CS765 Project Part 2 Report

Indian Institute of Technology, Bombay

Ananya Kulashreshtha : 22B0906
Nitin Singh Patel : 22B0974
Tanish Agarwal : 22B0978

March 23, 2025

Question 1: How does increasing timeout time (T_t) affect block propagation with eclipse attack?

Increasing timeout time (T_t) affects block propagation with eclipse attack as follows :

1. **Blocks Take Longer to Move Around:** In the eclipse attack, the bad nodes (malicious ones) only send the hash of an honest block and don't give the full block even when asked with a "get" request. If T_t gets bigger, honest nodes sit around waiting longer before they give up and ask someone else. Thus, the block propagation would be reduced in the network and the private chain of the malicious nodes gets more time to grow.

2. **Probability of More Forks:** If honest nodes don't get the block quick enough because T_t is big, they might start mining on whatever they've got already. This could make different honest nodes have different blockchains, which means more forks.

3. **Numerical Analysis:** The assignment says block size is 1 MB, and link speed c_{ij} is 100 Mbps for fast nodes and 5 Mbps for slow ones. So, sending a block takes like 80 ms (1 MB = 8 Mbits, 8/100 = 0.08 s) for fast links, or 1.6 s for slow ones. Propagation delay ρ_{ij} is 10–500 ms, so total time is maybe 0.1–2 s. If T_t is way bigger than that, honest nodes are waiting way too long, and the eclipse attack succeeds.

Question 2: Suggest countermeasures to weaken the attack

1. **Smart** *Get Request* **and** *Timeout* **tracking:**

   - **Why It Works:** Honest nodes dynamically learn which peers are likely withholding data, that is which nodes are malicious and reduce reliance on them. This speeds up block propagation and weakens the eclipse attack.
   - **Adaptive Retry Mechanism:** Nodes first retry the *GET* request with a small T_t delay (e.g., 100 ms). If the peer still doesn't respond, they increase the delay exponentially (100 ms $\rightarrow$ 200 ms $\rightarrow$ 400 ms) and simultaneously request from other peers to reduce waiting time.
   - **Multi-Peer Requesting:** Instead of waiting for a single peer to respond, nodes can send *GET* requests to multiple peers at once. This increases the chances of getting the full block quickly.
   - **Malicious Node Identification:** By tracking *GET* requests and their corresponding timeouts, nodes can determine which peers consistently fail to provide blocks. This helps identify malicious nodes and avoid them in future communications.

2. **Network Reformation:**

   - **What:** Once a node speculates that a peer is malicious, it actively reforms its connections by disconnecting from suspected malicious nodes and prioritizing connections with honest peers.
   - **Why It Works:** By constantly refining its peer list, the node increases the probability of interacting with honest nodes, reducing the attack's effectiveness over time. This ensures a healthier network with better block propagation.

3. **Shout Hashes Louder:**

   - **What:** Honest nodes send the hash to everyone right away, with a note saying if they've got the block yet or not. If they haven't, other nodes can ask around for it.
   - **Why It Works:** Makes it harder for malicious nodes to keep blocks secret, since everyone's talking about the hash and looking for the block.

4. **Trust Scores for Peers:**

   - **What:** Give peers a score—like +1 if they send a block, -1 if they don't. Ask high-score peers first and ditch low-score ones.
   - **Why It Works:** Malicious nodes get ignored over time, so eclipse attacks get weaker.

