Assignment 1

200050039 : Ebrahim Sohail Haris
200050089 : Nishant Singh
2000500116 : Chetan Hiraman Rathod

March 2023

Contents

1 Introduction **2**

2 Selfish Mining Attack **2**

3 Analysis of Selfish Mining Attack **2**
3.1 Ratio of adversary node in longest chain vs hashing power 2
3.2 $\text{MPU}_{node_{avg}}$ vs Hashing Power 3
3.3 $\text{MPU}_{node_{overall}}$ vs Hashing Power 3

4 Stubborn Mining **4**
4.1 Ratio of adversary node in longest chain vs hashing power 5
4.2 $\text{MPU}_{node_{avg}}$ vs Hashing Power 6
4.3 $\text{MPU}_{node_{overall}}$ vs Hashing Power 7

5 Conclusion **7**

1 Introduction

We have previously simulated how a blockchain propagates in real life. But we know that with a technology such as blockchain technology which is all about security, it becomes extremely crucial to analyse what are the types of the attack that is possible. In this assignment we are going to analyse two such famous attacks using our previously created simulation. The first being the **selfish mining** attack proposed by Eyal and Sirer in 2014, and the second being the **stubborn mining** attack which is an extension of the selfish mining attack.

2 Selfish Mining Attack

Normally when a node receives a block he sends it to all his peers and starts mining on top of his longest chain. But in a selfish mining attack a miner wants to increase his overall profit by keeping his chain private and releasing appropriate blocks at appropriate time. When a selfish miner creates a block, instead of releasing it he will try to keep increasing this chain by mining on top of it. There are different cases how the selfish miner will behave. These cases are as follows:

- **Case 1:** When Selfish miner just started mining or he lost to make his private chain(chain on which he was mining secretly) as longest chain. In this case he will just start mining on the top of the longest chain.
- **Case 2:** When selfish miner gets a block on honest chain and the chain length of private chain of selfish miner gets equal to that of honest chain. In this case the selfish miner will release his block and will not send the received block to the peers, because he wants to diverge(get more hashing power on his block so that his chain will win) the hashing power of the honest miners.
- **Case 3:** When the selfish miner is 2 blocks ahead and the honest miner mines new block, here in this case the selfish miner will release/broadcast his 2 blocks so that he can discard the longest chain and make his private chain as longest chain. In this case also the selfish miner will not send the received block to his peers.
- **Case 4:** When the selfish miner is more than 2 block ahead of honest chain. In this case the selfish miner will release his 1 block, and will not send the received block to peers.

We could very easily implement this on top of our previous simulation as we only need to change the receive_block function to handle differently for selfish and honest miners. We now do some analysis of the selfish mining attack using our simulation.

3 Analysis of Selfish Mining Attack

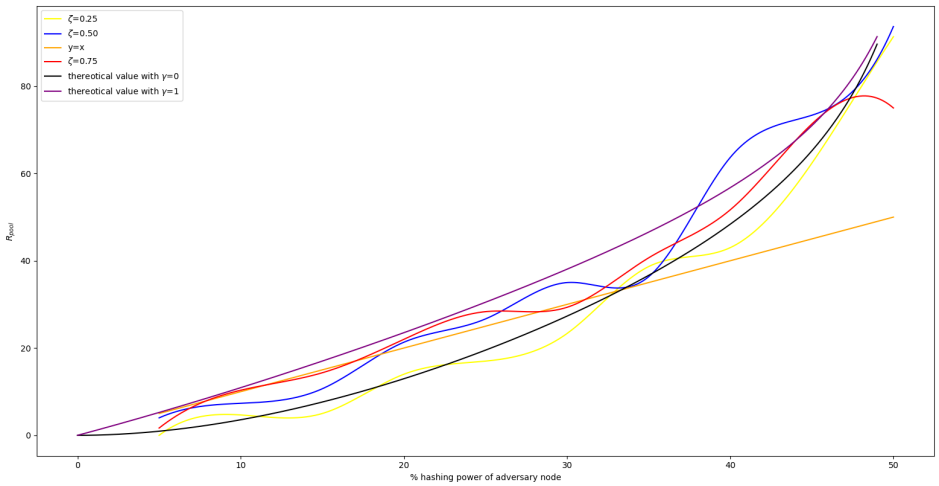
Simulating the attack and visualising in is indeed helpful, but a more vigorous analysis is needed on these types of attacks in safeguarding our blockchain. We also note the values of the 2 ratios we are asked to find, MPUnodeavg and MPUnodeoverall.

