Simulating Selfish Mining Attacks on P2P Cryptocurrency Network

Shalabh Gupta	180050095
Vrinda Jindal	180050120
Pratyush Agarwal	180050078

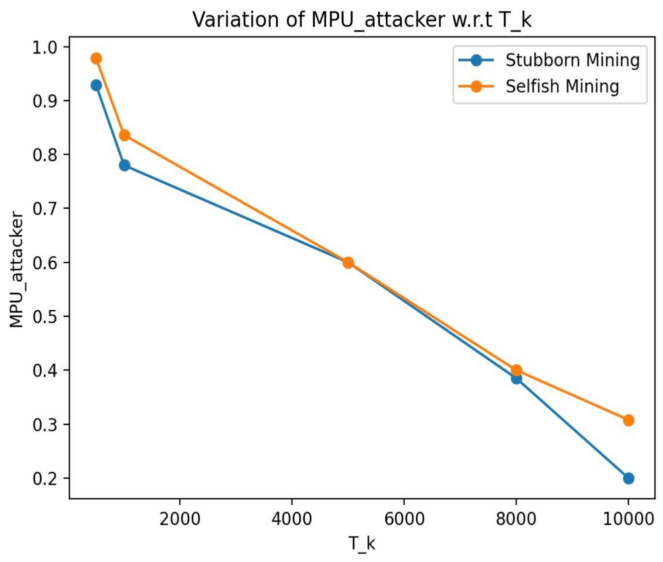
Observations and Inferences

1) Variation of MPU ratios with Mean of T_k (in milliseconds)

Parameters

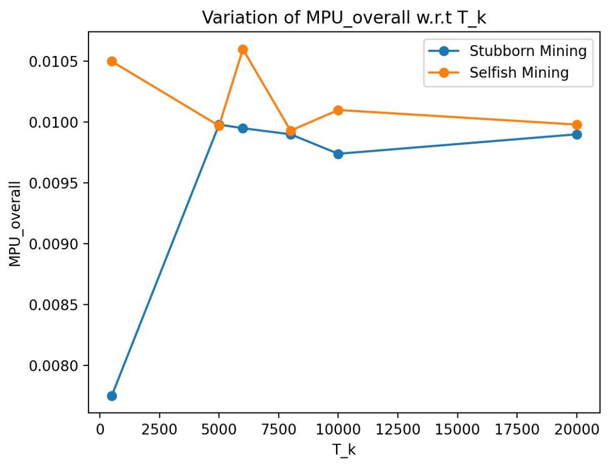
- T_tx : 10s
- Zeta (Attacker's neighbour fraction) : 25%
- N (number of nodes) : 100
- Simulation time : 100 seconds
- Adversary hashing power : 40%

MPU_adversary



Observation : MPU of attacker decreases as mean block mining time increases.This is because the attacker will mine less blocks having less lead on avg and hence the attack won't be as strong as when mean of T_k is less.

MPU_overall



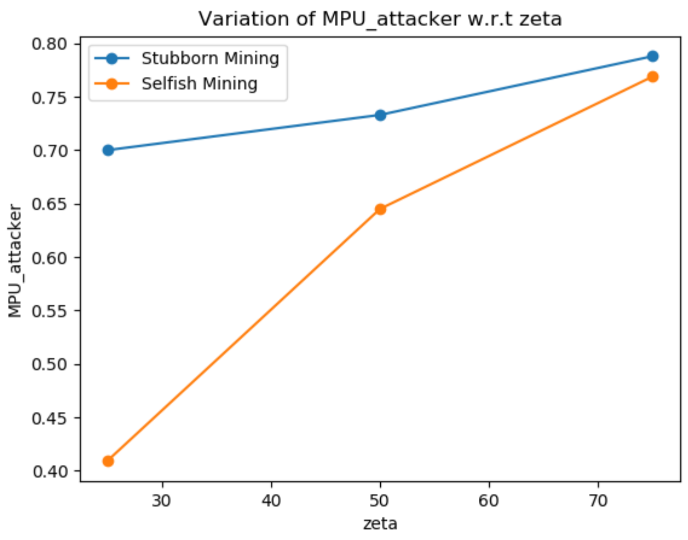
Observation: On varying mean of T_k, all nodes are affected the same i.e. the same factor change in hashing power of all the nodes takes place. This is why, on average, the MPU_overall remains the same overall in both the attacks with small up-down variations.