Experiments

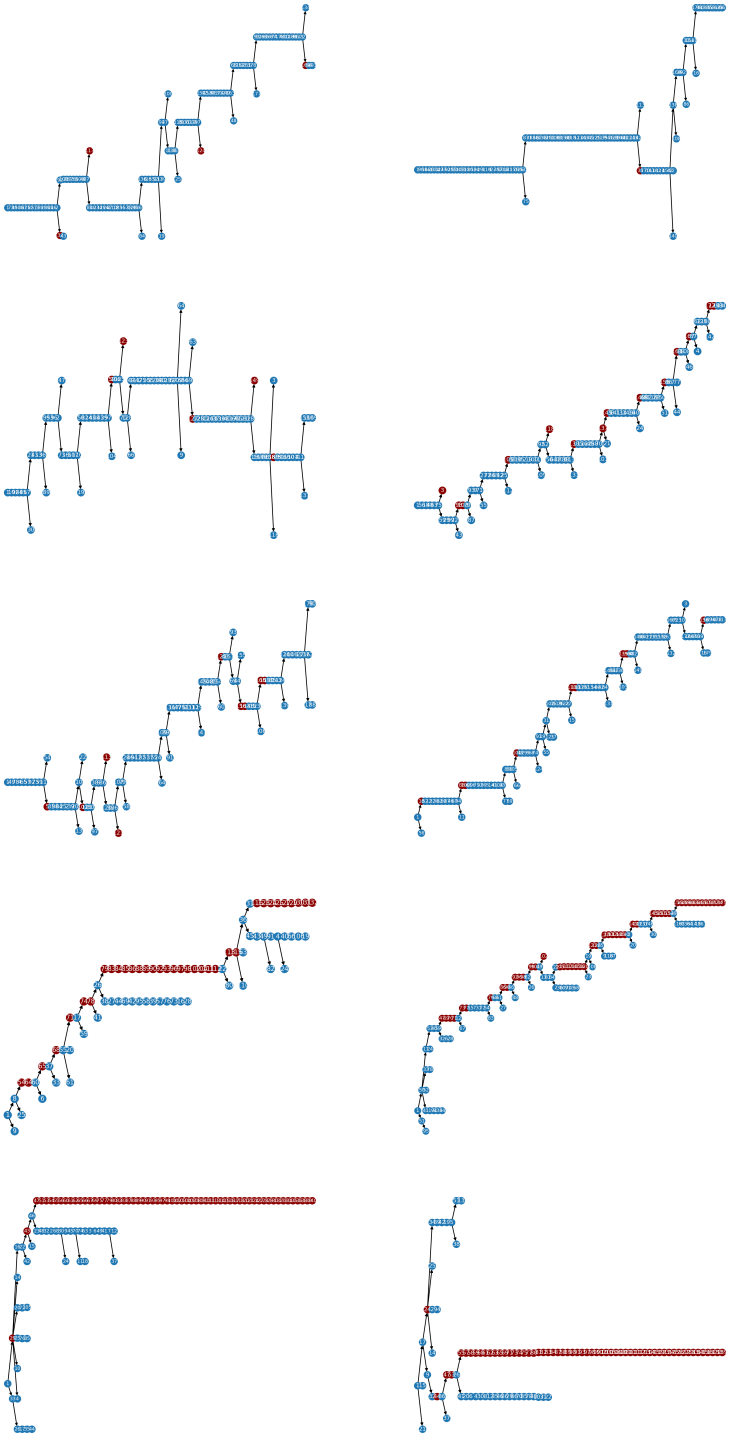

Figure 1: Left with eclipse attack right non-eclipse attack
top to bottom malicious node percent 5,10,15,40,60
timeout 200 μs

- On moving left to right, top to bottom, the ratios we get are:
  - $R1$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total length of longest chain}}$
  - $R2$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total blocks generated by Ringmaster}}$

1.	**R1:** $\frac{1}{94} = 0.01$	**R2:** $\frac{1}{4} = 0.25$
2.	**R1:** $\frac{1}{94} = 0.01$	**R2:** $\frac{1}{1} = 1$
3.	**R1:** $\frac{3}{80} = 0.0375$	**R2:** $\frac{3}{5} = 0.6$
4.	**R1:** $\frac{12}{75} = 0.16$	**R2:** $\frac{12}{14} = 0.857$
5.	**R1:** $\frac{7}{78} = 0.089$	**R2:** $\frac{7}{9} = 0.778$
6.	**R1:** $\frac{9}{84} = 0.11$	**R2:** $\frac{9}{9} = 1$
7.	**R1:** $\frac{26}{44} = 0.59$	**R2:** $\frac{26}{35} = 0.743$
8.	**R1:** $\frac{36}{71} = 0.507$	**R2:** $\frac{36}{49} = 0.735$
9.	**R1:** $\frac{2}{23} = 0.087$	**R2:** $\frac{2}{60} = 0.033$
10.	**R1:** $\frac{3}{28} = 0.107$	**R2:** $\frac{3}{59} = 0.05$

- We see that on increasing the malicious node percent, in general, the number of malicious blocks increases. This is expected because hashing power of the ringmaster increases with increasing malicious node percent.
- Moreover, on increasing malicious node percent, we see the malicious node chain becoming longer and straight. This implies that honest nodes are unable to catch up the private chain of the ringmaster and these private blocks were released at the end of the simulation (Actual selfish mining attack never happened).
- Also, from left to right, on disabling eclipse attack, we see honest blocks propagating a bit more so forks are increased and also the private chain is a bit disrupted.
- With eclipse attack blockchains are more fragmented - multiple small forks and isolated paths.
- Without eclipse attack blockchains are tighter and deeper, showing better agreement among honest nodes.
- Eclipse attack not only delays blocks but disrupts consensus formation.