3.1 Ratio of adversary node in longest chain vs hashing power

We now begin to analyse when does such an attack become profitable for an adversary. To achieve this it is important to plot the ratio of adversary node in longest chain(R_{pool}) vs hashing power(α). We also compare our results with theoretical values obtained by Eyal and Sirer. The relation between R_{pool} and α given by Eyal and Sirer is

$$R_{pool} = \frac{\alpha(1-\alpha)^2(4\alpha + \gamma(1-2\alpha)) - \alpha^3}{1 - \alpha(1 + (2-\alpha)\alpha)}$$

Now we plot the graph for $\gamma = 0$ and $\gamma = 1$ and we also show the variation of R_{pool} with α using our simulation in the same plot. We conduct our simulation for three different values of ζ-25%, 50% and 75% where ζ represents the fraction of honest nodes the adversary is connected to.



From the above plot we can conclude that indeed our simulation is working well and is in accordance with the theoretical values mentioned above. We can see that for ζ=25% the R_{pool} resembles more closely with the theoretical values with $\gamma = 0$ and as we increase ζ the plot resembles closely with that of γ=1. We note that when the adversary is connected to lower number of honest nodes then it requires about 33% of the total hashing power to actually have a profit than how much he actually deserves. But as he is connected to more and more peers he can diverge the computing power of the honest miners more and it becomes profitable for him to mine selfishly even when he has lower than 33% of the total hashing power

3.2 MPU$_{node_{avg}}$ vs Hashing Power

The MPU$_{node_{avg}}$ is defined as

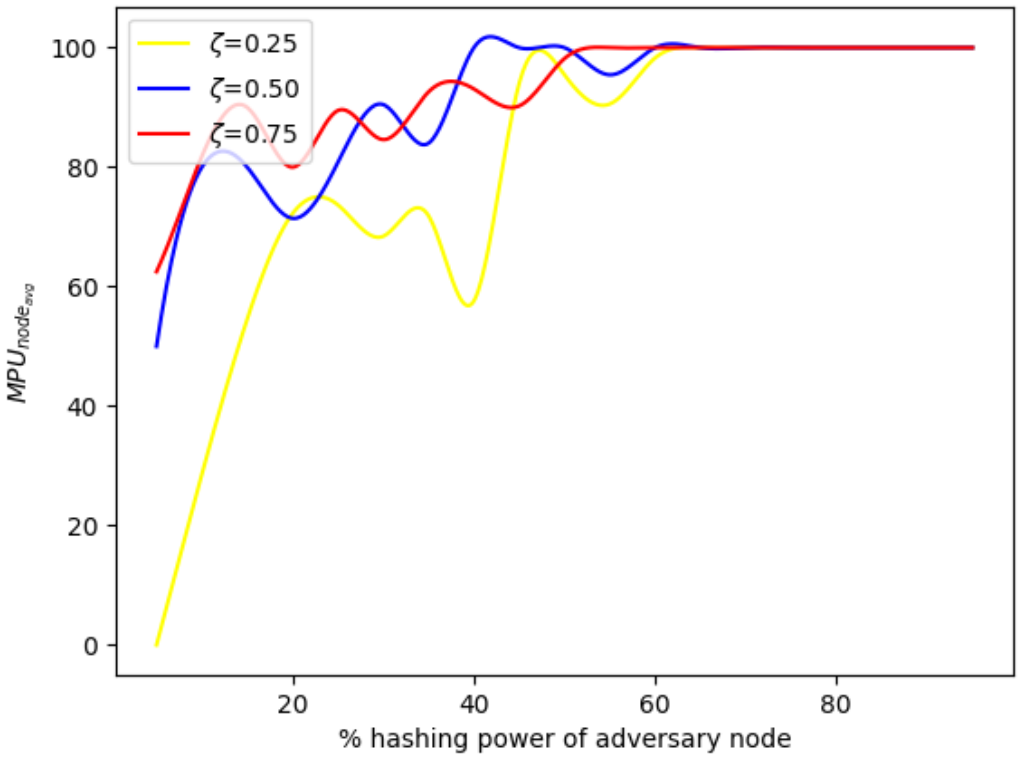
$$MPU_{node_{avg}} = \frac{\textbf{Number of block mined by an adversary in main chain}}{\textbf{Total number of blocks mined by an adversary}}$$

