BFT with Additional Components

Administrivia

- Logistics
  - Poster session next week
  - Report due May 7
  - Exam May 8

} See posting on Campuswire

<u>Where we are</u>

Impossibility results

Async { - FLP

All { - Unauth BFT w $\leq 3f$ nodes

- ...

But Solvable in some practical scenarios

New model assumptions

- Partial synchrony
- Authentication
- Failure Detectors

Reality

- Fail-stop / fail-recover protocols are widely used
  - ↳ Likely interacted with one in the last few hours
- Modern networks engineered & provisioned so that partial synchrony holds
  - ↳ Warning: Not a formal statement
    - ↳ But things "mostly just work"
- BFT deployments are rarer
  - ↳ Concern: Cost ($$$) / Performance vs Utility

→ Where deployed – trade-offs make these concerns moot

- Specialized areas
  ↳ Space
  → Critical Infra

- Cryptocurrency, etc.

Common (practical) question

↳ Can we add something to our networks on machines to change tradeoffs?

Ideally

Cost of something (think like FD)
+
Cost of BFT w/ something
$<$ Cost of BFT today

Today's papers

- Proposal for something
- Analysis of whether, why & how

Notes

① The precise elements are somewhat influenced by history.

TPMs (~2009): Platform integrity checks

- ↳ H/W RNG
- → Attest to hardware & software
- → Binding
- → ...

Enclaves (~2013/14): Protect state from adv. w/ physical access

- ↳ Encrypt data on the memory BUS

Both of these are reasonably deployed

- ↳ Nearly everyone has a TPM module
  - ↳ Requirement to boot some versions of Windows & OSX
- → Most reasonably new phones have enclaves

Deployment concerns are not the main impediment to adoption.

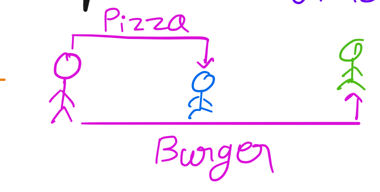
② The analysis (both in Power of Non-Equiv & Disecting BFT) while a little less formal than previous papers take a similar form to before, so should be something you can follow

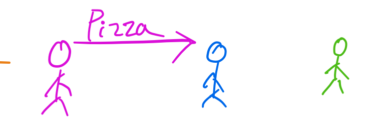
So should be something

From two classes ago:

How can Byzantine failures manifest

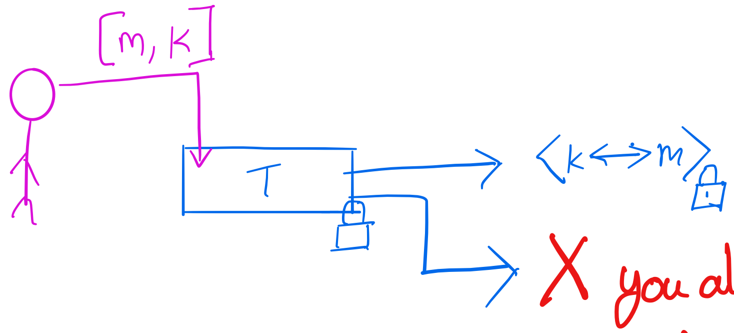
- Crashes ——— Others
- Sending malformed messages } ——— Easy to detect & avoid effects
- Internal state corruption ——— Unnoticable
- Equivocation
- Omission
- ...





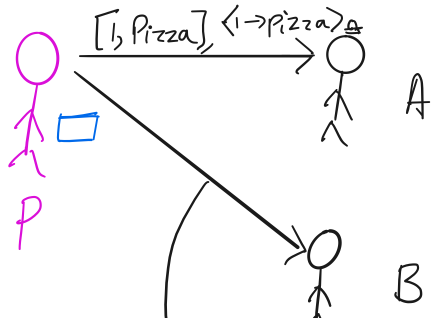
Avoiding Equivocation

- Core idea binding (TrInc, ...)