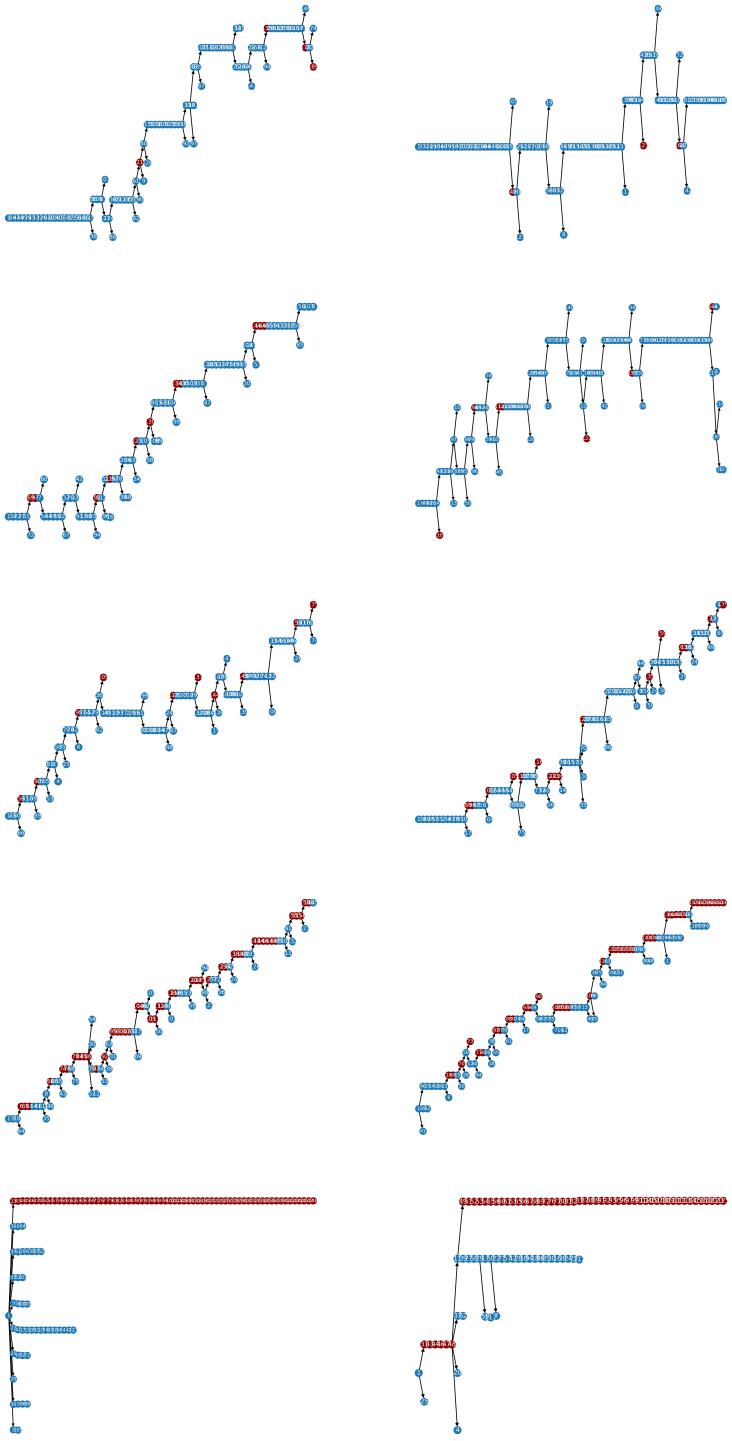

Figure 2: Left with eclipse attack right non-eclipse attack
top to bottom malicious node percent 5,10,15,40,60
timeout 2000 μs

- On moving left to right, top to bottom, the ratios we get are:
  - $R1$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total length of longest chain}}$
  - $R2$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total blocks generated by Ringmaster}}$

