Wei Kang Lim, Neil Patel -- MP2

<u>Design</u>

Network topology

We designed the distributed network topology to accommodate the requirements.

1. There are 5 introducers in the network to account for up-to 4 failures.
2. Each node in the network is arranged in a virtual extended ring.
3. Each node is responsible for sending its heartbeat to 2 immediate successors and 2 immediate predecessors. This is to satisfy the requirement of guaranteed response to failures within 2 seconds, and up-to 4 failures. Consequently, each node is monitored by 4 other successor/predecessors nodes which satisfies the requirement.

Joining & leaving the network
Introducers are responsible for handling joins and leaves to the network. It is understood that introducers are necessary for handling joins, since they must be known before hand. For handling leaves, we also decided to make the five introducers responsible for handling leaves, but it is trivialized to change this to 4 random nodes or successor/predecessor nodes in the system.

When a node joins the network, it sends a join request to one of the five introducers. The first introducer that is part of the network responds to the join request, and sends back the membership list. The node is then responsible for forwarding the updated membership list to its neighbors. This is done by including the membership list as part of the heartbeat message.

Detecting failures
As required by the specifications, failures are detected through heartbeat mechanism. Each node is responsible for sending heartbeats to its 2 immediate predecessors and 2 immediate successors. For failures to be detected under 2 seconds, but also reducing false positives due to changes in the network topology or message loss rate in the network, we decided on 600ms as the heartbeat rate. This reduces false positive rate, by allowing 2-3 heartbeats failed heartbeats before a node is detected as failure.

Membership list dissemination
After failures are detected, the updated membership list is forwarded to the entire group via gossip-style dissemination. To satisfy completeness of updated membership list reaching all members at N=10 after 6 seconds, we chose protocol rounds, b = 2, and round interval = 0.75s, which guarantees that after 8 rounds at t=6s, all nodes will be infected.

Scalability
If the requirements of accuracy and reliability do not change as N grows, the algorithm is scalable to large N. The load from heartbeat is constant at heartbeat_targets = 2, and the load by gossip membership list for time-bound completeness at 6s increases slowly. For N=1,000,000, protocol round, b = 6 is sufficient to satisfy the requirement. If the requirements to

do change with respect to N, there will be a load increase of size O(n) arising from the heartbeat requirement.

Miscellaneous
To marshal the messages into platform-independent formats, we use Google's protocol buffer through a third-party library, ProtoBuf-Net. Integers are marshalled into 128 Varint encoding. The cost of serialization of the messages was reasonably well-sized for our use-case.

Debugging/testing our project was broken down into two phases. The first phase was debugging individual commands. This was set up by having two nodes talk to one another, with remote debuggers attached to each. This helped us resolve any unhandled exceptions or erroneous code. In the second phase, we debugged issues arising from network size N>5, which we briefly used MP1's grep command to filter out the logs. It wasn't as useful as expected, since debugging the result required correlation between log lines to understand the state of the node. It was easier to view the logs on the individual machines to perform the correlation.

Measurements

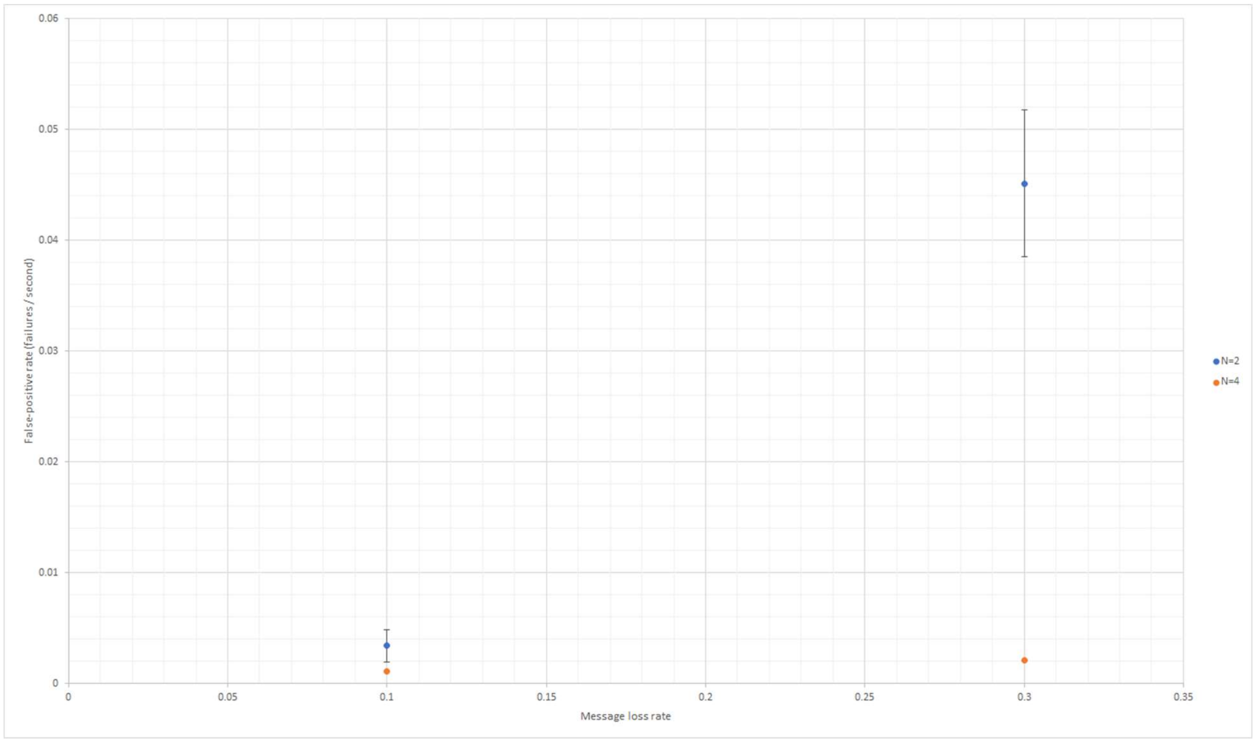
Background bandwidth usage for 4 machines (no membership changes):
9.07 messages / second, 4.19 KB / s (uniform across all machines)
Expected bandwidth for node join: 1 additional packet, O(1)
Expected bandwidth for node leave: 1 additional packet O(1)
Expected bandwidth for node fails: No additional packets

Message loss rate vs. False positive failure detection rate
To measure false positive failure detection rate when message loss is present (3%, 10%, 30%), for N = 2, N = 4, we constructed the network with N = 2, N = 4 machines, and dropped packets at the receiving with an acceptance probability determined by the message loss rate.

The false-positive rate is measured in terms of the time taken for a positive failure to occur for the given network size, instead of the usual definition of ratio of total failures, due to the lack of measurement points. More precisely:

False-positive rate = 1 / time taken for first occurrence of false-positive failure

Message loss rate = 3% was not measured due to time constraints. In our test runs, we did not see failures even after 30 minutes, and decided that it was too time consuming to perform measurements this way.



From the measurements, it is clear that the false positive detection rate is significantly less affected by message loss rate for N = 4 than it is for N = 2. This is due to the increased number of heartbeat monitors, and also due to cooperation between heartbeat monitors. Since each heartbeat monitor also relays its membership list information to other monitors, the heartbeat packet must fail to arrive at each monitor (3 monitors, for N=4 case) for 2 seconds before it is detected as failed.