X you already assigned k to a different message

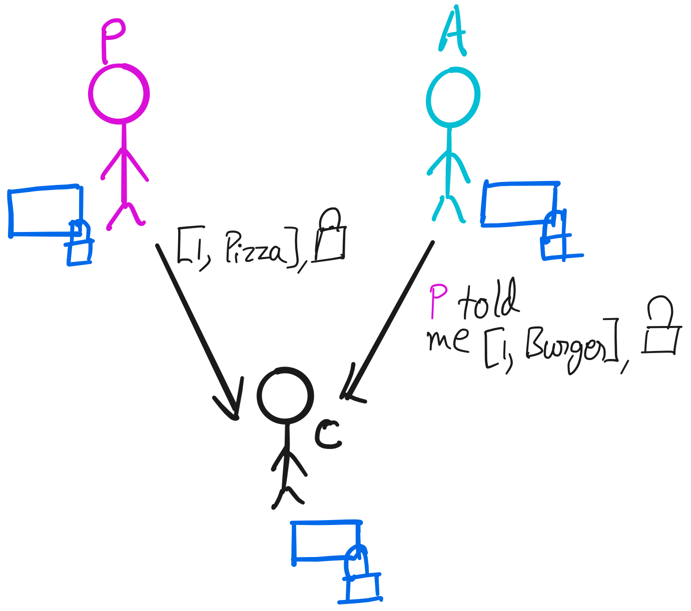
What does this mean



[2, Burger], <2, Burger>$_B$ ✓

[1, Burger] <1, Burger>$_B$ ✗

Core concern / conflation Power of Non-Equivocation is trying to raise



When possible?

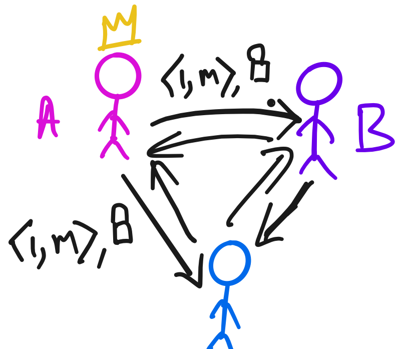
Non-equivocation $\neq$ Authentication

Result: Even with non-equivocation, BFT requires $\geq 3f+1$ nodes

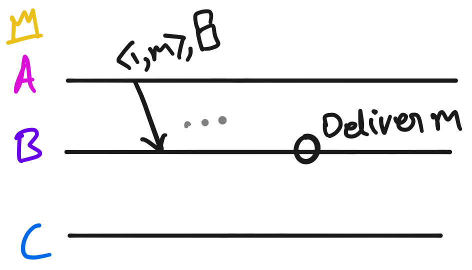
Agreement

Termination

Validity $\longrightarrow$ If proposer is honest, deliver proposal

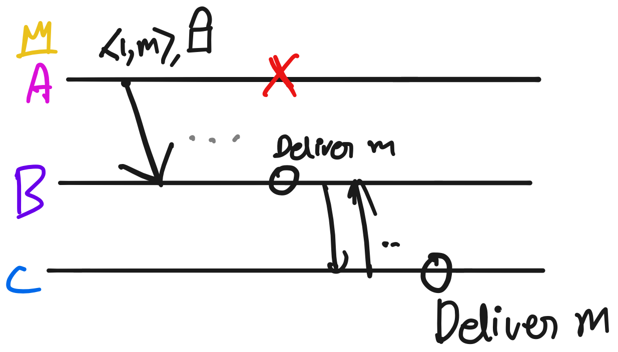


1



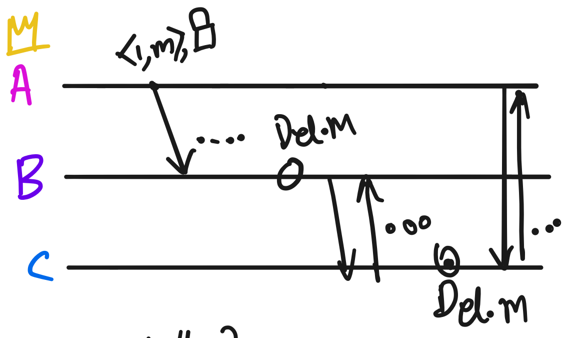
Why?

2



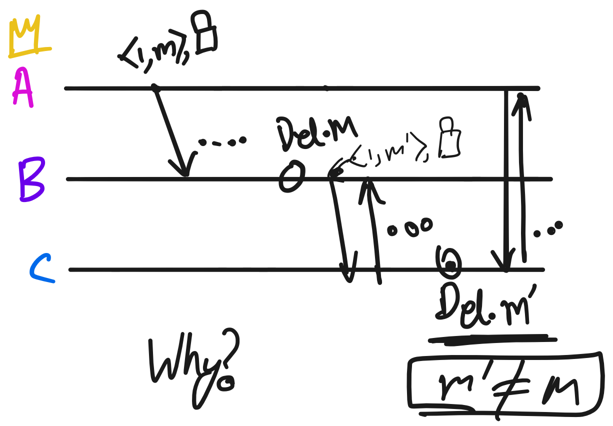
Why?

3



Why?

4



Why?