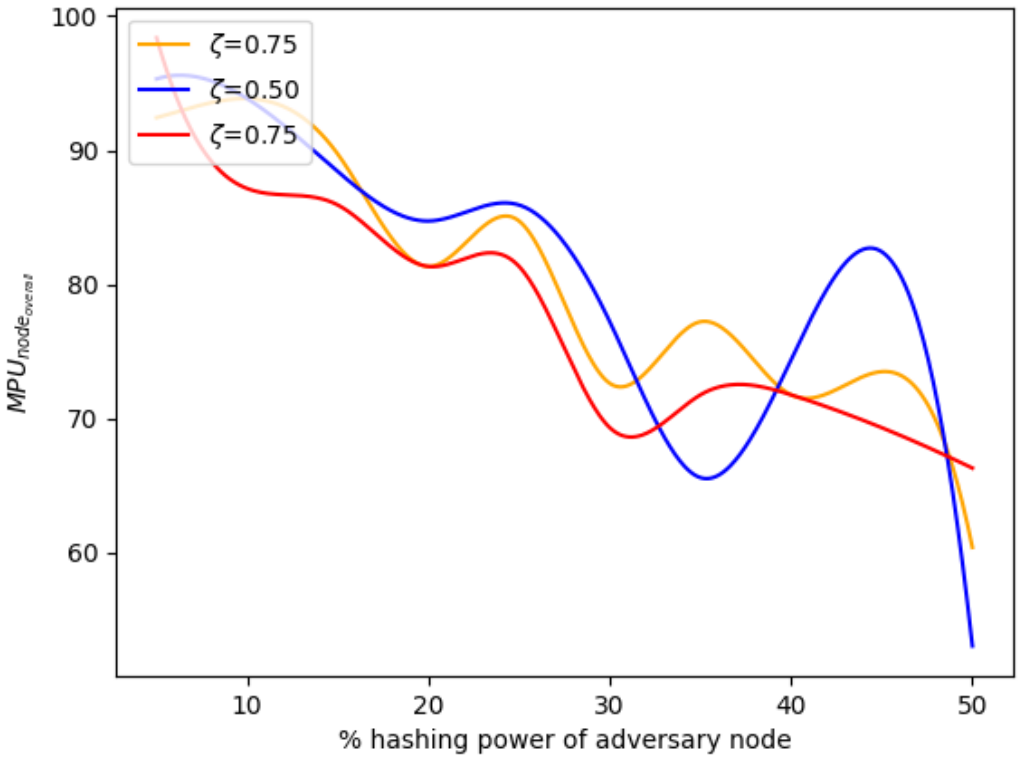
We plot the MPU$_{node_{avg}}$ vs the hashing power of the adversary for ζ = 25%, 50% and 75%.

Let us dig deep as to what the plot means. We can see as the hashing power increases the ratio becomes equal to 1. This means that all the blocks mined by selfish miners gets into the longest chain as expected but as the hashing power becomes lower we can see that the ratio increases significantly with increase in ζ this is also expected as, when the hashing power is low, There is more probability for there to be a race condition as given by our case 2 in selfish mining and this race condition is influenced by γ which in turn depends on ζ

3.3 MPU$_{node_{overall}}$ vs Hashing Power

The graph of MPU$_{node_{overall}}$ is now plotted along hashing power. We get to see that this ratio is 1 when the hashing power is low, because the adversary cannot make any fork, but as the hashing power increases the ratio becomes almost 50% as the adversary creates an alternate block for each honest block.