2) Variation of MPU ratios with zeta (Neighbour fraction of Adversary)

Parameters

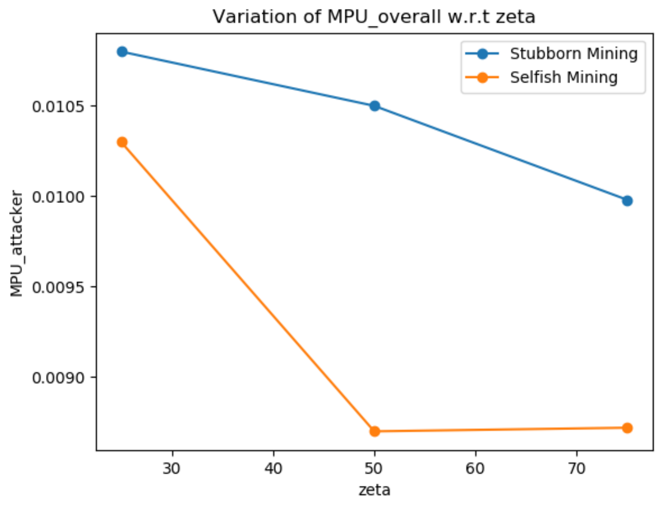
- T_tx : 10s
- Mean of T_k : 2s
- N (number of nodes) : 100
- Simulation time : 100 seconds
- Adversary hashing power : 40%

MPU_adversary



Observation : MPU_attacker increases with zeta, since the attacker has more neighbors, the chances of attack being successful are more as the released blocks reach the neighbors much faster and thus the blocks have a higher chance of getting in the main chain, hence mpu_attack increases.

MPU_overall



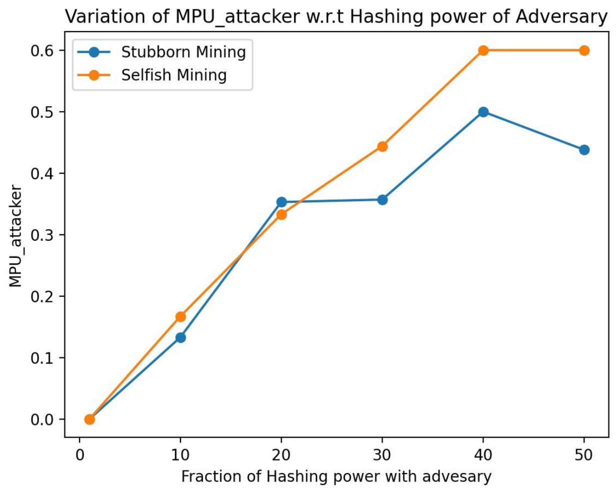
Observation : MPU_overall is seen to be constant or decreasing with zeta in general, which also seems correct intuitively. With more attacker blocks getting into the chain, many more honest blocks will be wasted, reducing MPU_Overall.

3) Variation of MPU ratios with attacker hashing power

Parameters

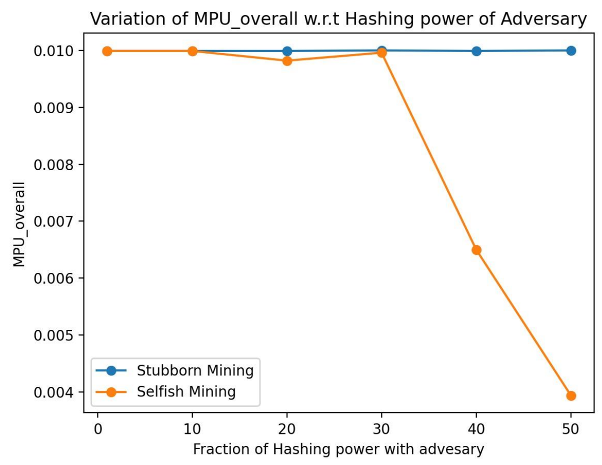
- T_tx : 10s
- Mean of T_k : 5s
- Zeta (Attacker's neighbour fraction) : 25%
- N (number of nodes) : 100
- Simulation time : 100 seconds

MPU_adversary



Observation : As the adversary hashing power increases, the number of blocks mined by adversary increases and the attack becomes stronger (leading to attacker always having a private lead over honest miners), hence MPU_attacker increases leading to a larger fraction of blocks mined by attacker to be included in the main chain.

