BFT State Machine Replication with 2f+1 Replicas

What good are hybrid models and what hybrid models are good

Miguel Correia

Joint work with Paulo Veríssimo, Nuno Neves, Alysson Bessani, Giuliana Veronese, Lau C. Lung

Outline

• 2002: Wormholes, TTCB, BRM
• 2004: BFT-TO and TOW
• 2007: A2M-PBFT-EA
• 2008-....: MIN-BFT, EBAWA, USIG
• 2010: 2f+1 Consensus
2002: WORMHOLES, TTCB, BRM


Wormhole model / hybrid fault model

- Most of the system has weak guarantees
  - e.g., asynchronous, Byzantine faults
- **Wormhole**: a subsystem built to provide stronger properties (aka **trusted component**), e.g., partial synchronous, crash faults
Why hybrid system models?

- Expressive models w.r.t. reality
- Sound theoretical basis for proofs of correctness
- Naturally supported by hybrid architectures (like the wormholes architecture)
- Enablers of concepts for building totally new algorithms

TTCB

- TTCB – a wormhole to support the execution of intrusion-tolerant algorithms/applications
  - They run mostly in the payload system that can be attacked
  - They use the TTCB to execute some critical steps securely
BRM – 2f+1 BFT reliable multicast

- BRM = Byzantine-resilient Reliable Multicast
  - Based on the TTCB agreement service that runs inside the TTCB (crash faults, better synch)
  - The service tells which one is the correct hash

2004: BFT-TO AND TOW


M. Correia, N. F. Neves, P. Veríssimo. BFT-TO: Intrusion Tolerance with Less Replicas. Computer Journal, Accepted for publication. (extended version of the previous paper)
BFT-TO – 2f+1 BFT SMR

• Wormhole = TOW (Trusted Ordering Wormhole)
  – distributed like the TTCB, only in the servers (not clients)
• Basic algorithm:
  – Client sends request to one server, which sends to the rest
  – When getting the request, serves tell the TOW about it
  – TOW runs internally an agreement and tells servers the order in which
    they must run it
  – When a server processes the request, sends reply to client
  – Client picks the reply most voted

BFT-TO execution

n=3  f=1

P0

P1

M1

P2

TTCB

tmo

f+1 processes have M1 order = 1

message delivery
A2M-PBFT-EA – 2f+1 BFT SMR

- Chun et al. 2007
- Wormhole: A2M (Attested Append-only Memory)
  - equips a host with set of trusted, undeniable, ordered logs
  - interface with several ops: append, lookup, end, truncate, advance
  - local, not distributed (unlike the TTCB)
- A2M-PBFT-EA: first 2f+1 BFT SMR with a local wormhole

2008-....: MIN-BFT, EBAWA, USIG


Simpler wormhole: USIG

- TOW is complex (distributed, agreement); A2M has complex API, memory grows
- **USIG**: local wormhole, one service, one call, simple
  - Single call: `createUI(m)` — assigns a unique ID to a message `m`
  - Includes only (monotonic) counter + signature function
- How does it help?
  - Faulty server can’t send two messages with the same ID
  - Faulty server can’t “go back” and use/reuse “old” IDs
  - ...because the service won’t return such IDs signed

USIG

- Optionally: counter + MAC function
  - faster
  - but verification must also be part of the wormhole (a 2nd call)
- Local service means it can be some hardware chip in server
  - We've implemented it on top of the Trusted Platform Module (TPM), “a commercial wormhole”
- Very similar to Trinc, developed in parallel (1st pub. 2009)
MinBFT – $2f+1$ BFT SMR

- Wormhole: USIG
- Message pattern similar to Castro&Liskov’s PBFT...
- ...but less $f$ replicas, 1 communication step less:

MinBFT throughput (~2009)

- MinZyzyva: a similar algorithm but based on Zyzyva (speculative)
EBAWA – $2f+1$ BFT SMR for WANs

- Wormhole: USIG
- Rotating primary: the primary only orders a batch of reqs
  - performance attacks / load balancing (we did it before in the Spinning alg.)
  - Merge operation provides liveness when the primary is faulty
- Asynchronous views:
  - a server starts an agreement as soon as it receives a client request by sending a prepare message
- Servers without pending client requests skip their turn
  - by sending a special message
- Measurements in LAN / PlanetLab / emulated WAN ...
  - competitive in LANs, outperforms all in several WAN settings

CheapBFT – $f+1$ BFT SMR

- Kapitza et al., 2012
- Wormhole: USIG
  - Implemented USIG in hardware (FPGA)
- CheapBFT
  - Runs CheapTiny with $f+1$ replicas in the normal case
  - Falls back to MinBFT
2010: 2F+1 CONSENSUS

Miguel Correia, Giuliana Santos Veronese, Lau Cheuk Lung,
Asynchronous Byzantine Consensus with 2f+1 Processes, in Proceedings

Byzantine Consensus with 2f+1 Processes

- Question: how to do BFT consensus with 2f+1 replicas? Who’s
  the culprit behind 3f+1?
- Reliable multicast needs 3f+1 but if we use USIG (or TTCB or
  TOW or A2M), then f+1 are enough
- We have shown that (f+1) reliable multicast is enough to solve
  2f+1 consensus (with a few tricks more)...
- ...by giving a methodology to transform CFT consensus
  algorithms into BFT consensus algorithms
Transforming CFT->BFT consensus

Four steps:
1. reliable channels → authenticated reliable channels
2. broadcast → reliable broadcast
3. message reception → message reception + validation
4. Wait for messages from N-f processes → same thing + wait for either messages or suspicions of the other f processes (using special muteness failure detector)

Transforming Mostefaoui/Raynal’s CFT consensus algorithm

1. estimate ← proposal
2. loop
3. coordinator = round \( \mod N \)
4. // --------------------------- phase 1 ---------------------------
5. if coordinator then reliable broadcast message (phase1, estimate, round)
6. wait until valid phase1 message is received from the coordinator or the coordinator is suspected
7. if message received then estimate = estimate in message
8. // --------------------------- phase 2 ---------------------------
9. reliable broadcast message (phase2, estimate, round)
10. wait until valid phase2 messages received from at least N-f processes and the rest (if any) are suspected
11. if same estimate in N-f messages then broadcast decision message and decide
12. if same estimate in N-2f messages then set estimate to that one
13. endloop
14. upon valid decision message received, broadcast decision msg. and decide
Summary

• 2f+1 BFT SMR, 10+ years of research
• Based on a well-defined hybrid fault model
• Distributed vs local wormholes
• USIG: as simple as it can be?
• MinBFT: as simple/efficient as CFT SMR?