CS856: Presentation of "Rollback-Recovery for Middleboxes"

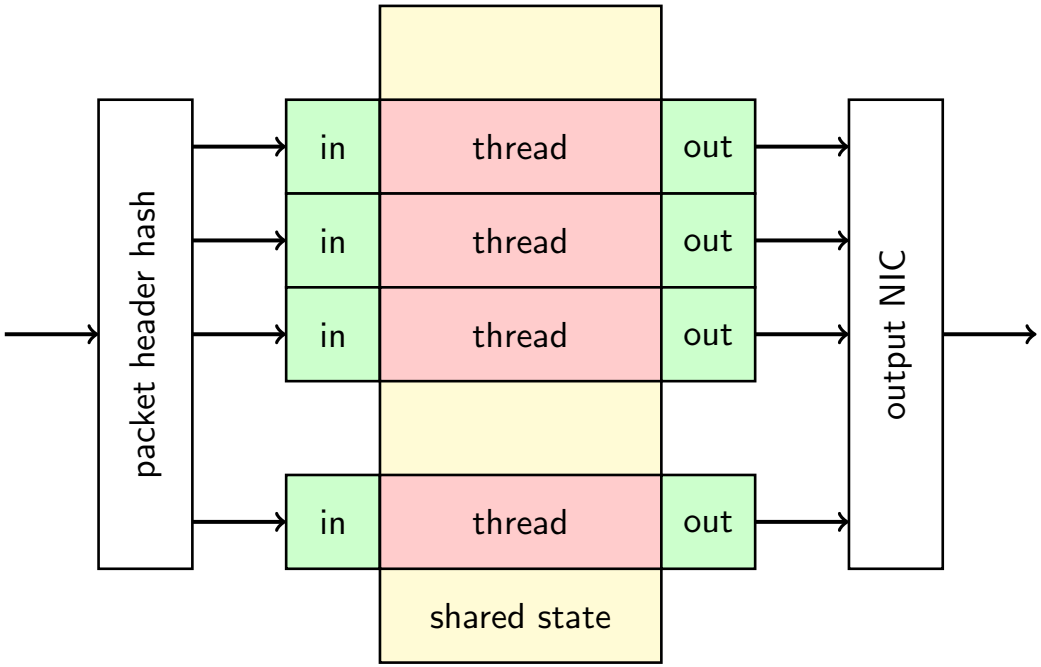
Adrian Nicoara

Motivation

- Dedicated middlebox hardware, with backup, is expensive
- Middlebox applications that use NFV run on diverse hardware platforms, which have a higher probability of failure

Middlebox application model

- **Thread local state:** packet data
- **Shared state:** counters, IDS state machine, rate limiter etc.



