For each test case, the configuration file is located in configs/TEST_CASE_NAME.da
The ledger files are located at ledger/TEST_CASE_NAME-REPLICA_ID, where REPLICA_ID is the name of the replica that ledger file belongs to. REPLICA_IDs start with “replica0” and count up.

1. leader_setattr
In this test case, a SetAttribute fault injection makes replica0 behave as though it is the leader for rounds 5 through 14, causing it to broadcast proposals out of turn.
Number of replicas: 4
Number of clients: 1

2. syncup_middle_of_chain
In this test cast, replica0 receives no messages from round 6 to round 9. It then syncs up with other replicas when it sees the proposal for round 10
Number of replicas: 4
Number of clients: 1

3. syncup_entire_chain
replica0 receives no messages for the first 6 rounds. It then syncs up when it sees the proposal for round 7. This is different from syncup_middle_of_chain because replica0 falls behind in the very first round.
Number of replicas: 4
Number of clients: 1

4. faulty_dropouts
In this test case, all messages to and from replica0 are dropped in rounds 2 through 6. Then, all the messages to and from replica1 are dropped in rounds 10 through 18. This is an interesting test case because it shows that the sync manager does not allow replicas to fall arbitrarily far behind.
Number of replicas: 4
Number of clients: 1

5. loss_medley
The goal of this test case is to create a long chain of different types of failures. First, all of the proposal messages sent by the leader of round 0 are lost. In round 1, all the timeouts sent by replica0 are lost, and all the timeouts sent to replica1 are lost. In rounds 2 and 5, all of the votes sent to the leader are lost. In round 6, all of the proposals sent by the leader are lost. In round 7, there is a high probability that any timeout message is lost, which leads to a long round ending in a timeout.
Number of replicas: 4
Number of clients: 2

6. heterogeneous_workloads
This test case does not contain any failures, but it has two clients with two different workloads and a larger number of replicas
Number of replicas: 7
Number of clients: 2

7. large_setattr
This test case has two faulty replicas that behave as leaders in overlapping intervals of rounds, submitting proposals out of turn. It also sets the highest vote round and the current round number of one of the faulty replicas to “0” in overlapping intervals of rounds.
Number of replicas: 7
Number of clients: 2

8. normal_operation
In this test case, none of the messages are dropped/delayed and none of the replicas are faulty. There are two clients that send requests in “retransmit” mode.
Number of replicas: 7
Number of clients: 2