PRACTICAL BYZANTINE FAULT TOLERANCE

JY

04.2018

AGENDA

- System Model
- Service Properties
- The Algorithm
- Resources

SYSTEM MODEL

- Asynchronous
- Cryptographic techniques : prevent spoofing and replays and to detect corrupted Message

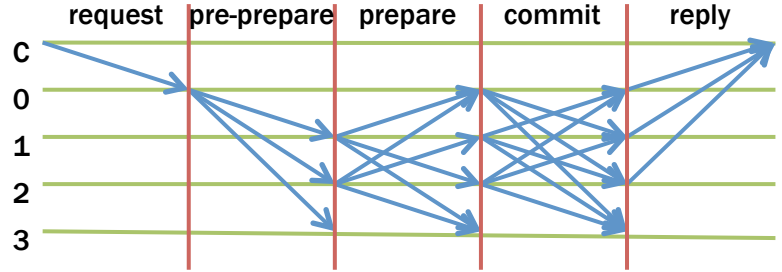
SERVICE PROPERTIES

- PBFT is a new, *practical* algorithm for state machine algorithm replication that tolerates Byzantine faults. The algorithm offers both liveness and safety provides at most (rounding down) (n-1)/3 out of a total of *n* replicas are simultaneously faulty.

- Minimum number of replicas that allow an asynchronous system to provide the safety and liveness properties when up to *f* replicas are faulty : 3*f*+1.

f	**faulty AND not respond**
f	**not faulty AND not respond**
f	**faulty AND not respond**
N* – 2*f* > *f	***N* > 3*f***

THE ALGORITHM



- Set of replica = R.
- Each replica represents $\{0,\ldots,3f\}$.
- For simplicity, assume $|R| = 3f+1$.
- The replicas move through a succession of configurations called *views*.
- In one view, one replica is *public* and the others are *backups*.
- The primary of a view is replica p such that $p = v \pmod{|R|}$, where v is the view number.
- View changes are carried out when the primary is faulty.

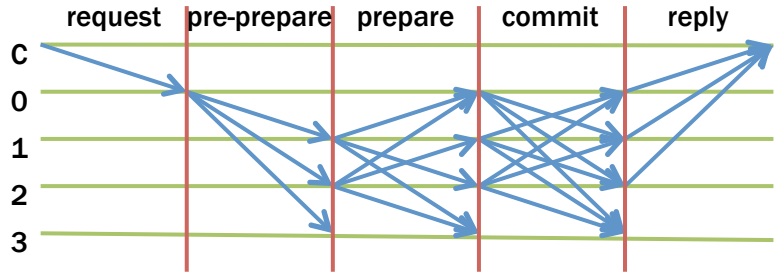
- Roughly,
  1. A client sends a request to invoke a service operation to the primary
  2. The primary multicasts the request to the backups
  3. Replicas execute the request and send a reply to the client
  4. The client waits for $f+1$ replies from different replicas with the same results; this is the result of the operation.

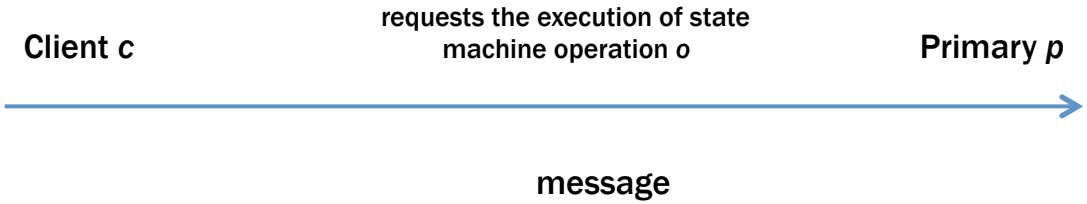
- Two requirements on replicas : deterministic and start in the same state
  All non-faulty replicas agree on a total order for the execution of requests despite failures.