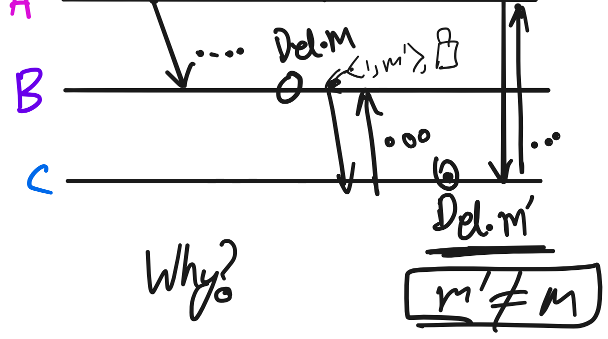
Q1. What is the prog assuming

synchrony?

part synchrony? ✓

asynchrony? ✓

Q2. How to fix?



Non-equivocation + Authentication is better?

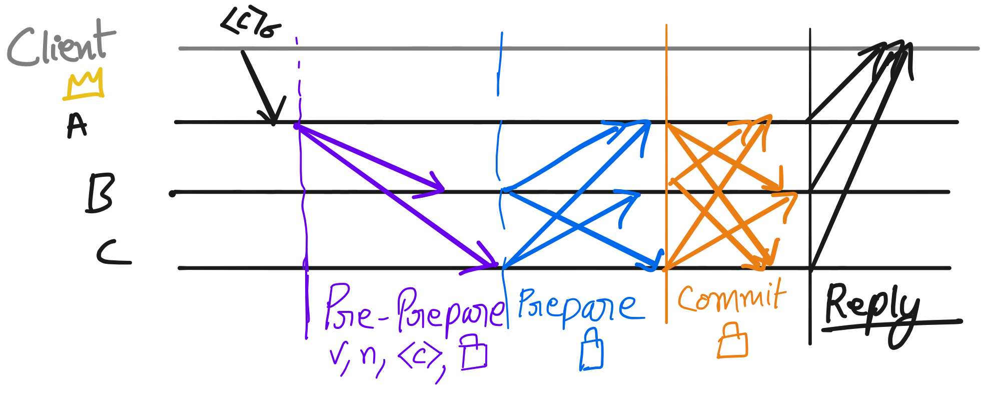
Basic idea: - No sender can equivocate

- Safe to use relayed messages

[If process A relays a message for process B by forwarding it to C ; C can check relay correctness]

$\Longrightarrow$ Broadcast received messages to avoid omission

PBFT-EA [A2M]



Core idea : 2f + 1 is enough

Why? Limited how faulty nodes

can behave

What doesn't change

- Clients still wait for $f+1$ responses.
  - $\hookrightarrow$ Why?

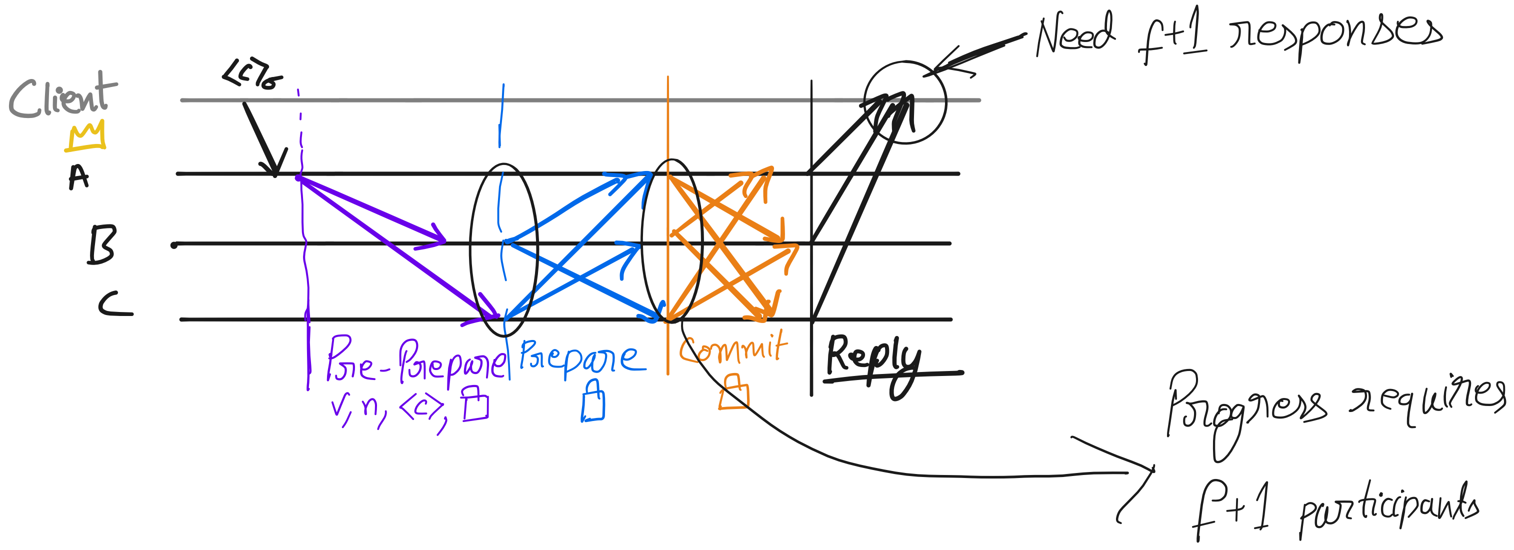
- Clients still broadcast command after delay
  - $\hookrightarrow$ Why?

