locks  $\Rightarrow$   
Need to avoid deadlocks in the system

Two solutions

- Single master
- Multi-master

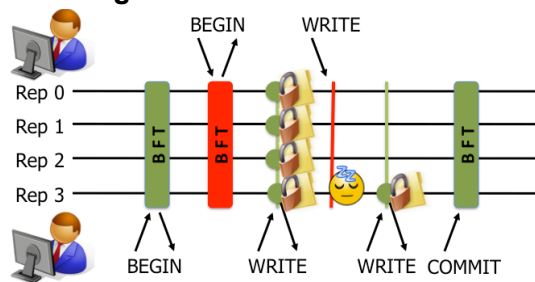
(2) Read-only transactions execute in all replicas  
Execute read-only transaction in  $f+1$  replicas  
Striping transactions among different replicas

## 4. Multi-master



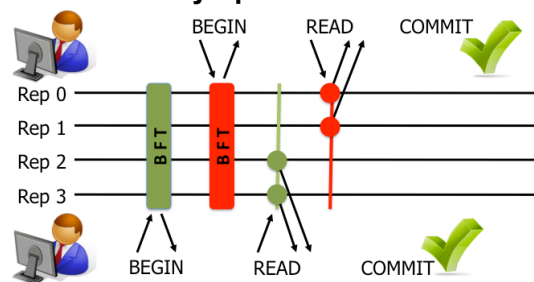
- At commit, execute all operations at non-masters
- Non-masters may have to undo local transactions

## 5. Single master



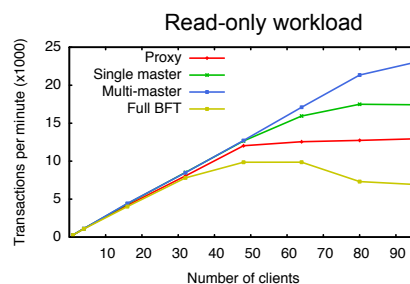
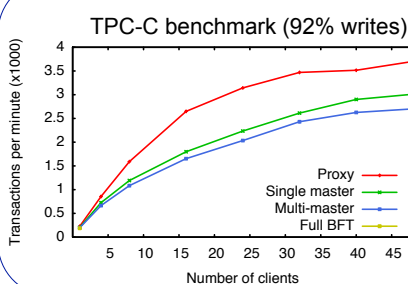
- Optimization: non-masters execute penultimate op
- At commit, only one operation left to execute

## 6. Read-only operations



- Read from  $f+1 \Rightarrow$  correct reply
- Commit confirmed locally if all reads ok

## 7. Evaluation and conclusions



First solution for efficient BFT DBMS without trusted central components

Good performance results

- Modest overhead for R-W
- Striping for improving read-only performance

Several new techniques can be reused