Homework 2

Due on Monday, Nov. 6 , 2023, 11:59 PM (via Canvas)

Problem 1

1 Combining Nakamoto's Private Attack with Premining

In this scenario, we explore a unique attack strategy that combines Nakamoto's private attack with a premining phase. The primary objective of this attack is to reverse a specific transaction, TX, which is part of the i-th block within the public chain.

The attack unfolds as follows:

1. **Premining Phase:** Commencing from the genesis block, the attacker initiates private mining operations to construct a separate private chain. When the first honest block, denoted as h_1, is mined atop the genesis block, the attacker has two options:

   i. If the private chain surpasses the public chain in length at that moment, the adversary continues to mine on the private chain.

   ii. If the private chain is equal or shorter than the public chain, the attacker abandons the current private chain and initiates a new private chain from block h_1. This process is iteratively repeated for all honest blocks h_2, h_3, and so on, up to h_{i-1}.

2. **Private Attack Phase:** Once block h_{i-1} is mined, the attacker's decision is based on which chain is longer. If the private chain is longer, the attacker launches Nakamoto's private attack from the current private chain. Otherwise, the attack begins from block h_{i-1}.

Now, let's address some key questions:

1. If $\beta = 0.3$, what is the probability of the attacker switching to h_1 upon its mining? What is the expected depth at which the attacker is mining when h_1 is reached?

   Solution: For every block creation event, with probability β it is an adversarial block and with probability $(1-\beta)$ it is an honest block. Thus, the number of adversarial blocks before the first honest block arrives, denoted by D_1^a, is a geometrically distributed random variable with parameter $1-\beta$, i.e., $D_1^a \sim \text{Geometric}(1-\beta)$. The adversary switches to h_1 if $D_1^a \leq 1$, which happens with probability

   $$\Pr[D_1^a = 0] + \Pr[D_1^a = 1] = (1-\beta) + \beta(1-\beta) = 1 - \beta^2. \tag{1}$$

   On the other hand, the expected depth at which the attacker is mining when h_1 is reached is

   $$\mathbb{E}[D_1^a] = \frac{\beta}{1-\beta} = \frac{3}{7}. \tag{2}$$

2. Consider the sequence of blocks being mined in the following order: a_1, a_2, h_1, a_3, h_2, h_3, a_4, a_5, h_4. Please illustrate the evolving block tree.

   Solution: The block tree is as depicted in the following figure.

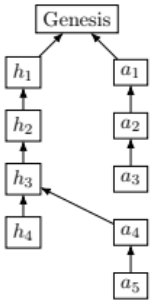

Figure 1: The block tree of problem 1, part 2.

3. Discuss why this combined attack is stronger than the pure Nakamoto's private attack.

   Solution: If the attacker decides to skip the pre-mining phase, they fall back to Nakamoto's private attack. In any situation where Nakamoto's attack works, the pre-mining attack works too. So, the pre-mining attack is at least as powerful as Nakamoto's attack. While pre-mining doesn't always give an advantage, there are times when it does. This means there's a chance the pre-mining attack succeeds even if Nakamoto's attack would have failed. In simple terms, pre-mining is stronger than Nakamoto's attack because the situations where it succeeds cover everything Nakamoto's attack can handle, and more.

4. In class, we analyzed the confirmation error probability under Nakamoto's private attack and demonstrated its exponential decrease with increasing k, bounded by c. Show that the confirmation error probability under the attack with premining also exhibits an exponential decrease. (Hint: Consider all potential scenarios in which the adversary switches during the premining phase.)

   Solution: Let's define $E_{j,m}$ as the event where the adversary switched to the honest chain when an honest block $j < i$ was mined and then successfully grew a chain with a length of at least $m \geq (i - j) + k$ faster than the honest miners. Additionally, let E represent the error event for the k-deep confirmation rule under

the pre-mining attack. We have

$$\Pr[E] = \Pr\left[\bigcup_{j=0}^{i-1} \bigcup_{m=(i-j)+k}^{\infty} E_{j,m}\right] \tag{3}$$

$$\leq \sum_{j=0}^{i-1} \sum_{m=(i-j)+k}^{\infty} \Pr[E_{j,m}] \tag{4}$$

$$\overset{(a)}{\leq} \sum_{j=0}^{i-1} \sum_{m=(i-j)+k}^{\infty} \exp(-cm) \tag{5}$$

$$= \sum_{j=0}^{i-1} \frac{1}{1-\exp(-c)} \exp(-((i-j)+k)c) \tag{6}$$

$$= \frac{\exp(-kc)}{1-\exp(-c)} \sum_{j=0}^{i-1} \exp(-(i-j)c) \tag{7}$$

$$= \frac{\exp(-kc)}{1-\exp(-c)} \frac{(1-\exp(-ci))}{\exp(c)-1} \tag{8}$$

$$= \frac{\exp(-c)(1-\exp(-ci))}{(1-\exp(-c))^2} \exp(-ck), \tag{9}$$

where (a) follows from exponential decay of the probability of the adversary winning the block production Poisson race, as explained in class.

