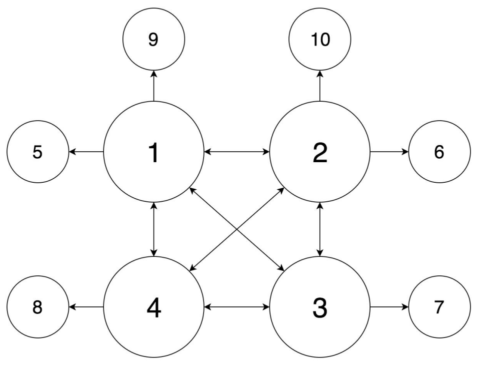
Group 32 - wajid2, lihaolu2

MP2 Report

Algorithm:



Instead of pinging randomly or pinging everyone, we generate ping list based on the membership list. Our topology consists of four core nodes which will put each other into their ping list. In addition, the remaining 6 nodes are leaf nodes. The core nodes will choose the leaf node to put into its ping list according to the algorithm mentioned later in the paragraph. Our system acts like SWIM, randomly pings nodes in the ping list. In case of a leave, join, or failure, every node recalculates the ping list accordingly. The algorithm for generating our topology is: keep a sorted membership list. The first 4 nodes in sorted order are chosen as the core nodes. For core node with index of i in the sorted member list, it adds node j into ping list when j % 4 == i.
For scales of large N, we could generate multiple sub group. Within the group, we could use the topology we showed above. Then, between the groups, we could connect leaf nodes together to act like core nodes. In this case, we only need to keep a O(k) size member list.

Message format: messageType:ip-ts:payloadType_ip-ts_(TTL or #Inc):other payloads….
Here's a example:
0:191.168.1.1-234543234:0_192.168.1.2-2343245234_3:1_192.168.1.3-24524234:....

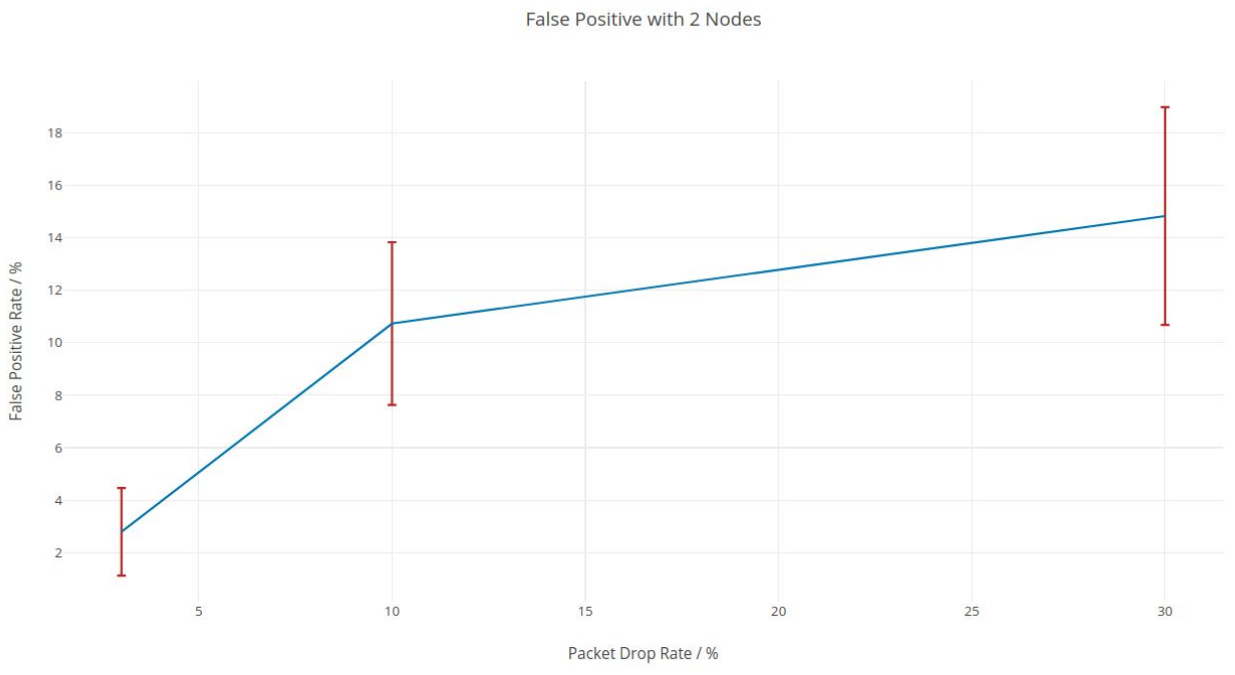
MP1 was helpful when we wanted to grep logs of join, leave, failure from different VMs. When developing the distributed system, it's hard to use some debugger. We count on analyzing all the log we created and pinpoint where the bug hides. With MP1, we can grep the log using keywords like joinMessageReceived, sendingPingMessage to see whether our program was working as expected.

