MP2 Report

Group 14 - Mihir Shah (mihirs2) & Manan Patel (manandp2)

## **Overall Design**

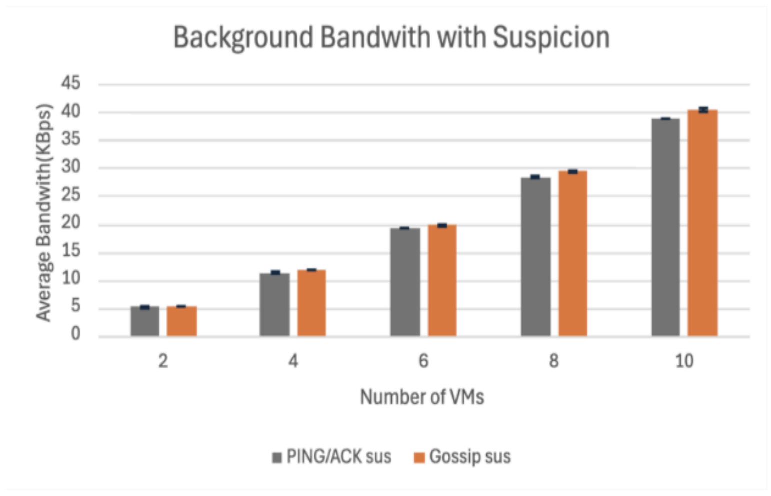
When a machine joins the network, it dials the introducer, who then adds the machine to its local membership list. Going forward, this machine will be included in the introducer's heartbeats/pings, introducing the node to the rest of the network. The introducer replies to the new machine with the network's current protocol and its membership list. The new node will now send heartbeats/pings to the nodes in its membership list.

For Gossip, each VM sends out a heartbeat containing its membership list every 500 milliseconds with a fanout of 2. To choose the node that the gossip will send to in that heartbeat, the node randomly shuffles all nodes other than itself in its membership list and randomly chooses 2. Heartbeats are only sent to machines that have not been marked as "Leaving" or "Failed." Each VM will continuously listen for heartbeats until the machine leaves, fails, or switches protocols. Upon receiving a heartbeat, the VM will update its membership list based on the memberships it received. With suspicion, the machine will act similarly except have an intermediate suspicion "second chance" prior to marking the node as failed. If a node is "healthy" and has not been updated in T_fail, then it will be marked as suspected. If a node is marked as suspected and is updated after that within T_fail, then it will be marked as not suspected. If a suspected node is not updated within T_fail, then it will be marked as failed.

For Ping/Ack, each VM will randomly choose a node (the same method as Gossip) to send a ping to every 250 milliseconds. Once a machine sends a node, it waits for the node to send an ack. If the node does not send an ack within a T_fail (1.3 seconds), then the node will be marked as "Failed". If the node does receive the ack within T_fail, then the machine will continue to send pings normally. With suspicion, if an ack is not received by the timeout, then it will be marked as suspected. Using the randomized choosing logic mentioned above, the suspected nodes will be pinged in the same manner. If a node is not updated by the timeout and it is marked as suspected, then it is marked as failed.

For both protocols, the membership list is asynchronously checked for nodes marked as failed or leaving. Once a failed or leaving node has passed T_cleanup, it is then removed from the machine's membership list. This allows for the failure/leaving to disseminate across the network.