REQUEST

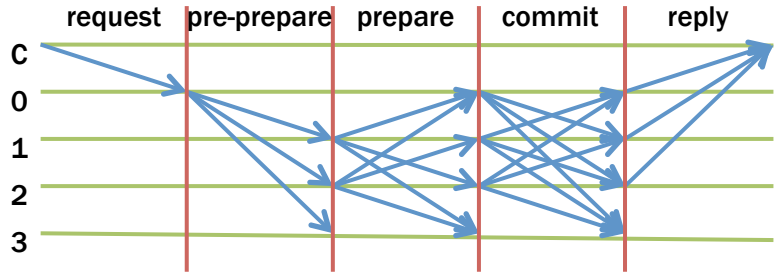
<REQUEST, *o*, *t*, *c*>

- Operation *o*
- Timestamp *t*
- Client *c*





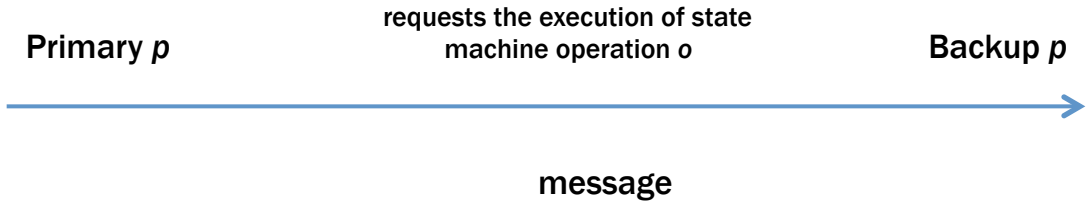
PRE-PREPARE



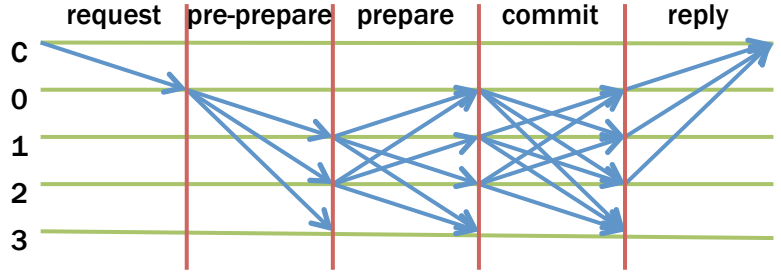
<<PRE-PREPARE, *v*, *n*, *d*>, *m*>

- Primary assigns a sequence number, *n*, to the request, multicasts a pre-prepare message with *m* piggybacked to all the backups, and appends the message to its log.
- View that the message is being sent *v*
- The client's request message *m*
- *m*'s digest *d*

- Requests are not included in pre-prepare messages to keep them small.
- The last condition prevents a faulty primary from exhausting the space of sequence numbers by selecting a very large one.

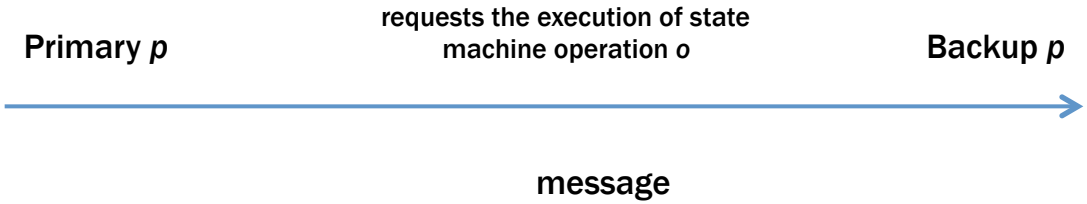


PRE-PREPARE

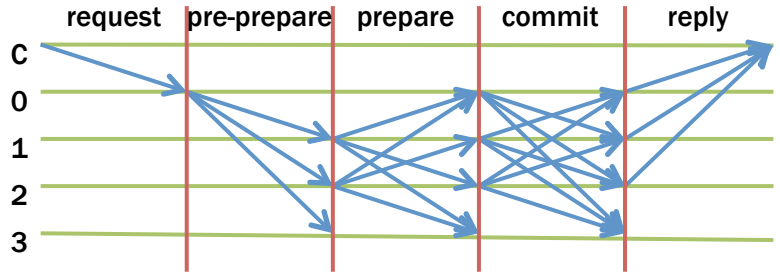


A backup accepts a pre-prepare message provide:

- The signature in the request and the pre-prepare message are correct and *d* is the digest for *m*;
- It is in view *v*;
- It has not accepted a pre-prepare message for view *v* and sequence number *n* containing a different digest;
- The sequence number in the pre-prepare message is between a low water mark, *h*, and a high water mark, *H*.



