CS765 Spring 2023 Semester, Project Part-2

Simulating a double selfish mining attack using P2P Cryptocurrency Network

C. Srikanth Yadav
23M0794

G. Siva Prasad Reddy
23M0747

Sourabh Kumar Kale
23M0783

Question: Find the effect of the different adversary mining powers on these below ratios.

Considering :
Number of Peers = 100
Percentage of slow honest peers =50
Percentage of lowCPU peers: (all honest miners will have same hashing power)
Mean Transaction Time in Network (Ttx) : 1 sec
Mean Block Time in Network (I): 10 secs.

Ratios required :

$\textbf{MPU}_{\textbf{advnode}}$ = Number of blocks mined by the adversary in the final public main chain to the total number of blocks mined by this adversary overall.

$\textbf{MPU}_{\textbf{overall}}$ = Number of blocks in the final public main chain to the total number of blocks generated across all nodes.

$\textbf{Fraction}_{\textbf{adv}}$ = Number of blocks mined by the adversary in the final public main chain to the total number of blocks in the final public main chain

Effect of Number of Peers in Network

Network Propagation Delay : The speed at which new blocks and transactions spread through the network can be affected by how many peers are connected. Adding more peers generally helps things move faster, but it's important to remember that network delays can still slow things down. If the peers are far apart or if the network is congested, it can take longer for information to get around, no matter how many peers there are.

Effect of Mean Transaction Time in Network (Ttx) :

Short Ttx : When the average transaction time (Ttx) is brief, it indicates swift confirmation and inclusion of transactions into blocks. Under such circumstances, selfish miners may encounter increased difficulty in executing their attacks as there is limited time for them to delay blocks or carry out double-spending attempts. A shorter Ttx can narrow the timeframe in which selfish miners might seek to exploit an advantage over honestminers.

Long Ttx : If the average transaction time (Ttx) is prolonged, it implies that the confirmation and inclusion of transactions into blocks require more time. Consequently, selfish miners could exploit this situation more effectively, benefiting from an extended timeframe to delay blocks or carry out double-spending attacks. The increased duration of Ttx raises the probability of selfish miners successfully executing their attacks and deriving profit from them

Effect of Mean Block Generation Time in Network :

Shorter Block Generation Time : When the average block generation time is reduced, it indicates a higher frequency of block production. Under such circumstances, selfish miners face a shorter window to catch up with the network after discovering a block. This heightens the difficulty for selfish miners to sustain a concealed fork of the blockchain and raises the likelihood of their blocks being orphaned when they endeavor to extend the fork.

Longer Block Generation Time : On the other hand, when the mean block generation time is extended, blocks are produced at a reduced frequency. This grants selfish miners an extended duration to construct a longer private fork of the blockchain without attracting notice from the rest of the network. Consequently, there is an elevated probability of selfish miners effectively executing their attack by refraining from sharing blocks with the network until their private fork surpasses the length of the public chain.

Impact on Selfish Mining Strategy : The strategy employed by selfish miners is directly influenced by the mean block generation time. In networks where block generation times are shorter, selfish miners might perceive less profitability in withholding blocks. This is because the quicker rate of block production increases the likelihood of their actions being detected by honest miners. Conversely, in networks characterized by longer block generation times, selfish miners may view attempting to extend a private fork as more lucrative. This is because they have a greater window of time to execute this strategy without attracting detection.

Other parameters :

1. **number of peers**
2. **slow% Peers**
3. **mean Transaction Time**
4. **mean Block Generation Time**
5. **Propagation Delay**

Number of Peers : As the number of peers in the network increases, the propagation time for blocks also increases. If the balance between propagation time and block generation time is not optimal, it can lead to situations where a fork occurs. For instance, if a newly generated block is transmitted before reaching a peer due to propagation delays, and then another block is generated, it can result in a fork in the blockchain.

# of Slow% Peers : When there are numerous slow peers in the network, the propagation delay tends to increase. Consequently, if there's an imbalance between propagation time and block generation time, the potential for forking arises. For instance, if a generated block takes longer to reach a peer due to propagation delays, and another block is generated in the meantime, it can lead to a fork in the blockchain.

Mean Transaction Time : The mean transaction time parameter determines how often transactions are generated within the system. A higher mean transaction time results in fewer transactions being generated within the same simulation timeframe. Subsequently, these transactions are propagated throughout the entire network.

Mean Block Generation Time : A decrease in the mean block generation time raises the probability of blockchain forks. With blocks being generated more frequently, the likelihood of multiple nodes generating blocks simultaneously increases, resulting in temporary forks in the blockchain. These forks can add complexity to consensus mechanisms and necessitate additional resources for conflict resolution.

Propagation Delay : Extended propagation times can delay the distribution of freshly mined blocks throughout the network. This delay may cause nodes to not promptly receive the most recent blocks, potentially resulting in forks or delays in confirming blocks.

Furthermore, prolonged propagation times elevate the likelihood of temporary forks in the blockchain. In such scenarios, different segments of the network may hold divergent perspectives on the current state of the blockchain. These forks may arise when miners at distant ends of the network simultaneously discover blocks but are uninformed of each other's blocks due to propagation delays.

