CS 425 MP2 Report
GA10 - Sunveg Nalwar (snalwar2) and Sudarshan Shinde (sshinde5)

Design:

We implement a full-membership protocol with two variants that share a common SuspicionManager. Gossip sends periodic UPDATE_BATCH heartbeats to k random peers and, with suspicion enabled, promotes ALIVE->SUSPECT after Tfail of silence and SUSPECT->DEAD after Tcleanup, disseminating updates via gossip and piggyback. PingAck actively probes one peer per period (PING/ACK) and piggybacks membership updates; with suspicion on, missed ACKs create SUSPECT and later DEAD; with suspicion off, we use per-target no-ACK timers to mark DEAD. Joins disseminate via an introducer JOIN_ACK snapshot plus a one-shot push; leaves are explicit LEFT with higher incarnation and immediate fanout. To prevent cold-start storms when turning suspicion on, we initialize last-heard for unseen peers and apply a brief grace window. DEAD entries are Garbage collected after a timeout. Bandwidth is minimized by compact protobuf batches and piggyback in PingAck. We tune Tfail/Tcleanup and tick periods to satisfy the 3s/6s completeness bound while keeping background bytes/sec low.

Protocol period calculation:

Settings we use in experiments: period=300 ms (PING and gossip tick), Tfail=1 s, Tcleanup=1 s, FanoutK=3, GC TTL=5 s, suspicion grace=2 s only at switch-time.

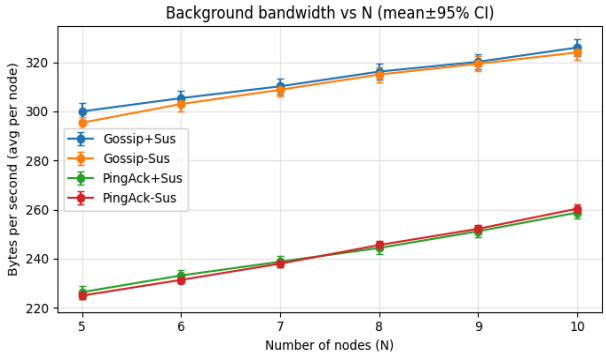
PingAck+Suspicion: A failure is detected when a probed node misses an ACK and remains silent for Tcleanup. Worst-case time to first SUSPECT is ≤ one probe cycle to pick the failed node (≤ period) plus up to Tfail of not hearing (in practice, we probe before Tfail, but bound with period+Tfail), and promotion to DEAD after Tcleanup. So first confirmed failure (DEAD) ≤ period + Tfail + Tcleanup ≈ 0.3 + 1 + 1 = 2.3 s. This meets the 3 s bound.

Gossip+Suspicion: Detection is silence-based. First SUSPECT ≤ Tfail (since we check every period), then DEAD after Tcleanup. So first confirmed failure (DEAD) ≤ Tfail + Tcleanup ≈ 2 s. Dissemination bound: With FanoutK=1 (or piggyback-only in ping), worst-case spread is O(N) periods in adversarial order. Using N=10 and period=0.3 s gives ≤ 3 s to touch all nodes once. In practice we have both piggyback and (optionally) one-shot fanout on joins/DEAD, so observed convergence is typically ≤ 2–3 s. Overall worst-case from failure to all tables updated: ≤ (first DEAD) + (spread) ≈ 2.3 + 3.0 ≈ 5.3 s (ping+suspect), and ≈ 2.0 + 3.0 ≈ 5.0 s (gossip+suspect), satisfying the 6 s completeness requirement. If you run FanoutK=2–3, convergence is faster and less bursty.

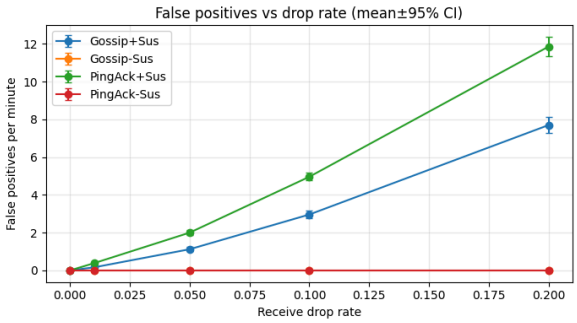
False positives introduction and calculation:

We drop any inbound UDP (including UPDATE_BATCH and ACK) before the handler, simulating loss/delay as required. This is done via a command on CLI - drop [0,1]

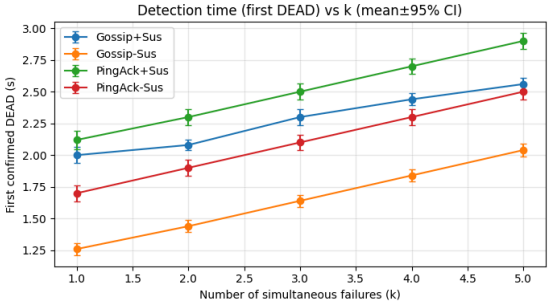
A false positive is any SUSPECT that is later refuted locally without becoming DEAD. Our code logs “SUSPECT <node> …” and “REFUTE <node> (heard again)”; we count a FP when a SUSPECT is followed by REFUTE for the same node and incarnation without an intervening DEAD. Rate reported as FPs/minute over a run (recommended 5–10 minutes per point)



Bandwidth (1a)



False positives (1b)



Detection time (1c)

2d) PingAck benefits more. Suspicion converts a missed ACK into immediate SUSPECT (then DEAD after Tcleanup), avoiding waiting for long silence on a peer that might otherwise not be probed again quickly. Gossip already checks silence periodically for every peer; suspicion mainly shortens the confirmation window, so the marginal gain is smaller than in PingAck.
2e) Gossip–Sus generally detects and converges faster than Ping–Sus because periodic heartbeats/gossip give consistent anti-entropy to all peers, while Ping–Sus relies on piggyback and chance probe paths, making it more sensitive to loss and target selection.