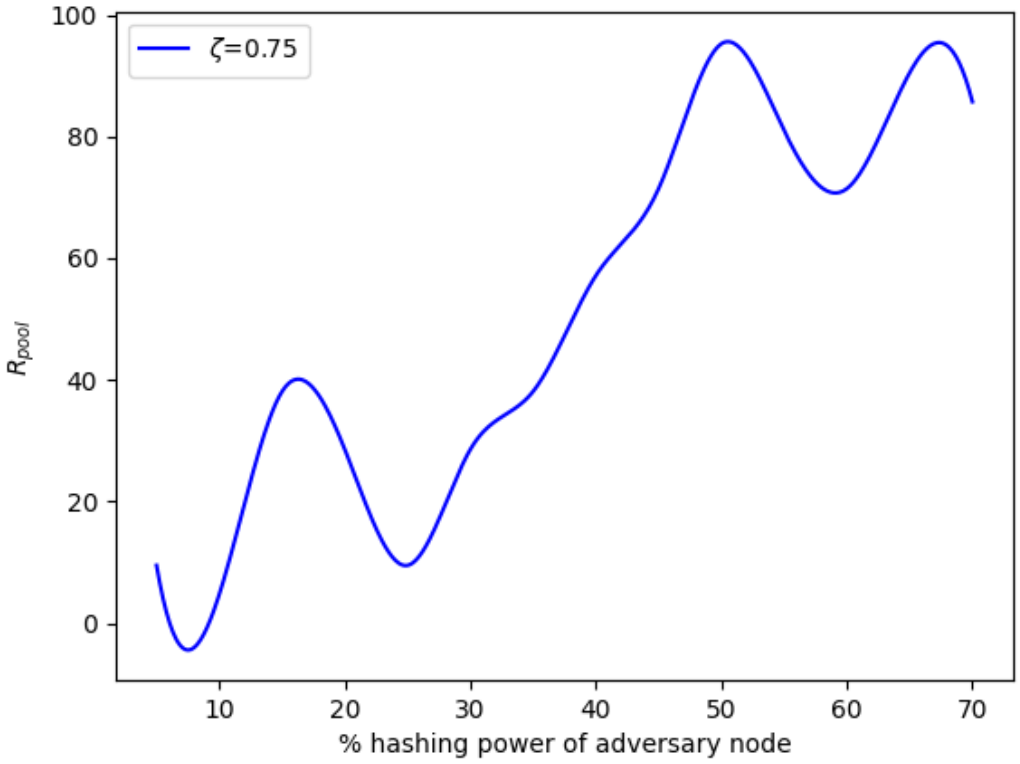


4 Stubborn Mining

Now we will implement stubborn mining which is almost identical to selfish mining, but

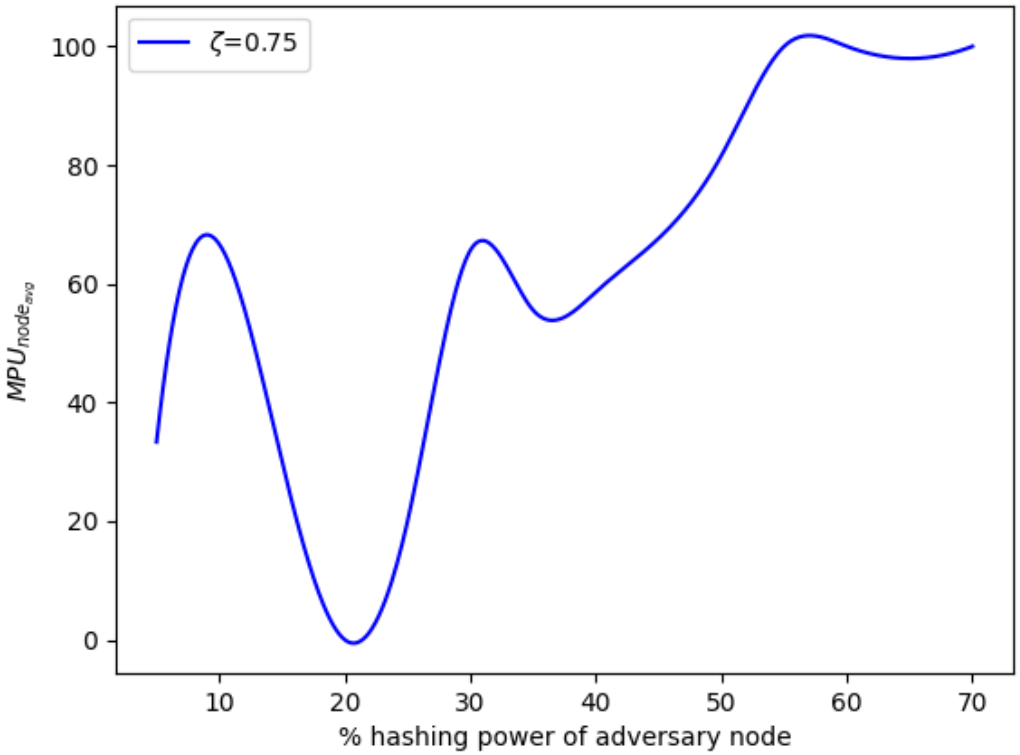
differ only in the fact that when the adversary is ahead by 2 blocks and honest miners mines a block, then instead of releasing the entire chain, only one block is released. We now analyse the stubborn mining attacks through various experiments:

4.1 Ratio of adversary node in longest chain vs hashing power



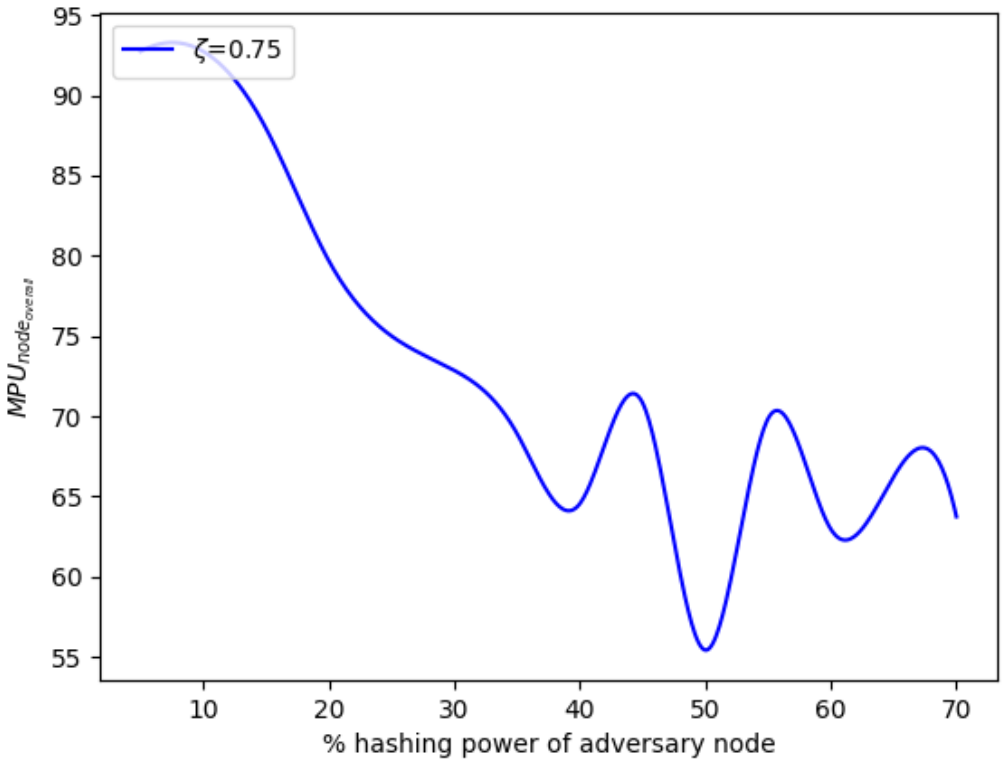
Just as in selfish mining the the graph indicates that as the hashing power increases the ratio of number of blocks in main chain is more than the fraction of hashing power it has. We also see that this can be potentially be a better attack than the selfish mining.

4.2 $\mathbf{MPU}_{node_{avg}}$ vs Hashing Power



As the hashing power increases all the blocks generated by the adversary ends up in the longest chain but as the hashing power decreases it becomes lower. We get to see that the behaviour is similiar to that of selfish mining

4.3 $\text{MPU}_{node_{overall}}$ vs Hashing Power



The $\text{MPU}_{node_{overall}}$ converges to 50% as the hashing power increases just as in the case of selfish mining. This shows that the adversary has indeed succeeded in making lots of fork and orphan blocks.

5 Conclusion

Using our simulation we have correctly shown how selfish mining attack and stubborn mining attack is dangerous to blockchains. Our plots indeed confirm the concept that majority is not enough. We also verify the theoretical value proposed by Eyal and Sirer are in close proximity to practical value. We end our report with the conclusion that stubborn mining and selfish mining are indeed vicious means by which a lot of hashing power of the honest miner can be put to waste. Hence every blockchain should ensure that no more than 20-25% hashing power be available to a single adversary.