Resolving forks can lead to the squandering of computational resources and slower transaction confirmations.

Selfish Mining Attack

In a selfish mining attack, a miner or group of miners strategically withhold mined blocks from the network while secretly mining their own private chain. Once their private chain becomes longer than the public chain, they release it, invalidating the work done by honest miners and claiming block rewards. By exploiting the delay in propagating blocks across the network, selfish miners can gain an unfair advantage over honest miners. This attack undermines the security and decentralization of the blockchain network by centralizing power in the hands of the attackers.

1. **Mining Blocks Privately :** The attackers begin by mining blocks in secret, without sharing them with the rest of the network. This enables them to build their own private blockchain fork, which isn't immediately visible to other miners and nodes in the network.

2. **Monitoring the Public Chain :** As they mine in private, the attacker keeps an eye on how the public blockchain is progressing. When other miners find a block and share it with the network, they compare their own private chain with the newly extended public chain.

3. **Strategic Release of Blocks :** If the attacker discovers that their private chain surpasses the length of the public chain, they strategically release their privately mined blocks to the network. Their aim is to substitute the public chain with their longer private chain, resulting in a chain reorganization.

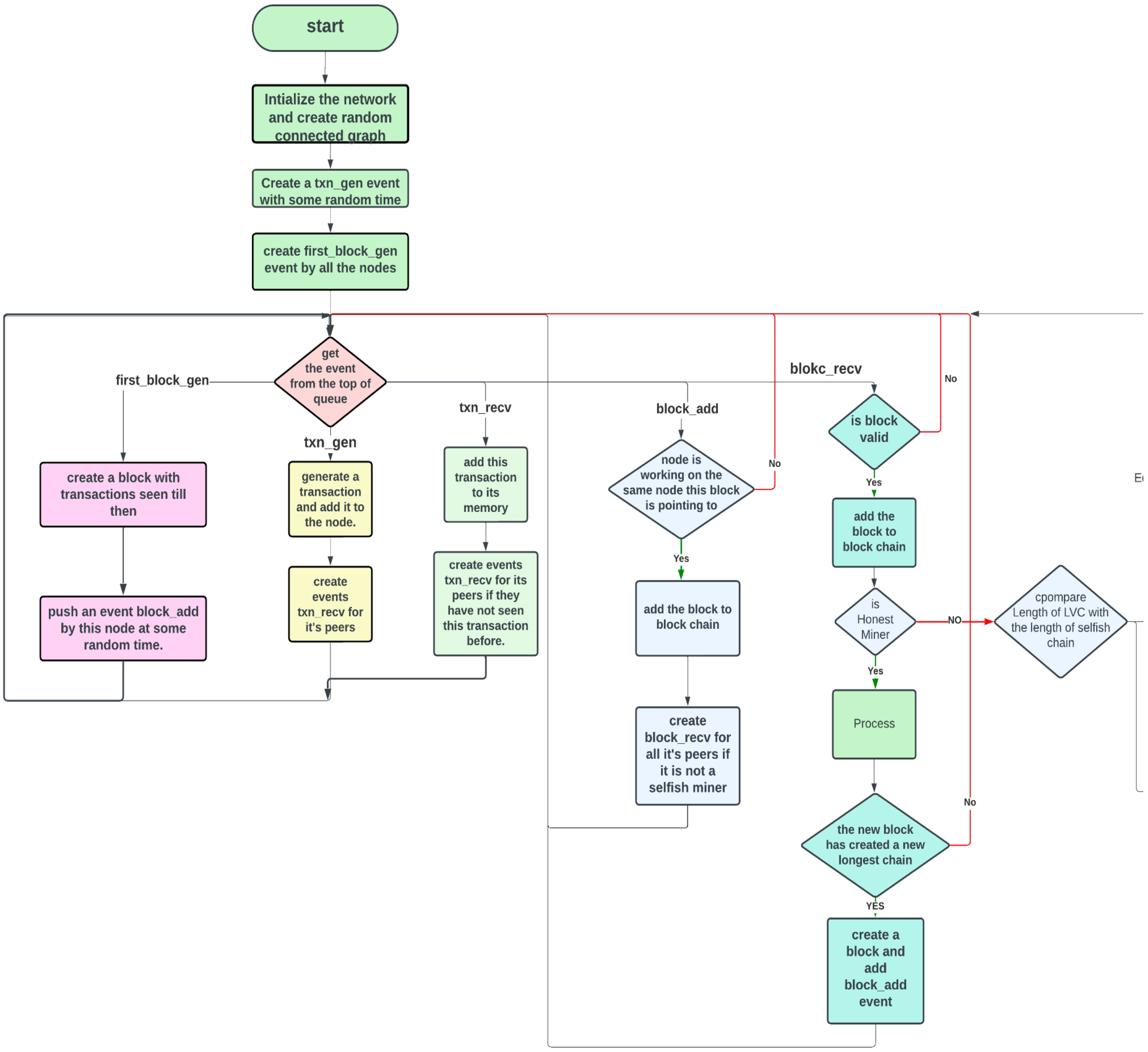
4. **Orphaning Honest Blocks :** After successfully replacing the public chain with their private one, the attacker or attackers cause blocks mined by honest miners, previously included in the public chain, to become orphaned. As a result, the hard work put in by honest miners goes to waste.