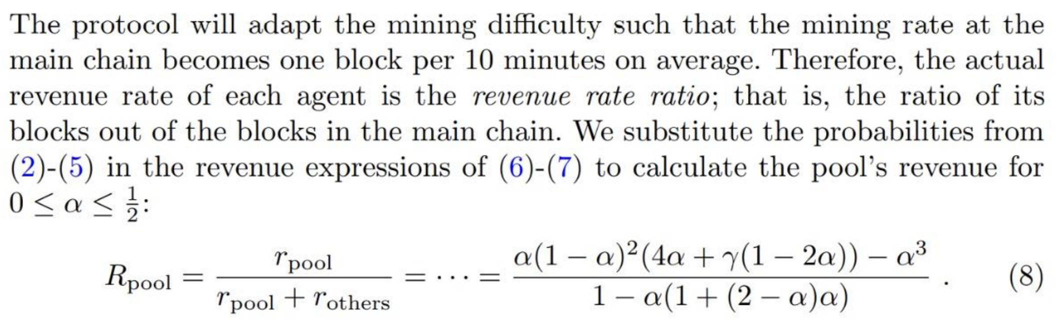
MPU_overall



Observation: With increasing adversary hashing power, MPU_overall is seen to be decreasing or remaining constant generally. This is because with more hashing power, the attacker will be able to maintain a strong attack, wasting most of the honest blocks, and reducing overall MPU.

Fraction of attacker nodes in the main chain vs hashing power of attacker

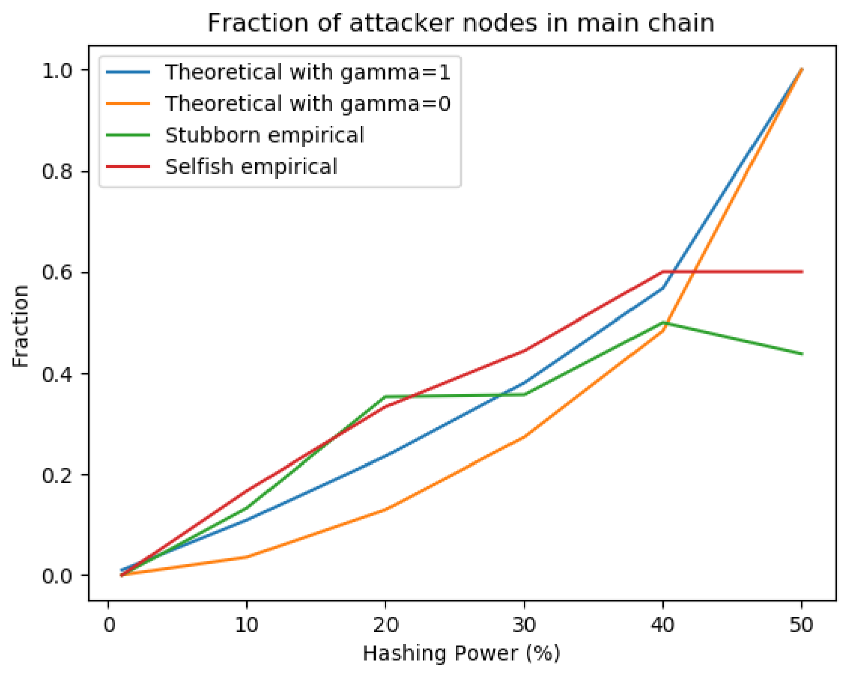
The theoretical value is found by using the formula given below (from the paper) at gamma=0 and gamma=1.



The protocol will adapt the mining difficulty such that the mining rate at the main chain becomes one block per 10 minutes on average. Therefore, the actual revenue rate of each agent is the *revenue rate ratio*; that is, the ratio of its blocks out of the blocks in the main chain. We substitute the probabilities from (2)-(5) in the revenue expressions of (6)-(7) to calculate the pool's revenue for $0 \leq \alpha \leq \frac{1}{2}$:

$$R_{\text{pool}} = \frac{r_{\text{pool}}}{r_{\text{pool}} + r_{\text{others}}} = \cdots = \frac{\alpha(1-\alpha)^2(4\alpha + \gamma(1-2\alpha)) - \alpha^3}{1 - \alpha(1 + (2-\alpha)\alpha)} \,. \qquad (8)$$

Image Source: Majority is not Enough



Observation : As expected the fraction of attacker blocks in the main chain increases as the hashing power of the attacker increases. We also note that the fraction (empirical and theoretical) is greater than the attacker hashing power indicating that the attack is successful (More blocks getting in the main chain than any peer in the network is entitled to).

For eg at hashing %ages 1,10,20,30,40,50 , the following are the fractions (empirical) :

Percentage	1	10	20	30	40	50
Selfish Fraction	0	0.167	0.333	0.444	0.6	0.6
Stubborn Fraction	0	0.133	0.353	0.357	0.5	0.438

Parameters for empirical :

- T_tx : 10s
- Mean of T_k : 5s
- Zeta : 25%
- N : 100
- Simulation time : 100

Final Submission Details

- Source code : big.cpp
- Details to run the code: README.txt
- Design doc: Modular Code Flow in DesignDoc.jpg
- Report.pdf