<u>Bonus Report</u>

An attempt was made to simulate node failures(kind of hotspotting in different regions) at random on different kind of networks and study the effectiveness of Gossip protocol.

The way failure was induced is as follows:

1. When a gossip actor receives a message , it randomly (either true or false) decides to bring to an end one of it's neighbor nodes.
2. It then find one of it's active neighbor node and propagates the message to it.
3. The process of bringing down a node continues until a threshold of failed nodes is reached, which is defined as a percent of total number of nodes in network and is the varying parameter.

The nature path of failure nodes can be thought of as similar to message propagation. As each failure nodes lies somewhere in neighborhood to the node which is propagating the message. The implementation is such that for killing each node we need 10ms. So we subtract this time from running time.

Following data was recorded for Gossip protocol for network size of 10000

Percent of failure node	Complete Network	Time Grid	Line	Imperfect grid
10 %	0.291s	0.223s	∞	0.9956s
12%	0.225s	∞	∞	2.290s
20%	0.659s	∞	∞	∞
30%	0.441s	∞	∞	∞

∞ denotes the protocol never converges

<u>Key Observations</u>

Following observations can be made based on above table:

- The table above clearly shows that Line is the least fault tolerant and complete grid is the most fault tolerant(it;s intuitive as it's connected to all other nodes.) and imperfect grid it moderately fault tolerant
- For Complete network there seems to be very little influence of node failures
- **If we relax the node termination and let the actor send a message to itself if it cannot find an alive neighbour then the protocol converges for almost all**

topologies(in worst case the actor keeps sending the message to itself until it converges)

The code for above mentioned observations is located in following files:

src/main/scala/main/Main.scala
src/main/scala/actors/GossipV2.scala