CS 425 MP2 : Group Membership

Team members: Imani Palmer (ipalmer2), Ramya Nayanaswamy (rpn2)

Programming Language Used: Python

Brief description of logic:

Ping-Ack based failure detection and piggyback based dissemination for membership updates were implemented in MP2.

Failure Detection & Dissemination:

Each active node in the system sends "ping" messages in a round-robin fashion to all other nodes in its membership list. The membership list is shuffled after one round of pinging of all nodes as per SWIM protocol to maintain randomness. If the "ack" message is not received within the predefined timeout period, "pinged" node is detected to have failed the group. To meet the specification of 3s for failure detection and membership update, predefined timeout is set to (2N-1) T = 2.8s (less than 3s), to account for time taken for message formation and other buffer updates that happen in a single protocol period (N is the size of membership list). This ensures time-bounded fail detection.

A failed node is recorded into recent_buffer with respective dissemination count. In our design, dissemination count is c.log N. The fail node is updated to other nodes in the system through piggybacking fail information through "ping" and "ack" messages. The fail information is piggybacked "dissemination count" times. The recent_buffer is garbage collected after dissemination count reaches zero and membership list is updated. The update of the membership list happens at the start of a new round-robin round. This ensures time bounded membership update for the time-bounded fail detection described above (with 3 sec at one node and 6 sec at all other nodes).

New join & Dissemination: There is a fixed introducer node in the system. Whenever a new node wants to join the system, it pings the introducer and gets the membership list. Both the introducer and the new node disseminate the new node information through piggybacking described in previous section.

Node Voluntary leave & Dissemination: The node which decides to leave piggybacks the node leave message to log N existing nodes, while pinging them and exits the group.

Underlying network changes and scaling to N: After each round robin phase, the membership list is updated with new joins/fails/leaving nodes, capturing network changes. Our design scales to N as the message load per protocol period is constant for any network size (N). With many fails/joins in huge systems, scalability could be predictable by limiting the maximum piggybacks in ping/ ack.

MP1 integration: MP1 was integrated and it was useful in checking for log file/debug membership updates of any node from any other node.

Messaging format: UDP is used for all communication. The messages are sent as byte streams. The message has 3-4 characters to indicate message type (join, ping, ack, init etc.). Each ping message has 34 bytes of data and each ack has 4 bytes (without piggyback). Including IPV4 (20) and UDP headers (8).For N= 4, the messages cost is 282 bytes for one ping/ack among them. One with node join (2 piggybacks): 282+75 bytes (for piggyback) + 130(memlist copy) = ~487 bytes. One node fail/crash: ~360 bytes.

False Positive Experiment Setup: Each node is set to ping others 1000 (m) times and each experiment is repeated 5 times. Random ping packets are dropped at one node based on packet loss rate. The false positive identifier is defined as how long does it take show up at sender for various experiments. If false positive occurred at m =66 for a single experiment, then we define false positive index as (1/66). The plot below has averages, standard deviation and confidence interval for false positive indices across 2 and 4 machines. From the figure, it is evident at that smaller packet loss shows at the sender end much later in the communication or rather sporadically. With higher packet loss, node is declared as fail pretty soon. With increase in packet loss, the average time required to detect a false result decreases. The false identifier index increases as it is inverse of detection time.

For N = 2; False positive Identifier, CI: Confidence Interval at 95% Confidence Level

Packet-loss	Avg	STD	CI
3	0.029079	0.031385	0.03
10	0.149648	0.105954	0.09
30	0.39	0.135647	0.12

For N = 4, False positive Identifier for each node observed at sender N0: (Avg, Std Dev, CI)

Packet Loss	N(4,1)	N(4,2)	N(4,3)
3	0.09,0.12, 0.11	0.05,0.033, 0.033	0.08,0.08,0.07
10	0.12,0.072,0.06	0.24,0.161,0.14	0.14,0.0977,0.09
30	0.163,0.056,0.05	0.238,0.157,0.14	0.298,0.1715,0.15