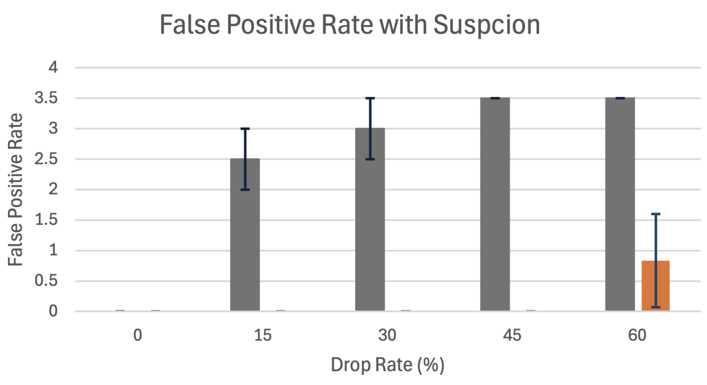
## **Analysis**



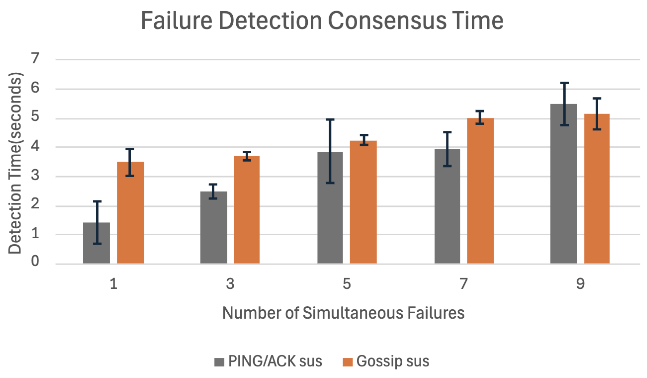
Gossip with suspicion tended to have a slightly higher bandwidth than Ping/Ack with suspicion, especially with a larger number of VMs. Them being similar makes sense since the packet being sent and received is the same for protocols. It also makes sense that Gossip is slightly higher with more VMs because of the fanout being 2.

MP2 Report

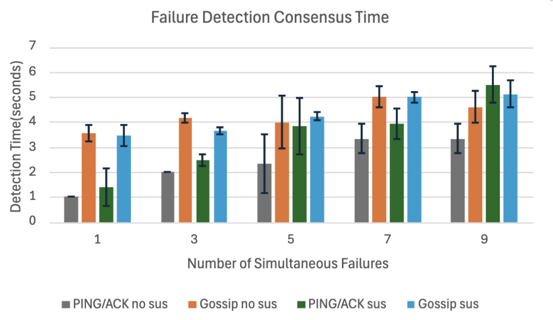
Group 14 - Mihir Shah (mihirs2) & Manan Patel (manandp2)

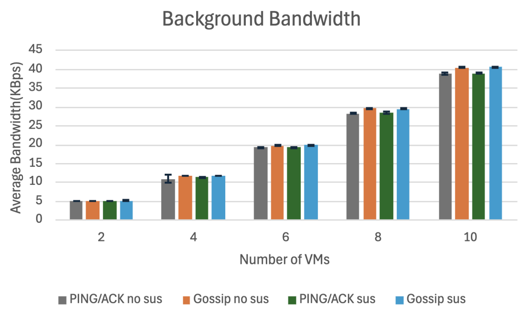


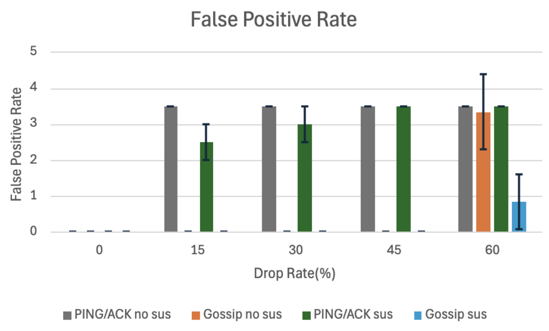
We defined the false positive rate as the number of false positives/number of true positives. Ping/Ack had a much higher false positive rate because it relied on messages from only one machine, whereas gossip could get data from any machine. When the drop rate got very high, even the spread of gossip caused false detections.



Gossip took longer for the last VM to detect the failure compared to Ping/Ack when fewer VMs failed. This makes sense because heartbeats are sent out less frequently than pings. When the number of VMs increased, Ping Ack became slower as the larger fanout in Gossip helped spread the failure quicker.







Without suspicion, it took longer for the last failure to be detected, especially in ping ack. Ping/Ack benefits the most from suspicion to help reduce the number of false positives. If the network is lossy between any two machines, then that machine will immediately be marked as failed, whereas through gossip, the message will reach indirectly. The no-suspicion variants have similar bandwidth usage. Ping/Ack is much faster to detect failures; however, it has more false positives with a lossy network due to the issues mentioned previously. This makes sense because PingAck relies only on the node it is pinging to detect its initial failure. However, Gossip relies on all other nodes in the network to update its membership list.