NOTE : The selfish mining tactic proves highly impactful when the attacker or attackers possess a substantial share of the network's computing power. This advantage enables them to leverage the inherent delays in transmitting blocks across the network, thus facilitating a quicker extension of their private blockchain compared to the public one.

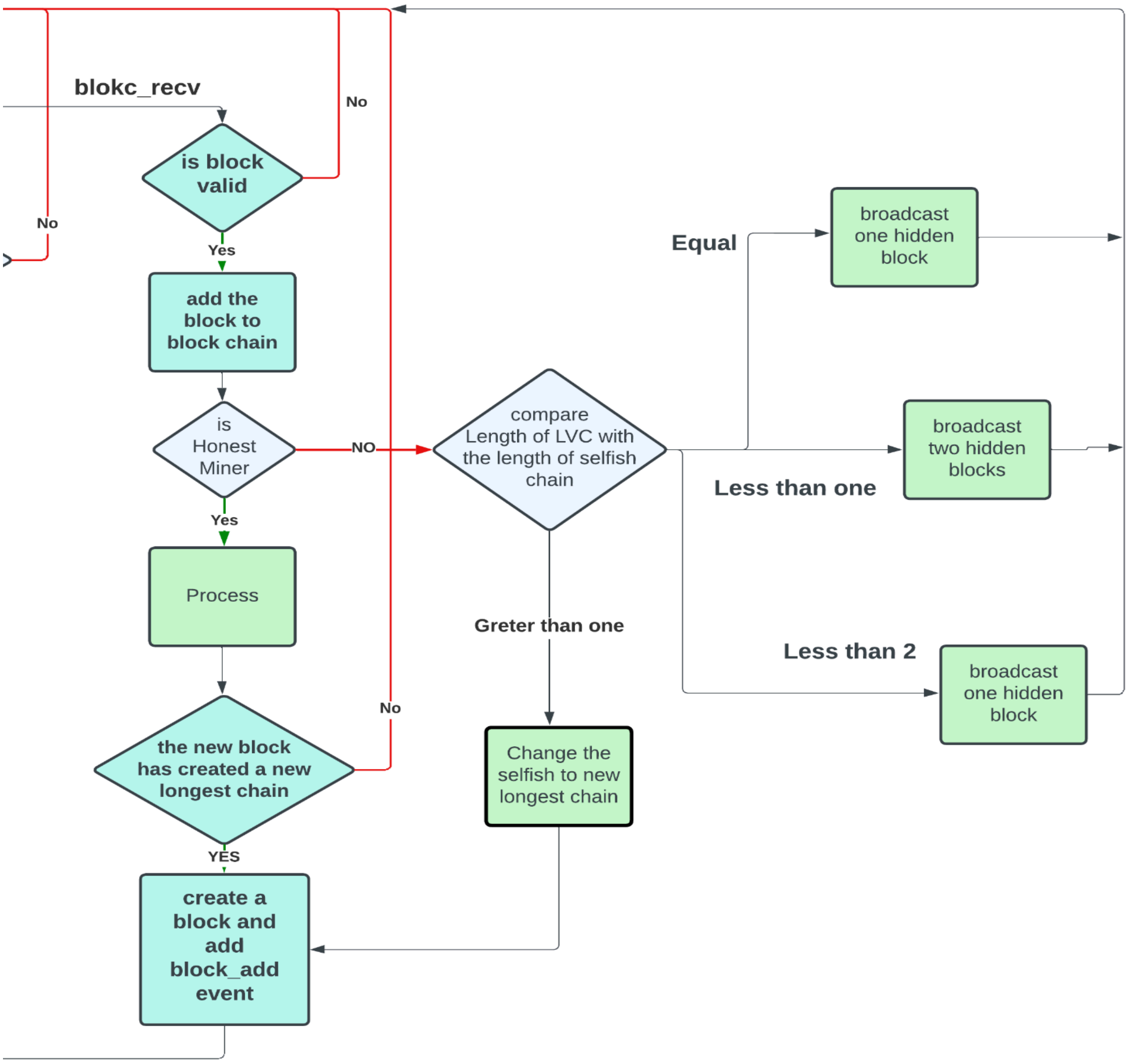
Double Selfish Mining Attack :

The double selfish mining attack involves two separate groups of miners independently attempting to execute selfish mining strategies without knowledge of each other's actions. Each group aims to create longer private chains in parallel, exploiting the latency in block propagation across the network. By strategically withholding and selectively releasing blocks, these groups aim to increase their chances of successfully replacing the public blockchain with their longer private chains. This coordinated effort allows them to undermine the integrity of the blockchain network and potentially gain unfair advantages in terms of block rewards and transaction confirmations