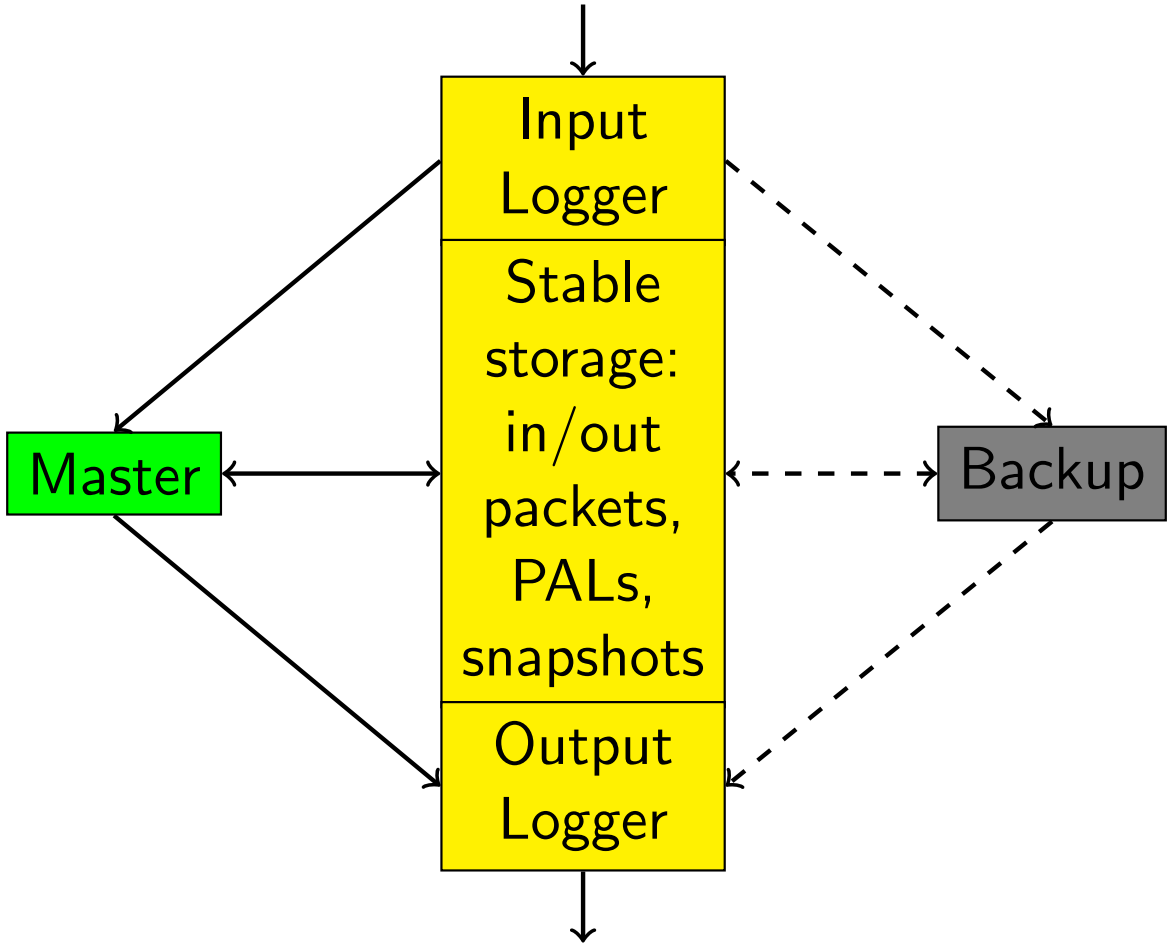
Challenges to recovery

- **Statefulness**: Shared variables (e.g. counters) need to be restored before processing new packets
- **Non-determinism**: Access order to shared variables is important; access of hardware clocks needs to be reproduced for stateful recovery
- **Low latency**: Normal operation needs to be in the order of microseconds

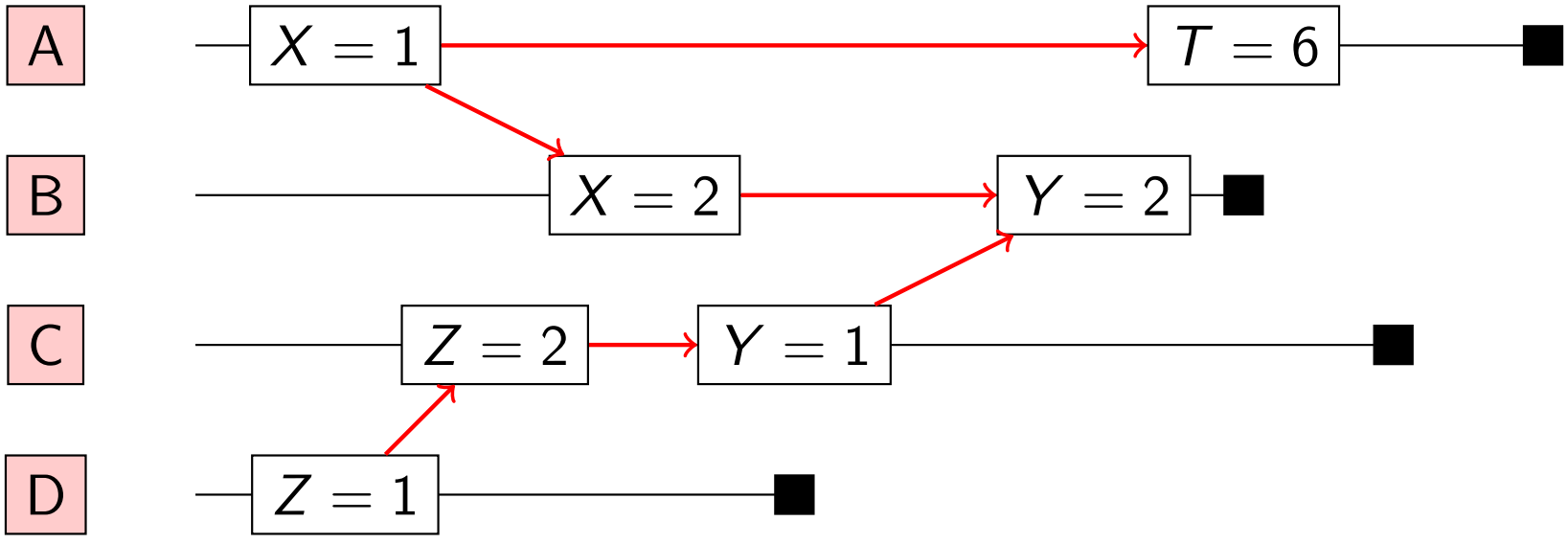
Replay vs No-Replay designs

- **No-Replay**: Snapshot of system and buffering of output until next snapshot. Simple, but slow.
- **Replay**: Snapshot of system and write-ahead-logging of input in between snapshots. Output is only released after input is safely logged. Lower latency per operation, but more expensive logging.