1.	**R1:** $\frac{3}{80} = 0.0375$	**R2:** $\frac{3}{4} = 0.75$
2.	**R1:** $\frac{2}{85} = 0.0235$	**R2:** $\frac{2}{3} = 0.667$
3.	**R1:** $\frac{11}{70} = 0.157$	**R2:** $\frac{11}{11} = 1$
4.	**R1:** $\frac{4}{88} = 0.045$	**R2:** $\frac{4}{7} = 0.571$
5.	**R1:** $\frac{7}{75} = 0.093$	**R2:** $\frac{7}{10} = 0.7$
6.	**R1:** $\frac{12}{74} = 0.162$	**R2:** $\frac{12}{16} = 0.75$
7.	**R1:** $\frac{45}{74} = 0.608$	**R2:** $\frac{45}{46} = 0.978$
8.	**R1:** $\frac{31}{68} = 0.456$	**R2:** $\frac{31}{41} = 0.756$
9.	**R1:** $\frac{0}{15} = 0$	**R2:** $\frac{0}{66} = 0$
10.	**R1:** $\frac{6}{30} = 0.2$	**R2:** $\frac{6}{54} = 0.111$

- We observe similar trend on increasing the malicious node percent but since timeout is a bit more so we observe that honest nodes have less propagation in the network when eclipse attack is enabled. Non-eclipse attack blockchain trees are similar to those of timeout value 200 μs.
- From left to right, we observe similar trend that honest blocks are able to disrupt the malicious chains a bit more.

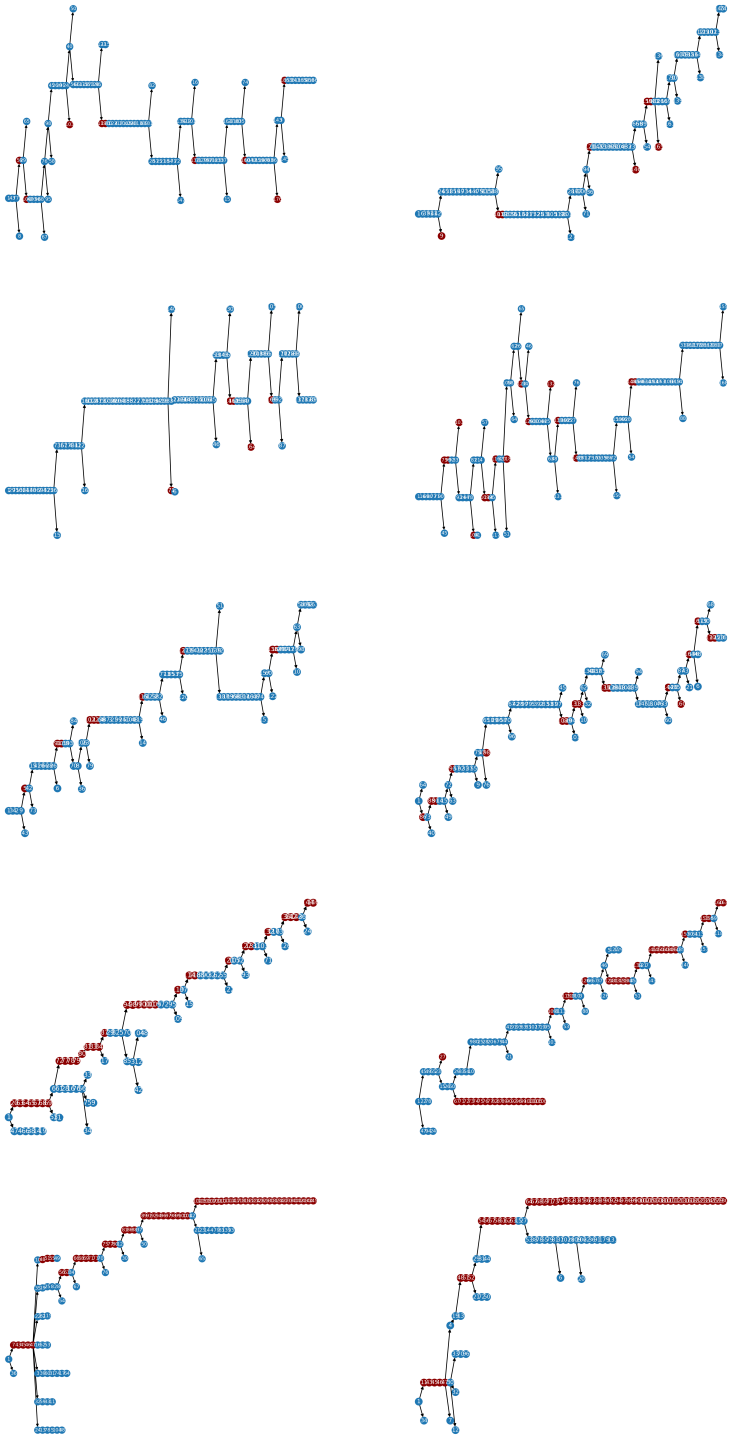

Figure 3: Left with eclipse attack right non-eclipse attack
top to bottom malicious node percent 5,10,15,40,60
timeout 20000 μs