Design Document


start
Intialize the network and create random connected graph
Create a txn_gen event with some random time
create first_block_gen event by all the nodes
get the event from the top of queue
first_block_gen
txn_gen
txn_recv
block_add
blokc_recv
create a block with transactions seen till then
push an event block_add by this node at some random time.
generate a transaction and add it to the node.
create events txn_recv for it's peers
add this transaction to its memory
create events txn_recv for its peers if they have not seen this transaction before.
node is working on the same node this block is pointing to
No
Yes
add the block to block chain
create block_recv for all it's peers if it is not a selfish miner
is block valid
No
Yes
add the block to block chain
is Honest Miner
NO
cpompare Length of LVC with the length of selfish chain
Yes
Process
the new block has created a new longest chain
No
YES
create a block and add block_add event
E

Design Document of the Selfish Miner


blokc_recv
is block valid
No
No
Yes
add the block to block chain
is Honest Miner
NO
Yes
Process
the new block has created a new longest chain
No
YES
create a block and add block_add event
compare Length of LVC with the length of selfish chain
Equal
broadcast one hidden block
Less than one
broadcast two hidden blocks
Greter than one
Less than 2
broadcast one hidden block
Change the selfish to new longest chain

Analysis With respect to Hashing Power of Adversary 1 and Adversary 2

Results of Simulation

Here, in the Simulation # 1 : Hashing Power of Adversary 1 is {0,20,30} .

Hashing power of adversary 1	Hashing power of adversary 2	MPU of adversary 1	MPU of adversary 2	MPU overall	Fraction of adversay1	Fraction of adversary 2
0	0	0	0	1	0	0
0.2	0.1	0.67	0	0.65	0.25	0
0.3	0.2	1	0	0.48	0.88	0
0.3	0.3	0.125	0.8	0.38	0.06	0.48
0.3	0.4	0	0.92	0.7	0	0.77

With the increase in the hashing power of selfish miners, the MPU of the selfish miner is increasing and the fraction of the miner blocks in the main block chain is increasing.

If both the selfish miners have almost equal hashing power then MPU overall is decreasing because both the selfish miners will create more forks.

If any selfish miner has some 40% hashing power, it is dominating the other selfish miner making the blocks of the other miner orphan, so it will decrease the MPU of other selfish miner and increase the MPU of this miner.

When the hashing powers of selfish miners are low, it allows the honest miners to mine more blocks so there will be less forks resulting in high MPU overall and low selfish miner MPU and fractions

Here, in the Simulation # 2 : Hashing Power of Adversary 1 is 0.4 .

Hashing power of adversary 1	Hashing power of adversary 2	MPU of adversary 1	MPU of adversary 2	MPU overall	Fraction of adversary 1	Fraction of adversary 2
0.4	0	0.88	0	0.76	0.48	0
0.4	0.1	0.8	0	0.69	0.54	0
0.4	0.2	0.89	0.5	0.65	0.60	0.15
0.4	0.3	0.9	0	0.52	0.69	0
0.4	0.4	0	1	0.54	0	0.875

We have fixed the hashing power of C1 as 40% and tested with C2= 0,10,20,30,40.

With C1=40, C2=0 and honest =60%, MPU of C1 is high and MPU overall is high since there will be less forks because only C1 is making the forks.

With the increase in C2, we can observe that MPU overall is decreasing because both the miners are creating blocks and more blocks will be orphaned.

When the hashing power of both the selfish miners are 40% , then they will completely dominate the honest miners and both will create long private chains. This resulted in the winning of only one chain making the private chain orphan. So the MPU of one miner is very high, while the other miner is very low.

Here, in the Simulation # 2 : Hashing Power of Adversary 1 is 0.4

Hashing power of adversary 1	Hashing power of adversary 2	MPU of adversary 1	MPU of adversary 2	MPU overall	Fraction of adversary 1	Fraction of adversary 2
0.5	0	0.95	0	0.78	0.75	0
0.5	0.1	1	0.4	0.83	0.88	0.06
0.5	0.2	1	0.14	0.69	0.94	0.03
0.5	0.3	0.89	0.44	0.69	0.77	0.13
0.5	0.4	1	0	0.57	0.93	0

We have took C1=50% and have tested with C2=0,10,20,30,40.

When C1=50% , most of the blocks will be created by the C1 and most of the blocks in the main chain will be C1's.

With the increase in the hashing power of C2, no.of MPU overall is decreasing because C2 will create forks which will not be included in the main chain.

When C2=40%, both the C1, C2 will have long private chains and the C1has more hash power so the main chain will be C1's and C2 blocks will be orphaned. So there will be more orphaned blocks when the hashing powers are the same resulting in the low MPU of C2 and high MPU of C1.