What changes

- View change
  - $\hookrightarrow$ - Can only require $f+1$ requests to initiate
    - Must ensure that a node can participate in view v after suggesting moving to $v+1$. Why?

- Details: See A2M paper (if interested, not required

Getting us to Dissecting BFT



Can set up scenarios where client gets stuck

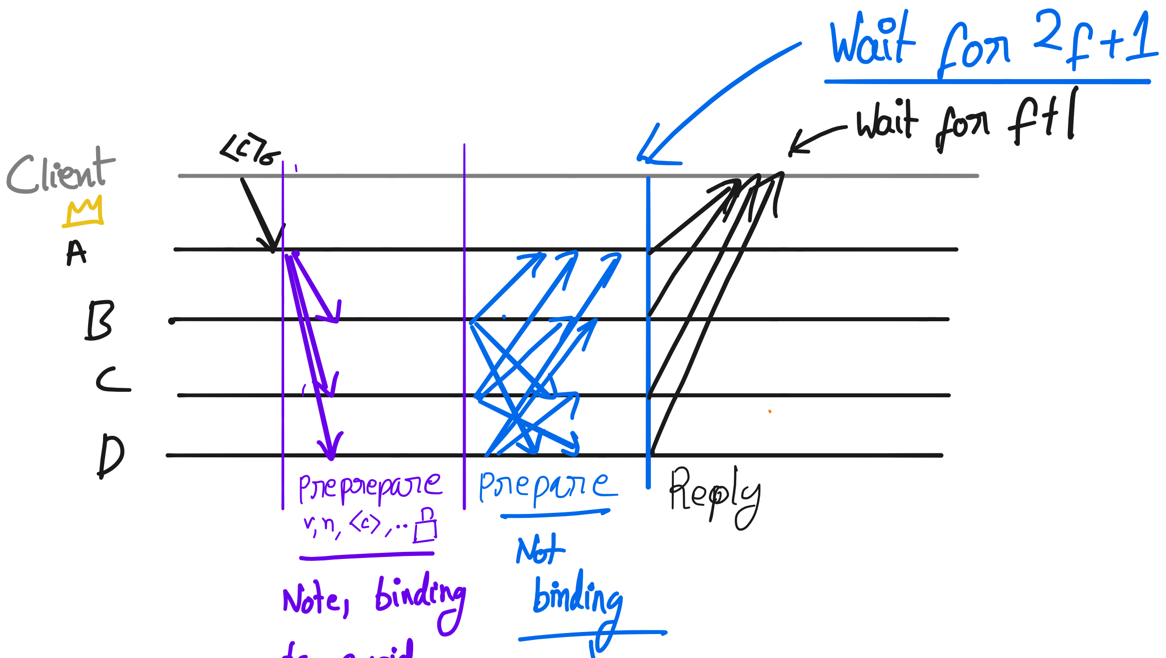
- A faulty – Runs protocol with only B
- Client gets response from B
  - ↳ Not sufficient to make progress
- Client cannot trigger view change
  - ↳ B won't trigger V.C, saw client message
  - → A is faulty
  - → C alone insufficient

Also, mismatch b/w what enclaves provide & TrINC/A2M

Not a sign that any paper got it wrong, just a reflection on what got deployed

So big claim

- TrInc / A2M + PBFT-EA w/ 2f+1 nodes
  - ↳ Not live
- Are they still useful in some way?
  - ↳ Reduce number of rounds



Relays command, preventing omission!

Aside: Non-Equivocation translation from CFT → BPF

Basic idea – ① Nodes send history w/each messag

② On receiving history, simulate & chec
that message would be sent, etc.

③ Deliver.

Trickier to do than it seems

Observations at the End.