- On moving left to right, top to bottom, the ratios we get are:
  - $R1$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total length of longest chain}}$
  - $R2$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total blocks generated by Ringmaster}}$

1.	**R1:** $\frac{7}{86} = 0.081$	**R2:** $\frac{7}{9} = 0.777$
2.	**R1:** $\frac{5}{81} = 0.062$	**R2:** $\frac{5}{8} = 0.625$
3.	**R1:** $\frac{3}{89} = 0.0337$	**R2:** $\frac{3}{5} = 0.6$
4.	**R1:** $\frac{11}{84} = 0.131$	**R2:** $\frac{11}{15} = 0.733$
5.	**R1:** $\frac{11}{76} = 0.144$	**R2:** $\frac{10}{11} = 0.909$
6.	**R1:** $\frac{15}{73} = 0.205$	**R2:** $\frac{15}{17} = 0.882$
7.	**R1:** $\frac{32}{54} = 0.592$	**R2:** $\frac{32}{34} = 0.941$
8.	**R1:** $\frac{19}{64} = 0.297$	**R2:** $\frac{19}{41} = 0.463$
9.	**R1:** $\frac{28}{47} = 0.596$	**R2:** $\frac{28}{56} = 0.5$
10.	**R1:** $\frac{15}{38} = 0.394$	**R2:** $\frac{15}{53} = 0.283$

- Again, we see similar trends on moving top to bottom. The chain lengths are increasing as the malicious node percent increases which is expected.
- Moreover, from left to right, the trend is similar. The timeout is 20000 μs so, we see less honest blocks as compared to earlier cases where timeout was less.