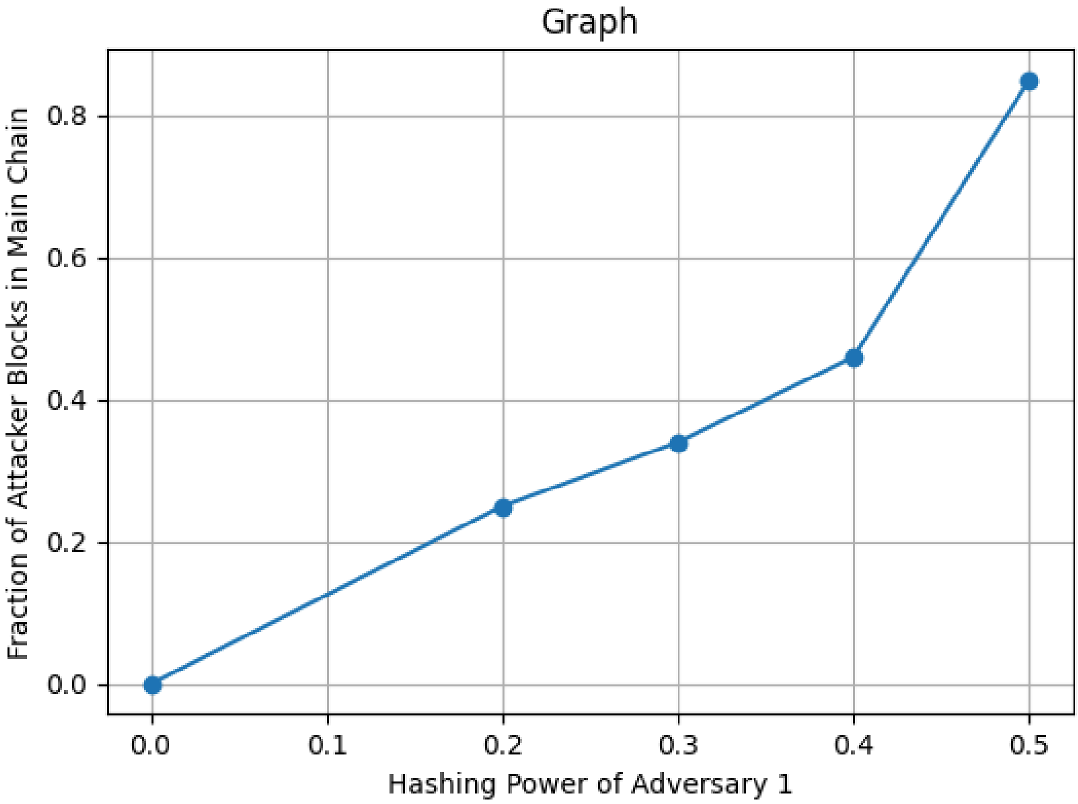


Figure 1: Hashing Power of Adversary is 0

When the hashing power of one selfish miner is 0%, It is more like a single selfish miner attack.
The fraction of selfish miner is increasing with the increase in hashing power of selfish miner, and it has increased exponentially when it reached 50% resembling 51% attack.

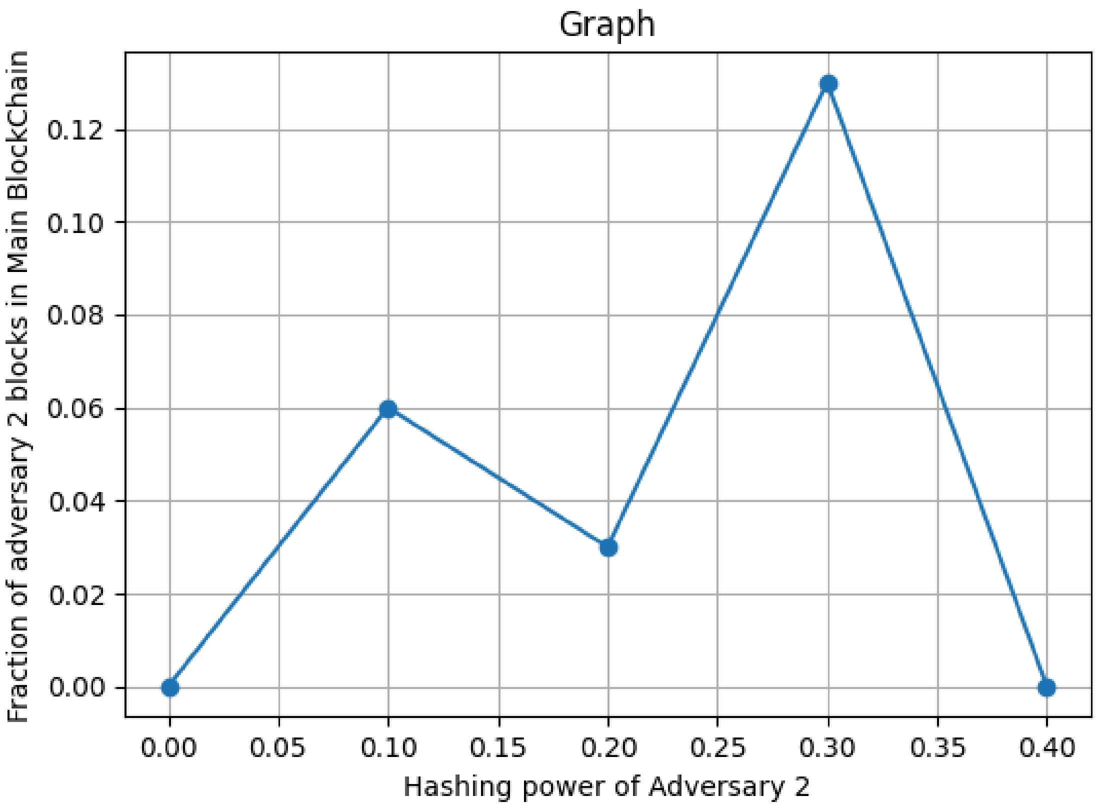


Figure 2: Hashing power of Adversary 1 is 0.5

The hashing power of C1=50%.