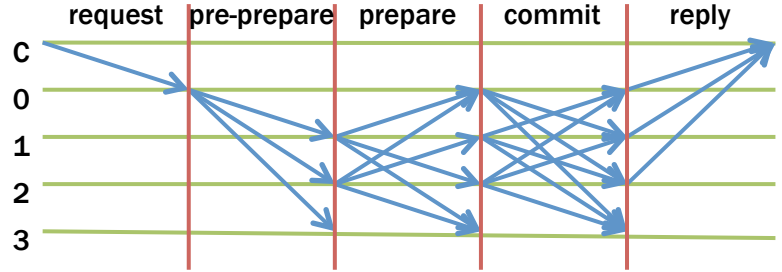
PREPARE



<PRE-PREPARE, *v*, *n*, *d*, *i*>

- Prepared(m,v,n,i)
- Request m
- Pre-prepare for *m* in *v* with sequence number *n*
- (2f) prepares that match pre-prepare

COMMIT

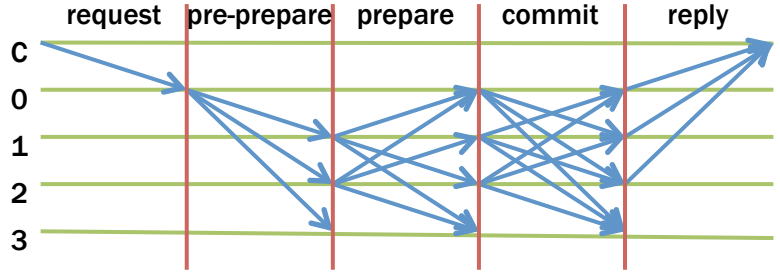


<COMMIT, *v*, *n*, *D(M)*, *i*>

- Replicas accept commit messages and insert them in their log provided they are properly signed, the view number in the message is equal to the replica's current view, and the sequence number is between *h* and *H*.
- *committed(m,v,n)* is true iff *prepared(m,v,n,i)* is true for all *I* in some set of *f+1* non-faulty replicas
- *committed-loca(m,v,n,i)* is true iff *prepared(m,v,n,i)* is true and *i* has accepted *2f+1* commits(possibly including its own) from differenct replicas that match the pre-prepare for *m*
- A commit matches a pre-prepare if they have the same view, sequence number and digest.

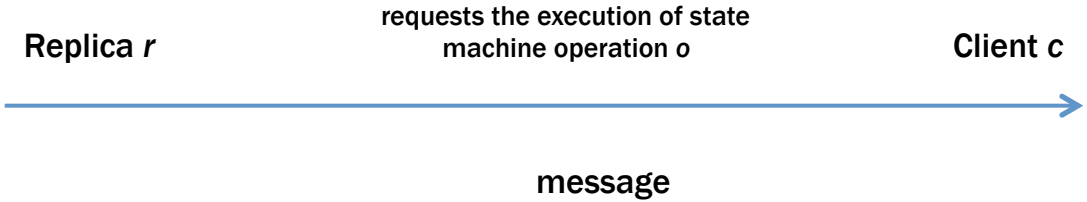
- **If *commited(m,v,n,i)* is true for some non-faulty *i* then *committed(m,v,n)* is true.**

REPLY



<REPLY, *v*, *t*, *c*, *i*, *r*>

- Current view number *v*
- Timestamp of corresponding request *t*
- Client *c*
- Replica number *i*
- Result of executing the requested operation *r*



- The client waits for *f+1* replies with valid signatures from different replicas, and with the same *t* and *r*, before accepting the result *r*.

RESOURCES

- Miguel Castro and Barbara Liskov. Practical Byzantine Fault Tolerance 19999.