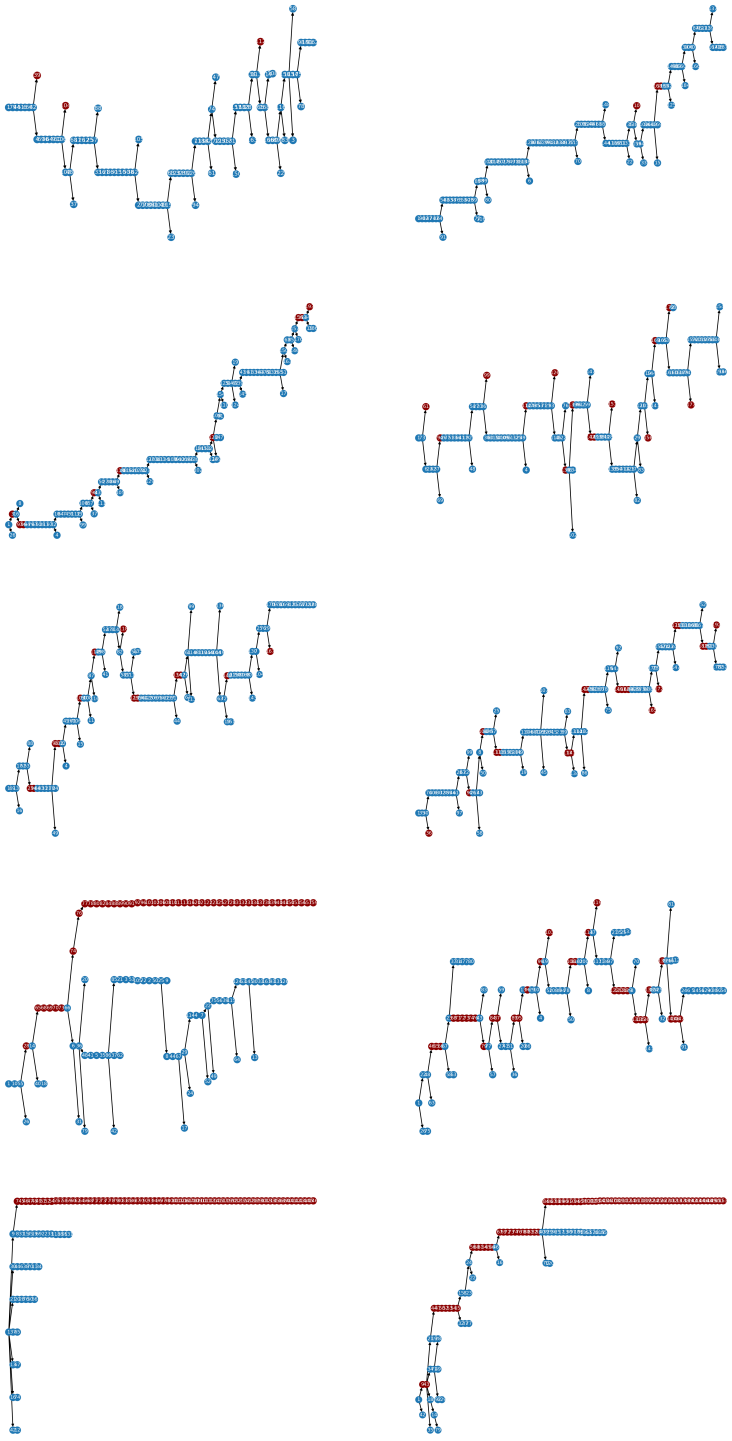

Figure 4: Left with eclipse attack right non-eclipse attack
top to bottom malicious node percent 5,10,15,40,60
timeout 200000 μs

- On moving left to right, top to bottom, the ratios we get are:
  - $R1$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total length of longest chain}}$
  - $R2$ denotes $\frac{\text{Ringmaster's blocks in longest chain}}{\text{Total blocks generated by Ringmaster}}$

1.	**R1:** $\frac{0}{76} = 0$	**R2:** $\frac{0}{3} = 0$
2.	**R1:** $\frac{2}{89} = 0.022$	**R2:** $\frac{2}{3} = 0.667$
3.	**R1:** $\frac{8}{83} = 0.096$	**R2:** $\frac{8}{9} = 0.889$
4.	**R1:** $\frac{7}{86} = 0.081$	**R2:** $\frac{7}{14} = 0.5$
5.	**R1:** $\frac{11}{86} = 0.128$	**R2:** $\frac{11}{13} = 0.846$
6.	**R1:** $\frac{16}{91} = 0.176$	**R2:** $\frac{16}{21} = 0.762$
7.	**R1:** $\frac{6}{48} = 0.125$	**R2:** $\frac{6}{48} = 0.125$
8.	**R1:** $\frac{31}{71} = 0.437$	**R2:** $\frac{31}{33} = 0.94$
9.	**R1:** $\frac{0}{70} = 0$	**R2:** $\frac{0}{15} = 0$
10.	**R1:** $\frac{26}{79} = 0.33$	**R2:** $\frac{26}{49} = 0.53$

- Again, we see similar trends on moving top to bottom. The chain lengths are increasing as the malicious node percent increases which is expected.
- Moreover, from left to right, the trend is similar. The timeout is 200000 μs so, we see less honest blocks as compared to earlier cases where timeout was less.

Notes

- R1 and R2 are computed upto the end of simulation. This means that in cases where 40% and 60% malicious nodes give ratio as 0, it means that they never got a chance to attack in the first place. The blockchain trees shown are after the simulation ends assuming private blocks were broadcasted hypothetically.
- We observed some nodes which have very short blockchain tree. This is probably because it didn't get the honest block on which malicious blocks were mined so it may happen that it considers all the further blocks as orphaned and doesn't add them to its tree.
- We also ran the simulation for malicious node percent 100% this resulted in just a straight malicious chain (all red) as expected.

R1 and R2 vs Malicious Node Percentage

To complement our timeout analysis, we also computed how R1 and R2 change as the percentage of malicious nodes increases (5% to 60%) across different timeout configurations.

Trends in R1

- R1 increases consistently with malicious node percentage, especially under the Eclipse attack.
- Without Eclipse, R1 grows more gradually and sometimes saturates, indicating honest nodes can better resist selfish mining.
- The gap between Eclipse and non-Eclipse R1 grows with timeout, reinforcing the Eclipse attack's reliance on delayed block propagation.