When C2=0, its fraction of blocks will be 0

With the increase in the hashing power of C2, the fraction of C2 is increasing.

When the hashing power of C2 is also high i.e 40%, both the miners will have long private chains , as the C1 is greater than C2 the main chain will be C1, making the private chain of C2 orphan. So the fraction of C2 is 0 when C2=40%. All the blocks will be of C1.

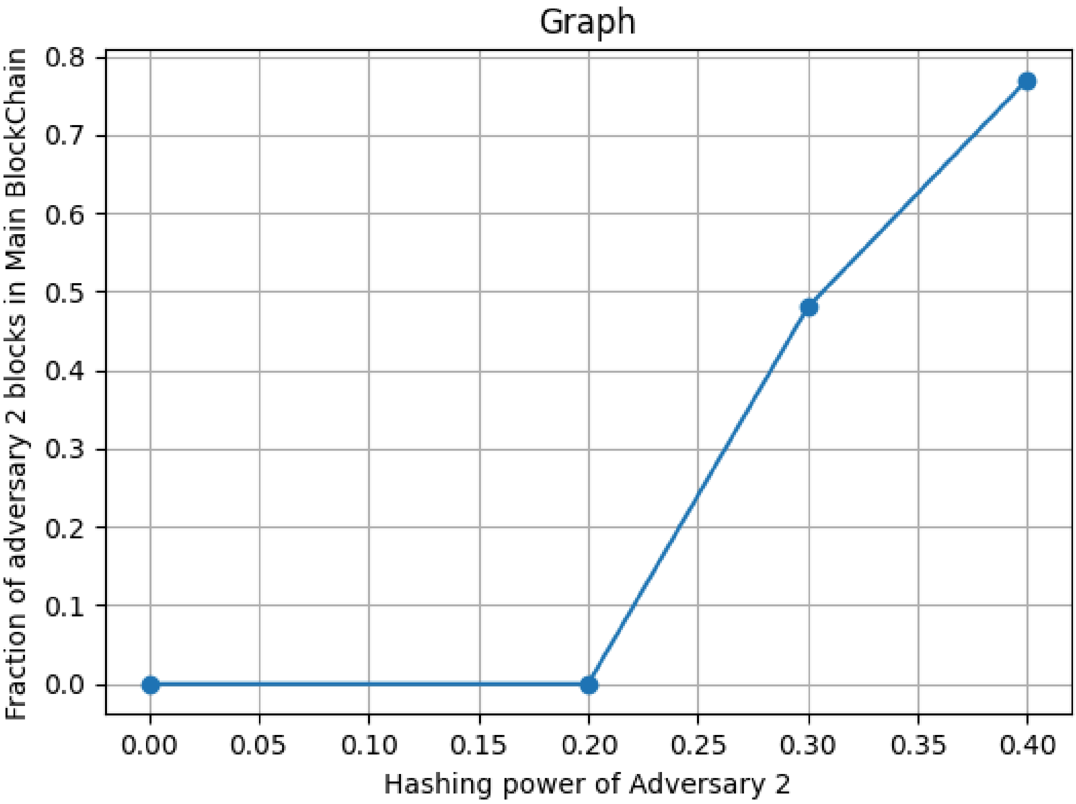


Figure 3: Hashing power of Adversary 1 is 0.3

When the hashing power of C2 is less than the C1, and C1 is dominating the C2 making the C2 nodes orphan.

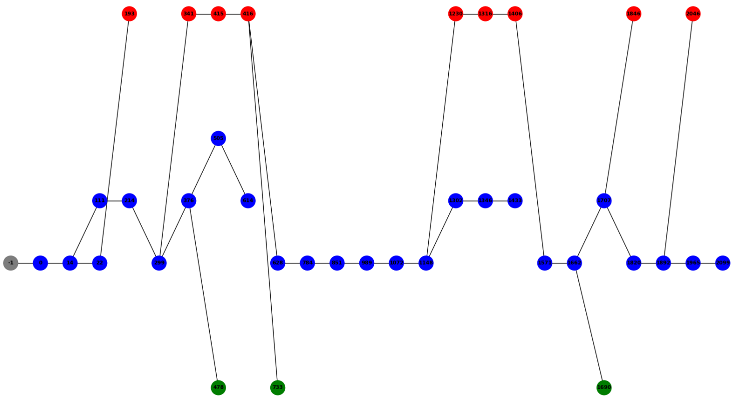
With the increase in hashing power of C2 , the fraction of blocks of C2 is increasing.

When the hashing power of C2 is very high, it dominates the C1 and honest miners resulting in the increase of C2 fraction exponentially.