FTMB architecture



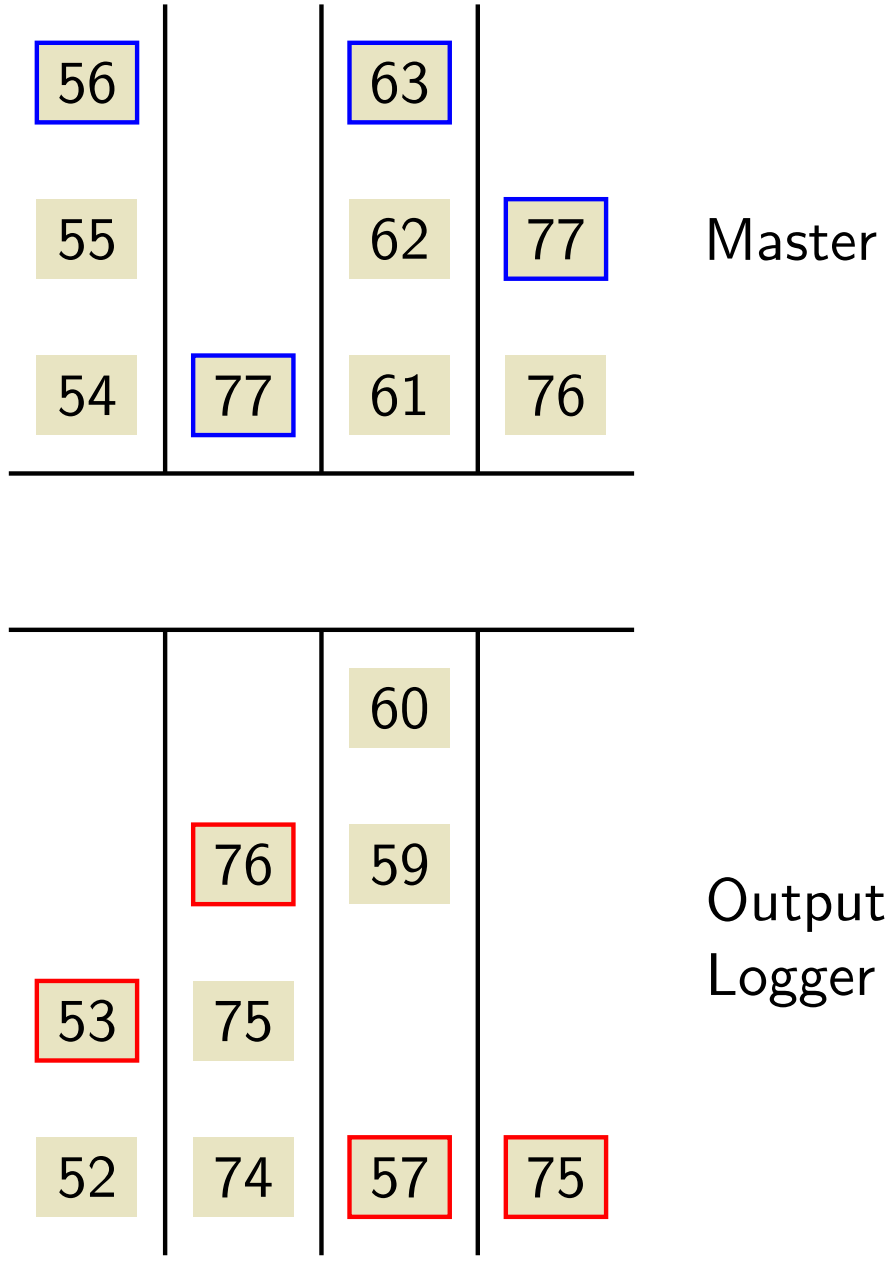
Packet dependencies



Packet can be released when all its causal dependencies have been recorded to safe storage as Packet Access Logs (PALs).

Parallel release

- **Master**: Packet is released together with a vector clock representing the number of PALs each queue has processed: e.g. next packet has vector clock $[56, 77, 63, 77]$.
- **Output Logger**: Packet is released when each queue has processed more PALs than the current value of the vector clock: e.g. $[45, 76, 60, 70]$ can't be released because of the third queue.



Implementation

- **Input Logger:** Buffers incoming packets, in between snapshots, for replay. Adds one hop delay.
- **Master:** Processes packet. Sends PALs and output packets to the stable storage and output logger. Adds processing + PAL generation and transmission delay.
- **Output Logger:** Buffers outgoing packets until the vector clocks increment to the per-packet vector clock value. Generally adds the one hop latency to assert that PALs have been stored.

Discussion

- Middlebox application code has to be annotated. Modulo that, the solution is generic.
- Performance numbers show feasibility of approach.
- The Input Logger, Stable Storage and Output Logger are now the points of failure.
- Causal consistency work in databases has more depth, and covers the "novelty" presented here. The application of causal consistency to middlebox code, however, might be novel.

Causal consistency work

- Don't Settle for Eventual: Scalable Causal Consistency for Wide-area Storage with COPS
- Stronger semantics for low-latency geo-replicated storage
- Orbe: scalable causal consistency using dependency matrices and physical clocks