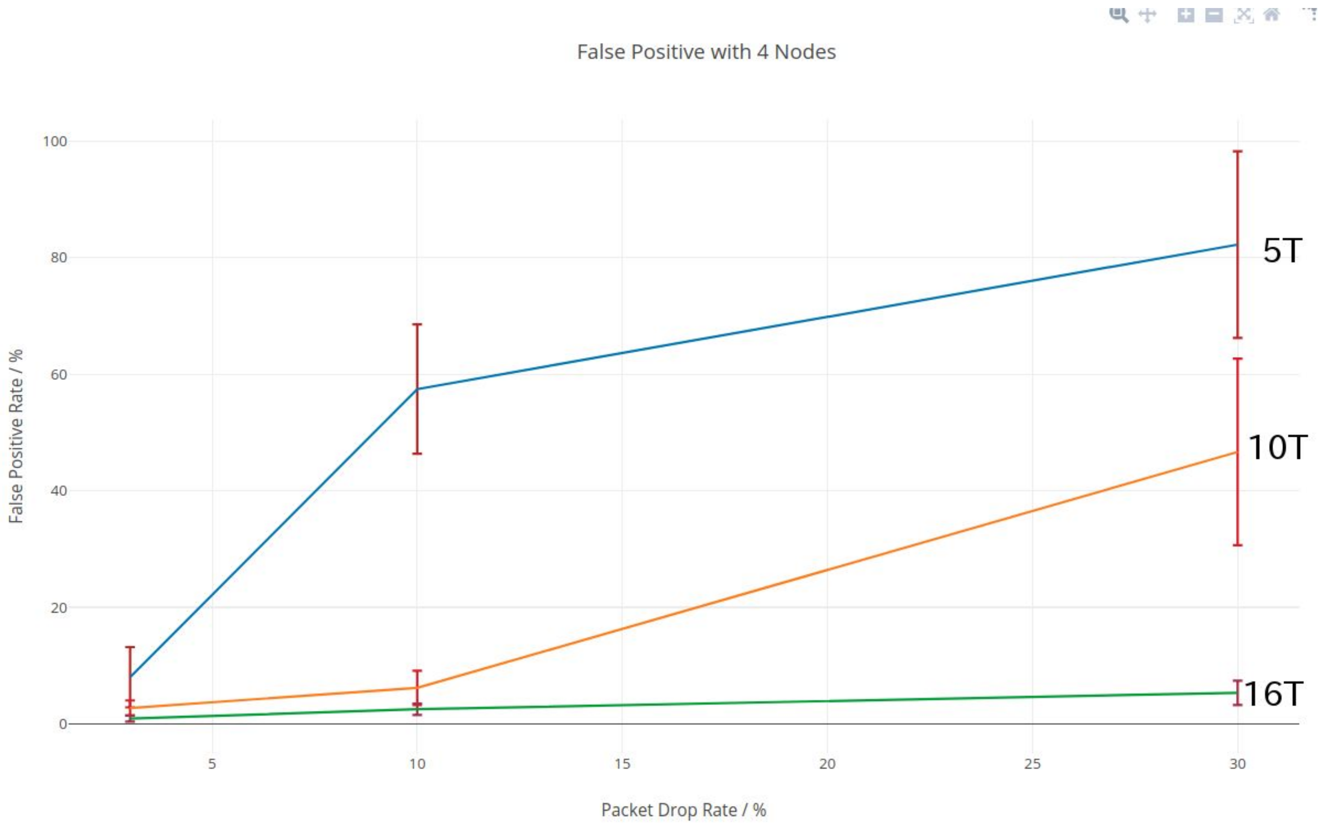
Average bandwidth for 4 nodes

Background (B/s)	Join (B/s)	Leave(B/s)	Failure(B/s)
1090.80	1386.64	1245.72	1277.80

False Positives
For a two node network, our false positive rate for a 3% packet loss was 2.79% and increased as expected with increase in drop rate and was maximum for a 30% drop rate (14.67%).
For a 4 node network, our results were very interesting. By default, in order to reduce average bandwidth usage, we choose a default failure timeout of 2T where T is the default ping time. However we found this to time to give a FP of 10.11% for a 3% loss rate and extremely high FP for 10% and 30 % loss rate.
For our measurements, we calculated the FP for 3 different values of failure timeout, 5T, 10T and 16T. The results show that the performance of our algorithm improves with an increase in failure

timeout. As the graphs show, the false positive increases with increase in packet loss rate as expected. More interestingly, with a high failure timeout, the false positive rate greatly reduces even for high packet loss rate. For example, the FP reduces from 6.17% to 2.51% by increasing failure timeout from 10T to 16T for a packet drop rate of 10%. The graph for 4 nodes displays the above mentioned trend.


False Positive with 2 Nodes
False Positive Rate / %
Packet Drop Rate / %


False Positive with 4 Nodes
False Positive Rate / %
Packet Drop Rate / %
5T
10T
16T