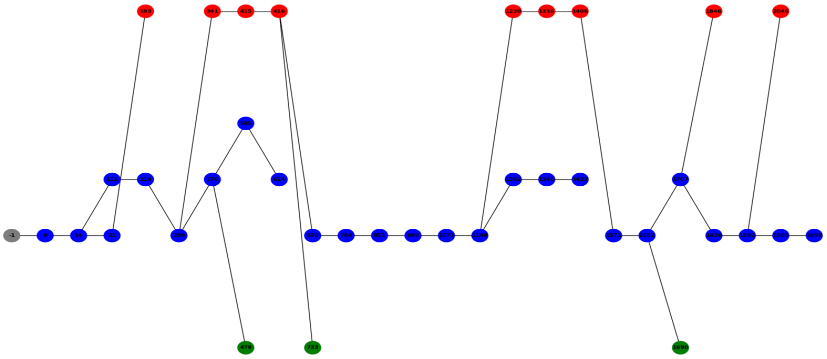
BlockChain Trees:

1) Adversary 1: 20% , Adversary 2:10%

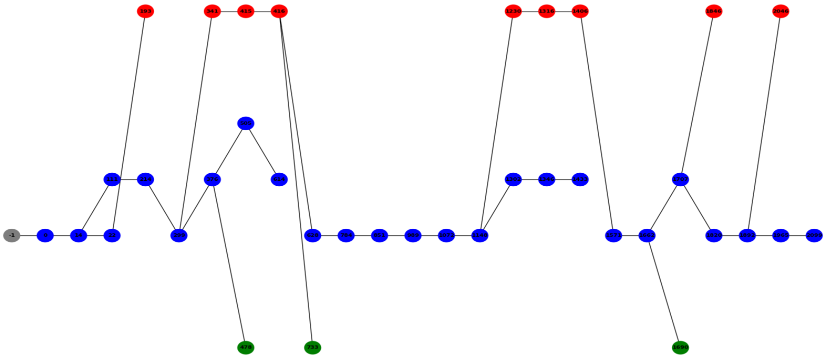
Honest:



Adversay1:



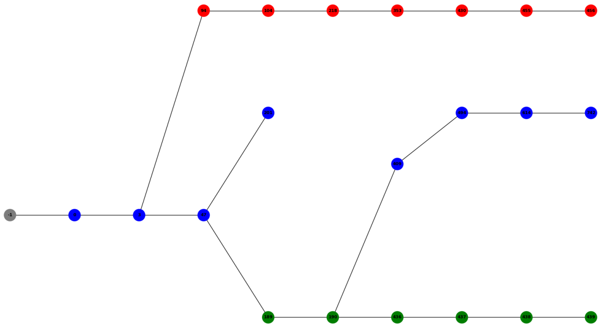
Adversary 2:



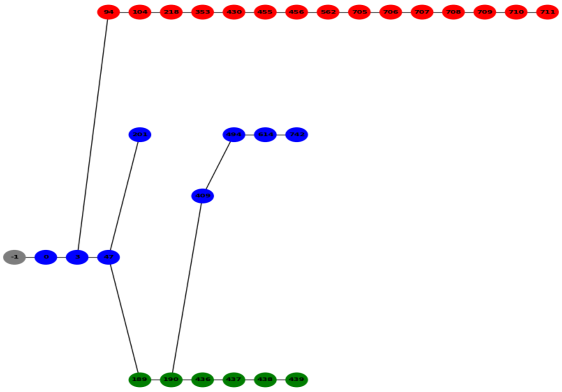
When the hashing powers of both selfish miners are low, they can’t create long private chains.

2) Adversary 1=40%, Adversary 2=40%

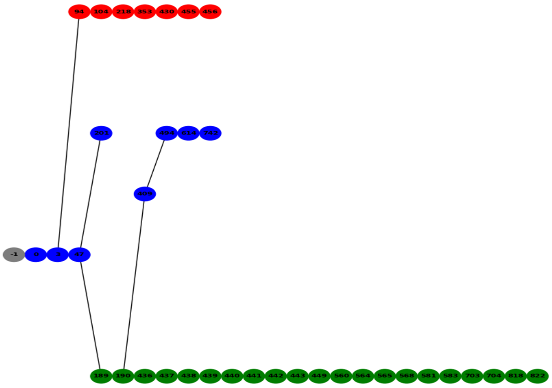
Honest:



Adversary1:



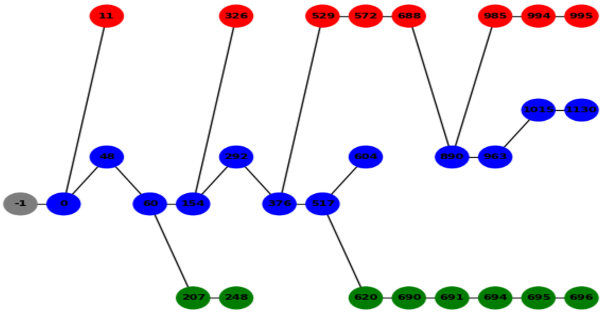
Adversary2:



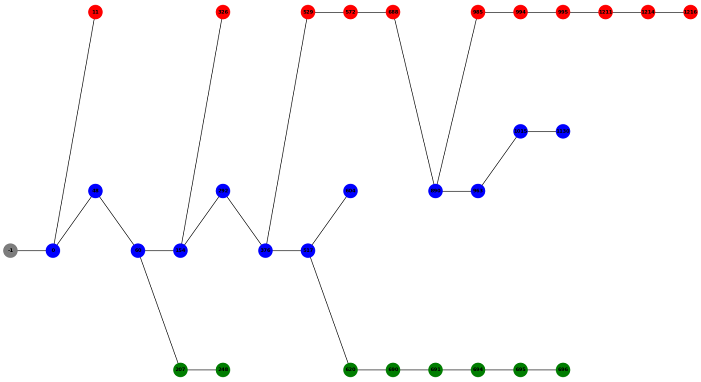
When the selfish miners have both high hashing powers, both will have longer private chains.