Trends in R2

- R2 under Eclipse is often near 1.0, showing that most malicious blocks are successfully inserted into the longest chain.
- Without Eclipse, R2 remains notably lower, meaning many attacker-generated blocks are eventually discarded due to forks or delays.
- The discrepancy between Eclipse and non-Eclipse cases widens with increasing timeout and malicious power.

Summary

The above analysis shows that both R1 and R2 are sensitive not only to timeout settings but also to the strength of the attacker (in terms of node percentage). The Eclipse attack dramatically increases the success rate of selfish mining, especially when malicious power is high and timeout values are large. Disabling Eclipse restores resilience to the network, reducing the inclusion rate of malicious blocks.

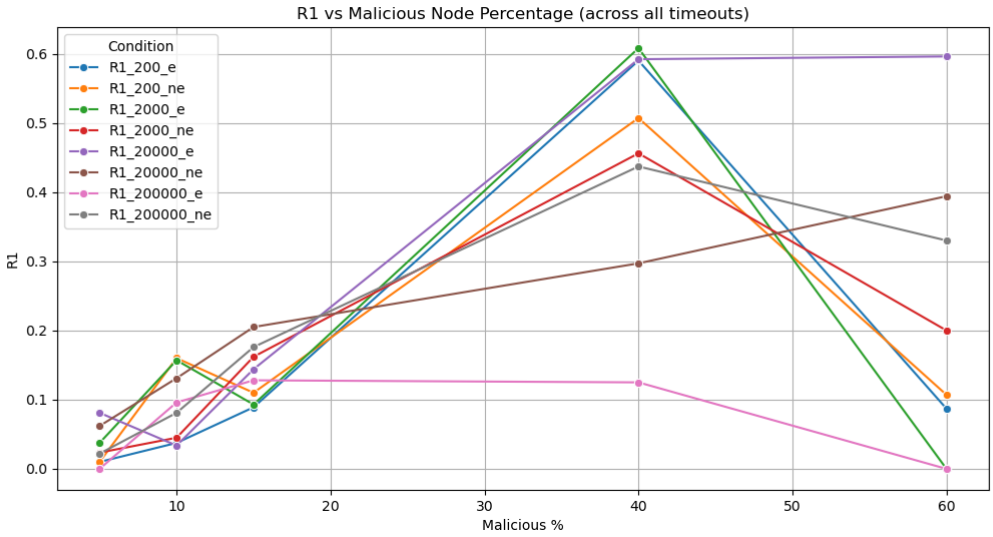

Figure 5: R1 vs Malicious Node Percentage across all timeouts and Eclipse configurations

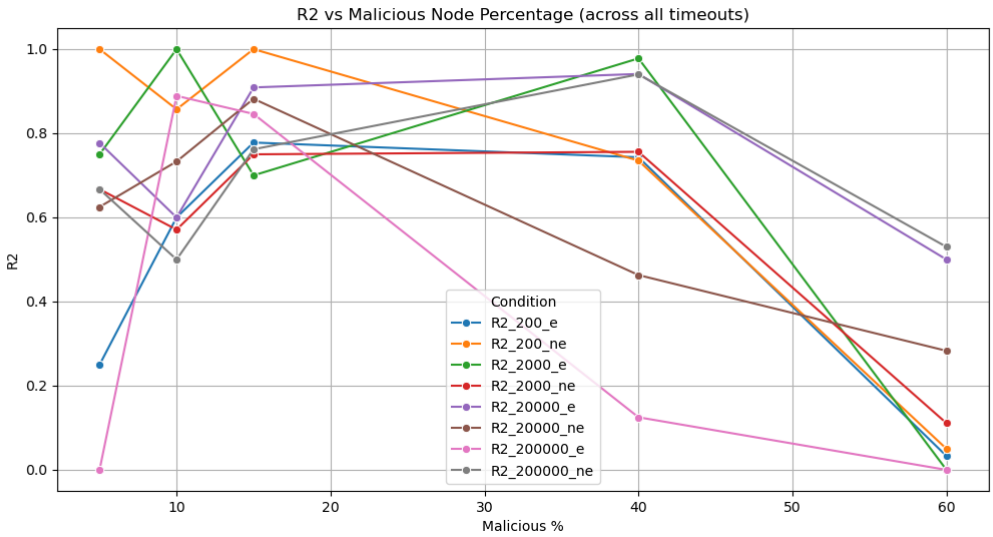

Figure 6: R2 vs Malicious Node Percentage across all timeouts and Eclipse configurations