3) Adversary 1:30% Adversary 2:40%

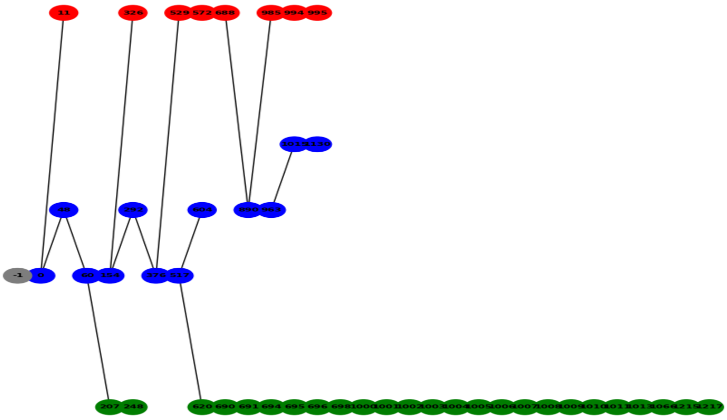
Honest:



Adversary1:



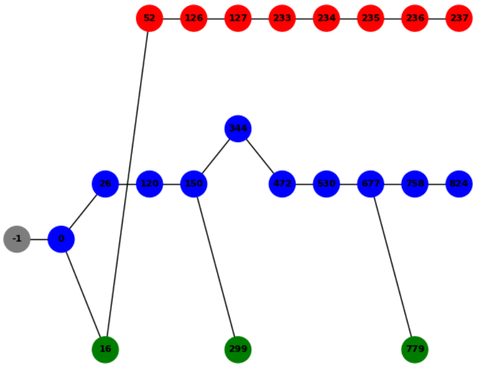
Adversary 2:



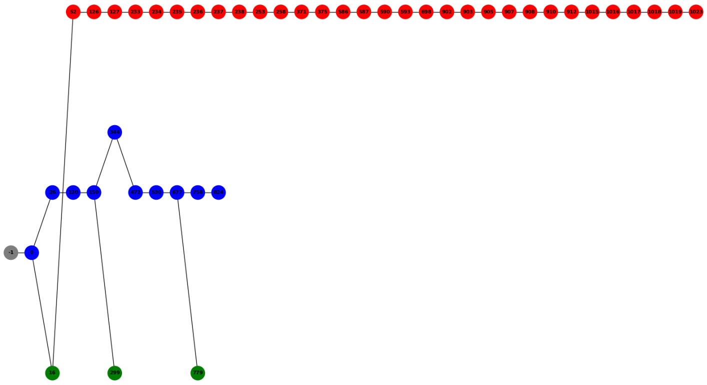
Adverversary1: 30% so it has a shorter private chain at last, and Adversary 2: 40% dominated the other.

4) Adversary 1: 50% Adversary 1: 20%

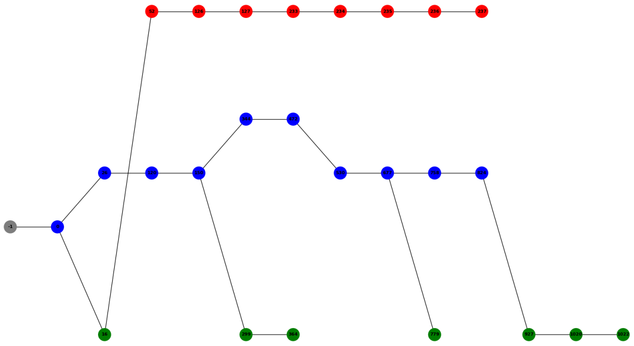
Honest:



Adversary1:



Adversary2:



Adversary 1: 50% will completely dominate all other miners and adversary 2: 20% will have some private blocks at last.

FINAL OBSERVATIONS:

1) With the **increase in the hashing power** of selfish miners the **MPU overall is decreasing** because both the miners will create private chains and most of them will be orphaned.
2) With the **increase in hashing power** of miners the **fraction of their blocks is increasing.**
3) When both miners have high hashing power(almost equal) then the **MPU** of one Adversary is almost one and while the other Adversary's **MPU** is going to be approximately zero.Because most of the the times, both gonna start mining from Genesis block and at each point they get forks ,but at somewhere, one gonna orphan big chain of other.
4) When both the miners have high hashing power (almost equal) one miner is dominating the other making the **fraction of blocks** of the other miner to be very less. Because both the miners will create long private chains and only one of them will win, making the other chain orphan. So all the blocks in the private chain of the other miner will be orphaned.
5) When both the adversaries have low hashing powers , they can't create long private chains.