Problem 2

2 Analyzing Blockchain Forks

Consider a blockchain network where block creation is regulated to achieve an average block generation rate of λ. Now, let's introduce a scenario in which it takes Δ seconds to disseminate a newly discovered block to other miners. During this interval, remaining miners persist in their efforts to mine a new block atop the preceding block, which is now considered outdated.

Let's delve into several critical aspects of this blockchain setup:

1. Fraction of Forked Blocks: Investigate how the fraction of blockchain heights with more than one block (indicating a fork) varies as a function of Δ. It's essential to assume that honest miners adhere to the longest-chain rule and resolve ties by building on the block with the earliest timestamp. We should also consider an infinite number of honest miners, each contributing infinitesimal mining power.

   Solution: Let's consider the scenario at block height i and assume that the first block $b_{i,1}$ at that height has just been mined. It takes time Δ until the remaining miners become aware of the existence of block $b_{i,1}$. During this time, the miners continue to mine on height i at a total rate of λ. In this context, we assume that the miner who mined $b_{i,1}$ contributes a negligible fraction to the total mining rate λ. This assumption ensures that the fact that this particular miner has already moved on to height $i+1$ does not significantly reduce the overall mining rate on height i. Since block inter-arrival times follow an exponential distribution,

the probability that no other block is mined on height i during Δ is given by $\exp(-\lambda\Delta)$. After time Δ, the remaining miners become aware of $b_{i,1}$ and move on to block height $(i+1)$. If, during Δ, additional blocks are mined on height i, these are later abandoned, and the blockchain continues on top of $b_{i,1}$ due to the tie-breaking rule. Applying the law of large numbers, the fraction of block heights that have only one block is $\exp(-\lambda\Delta)$, and the fraction of block heights that have more than one block is $1-\exp(-\lambda\Delta)$.

2. Consensus System Implications: Explore the implications of blockchain forking on the broader consensus system.

   Solution: Forking disrupts consensus by requiring additional blocks for miners to agree on which branch to follow, resulting in increased latency. Additionally, blocks that are later abandoned contribute to reduced throughput, representing wasted computational effort. Moreover, forking introduces vulnerabilities, enabling an adversary to launch successful attacks against the blockchain with less than half of the total hashing power. This is because forking divides the hash power of honest miners among multiple chains. Some honest miners may unwittingly contribute to the adversary's goal of disrupting consensus. As a result, the adversary only needs to actively work against a smaller portion of the overall honest hashing power. This makes the blockchain more susceptible to adversarial interference.

3. Scaling Proposals and Feasibility: Examine how conventional scaling proposals, such as enlarging block size or accelerating block creation rates, are related to the analysis in part (1). Utilize the insights gained in part (2) to make a compelling case for why neither of these proposals represents a viable solution for scaling.

   Solution: Suppose the block size is increased by a factor of k. This modification would result in a delay of $k\Delta$. Referring to the analysis in part 1, the fraction of block heights that do not undergo forking is then given by $\exp(-\lambda k\Delta)$. On the contrary, consider a scenario where the block creation rate is increased by a factor of k. This adjustment would result in a block creation rate of $k\lambda$. Employing the analysis in part 1, the fraction of block heights that avoid forking is then $\exp(-\lambda k\Delta)$. Given that the product $\lambda\Delta$ is the critical parameter determining the extent of forking and, consequently, the security of the chain (as argued in part 2), both scaling proposals lead to the same outcome of a deterioration in security.

4. Centralization and Security: Now, let's introduce a scenario in which there is a limited number of honest miners, denoted as n, with each miner possessing a fraction of the hashing power equal to $\frac{1}{n}$. Discuss the effects of increased centralization as n decreases on your answer to part (3). Can you clarify any trade off between centralization and security in this context?

   Solution: When there is a finite number of honest miners n, and each miner possesses a fraction of $1/n$ of the hashing power, the honest mining rate 'left behind' on block height i during a delay Δ is only $\frac{n-1}{n}\lambda$ rather than λ. Thus, via a similar argument as in part 1, the fraction of block heights that have only one block is $\exp\left(-\frac{n-1}{n}\lambda\Delta\right) > \exp(-\lambda\Delta)$. Hence, the deterioration of security by a scaling proposal such as in part 3, is partially offset by such centralization. This observation is consistent with the Blockchain Trilemma, which says that a blockchain system cannot have all three of the decentralization, security, and scalability. In the case discussed here, decentralization is sacrificed to retain security and gain scalability.

Problem 3

3 Lower Bound for Chain Quality

In this question, we aim to establish the lower bound for chain quality. Let's consider the scenario where there is zero delay, i.e., $\Delta = 0$, implying that honest blocks are immediately known to everyone.

1. What level of chain quality can be achieved if the adversary mines in a manner identical to an honest node?

   Solution: If the adversary mines like an honest node, it mines β fraction of the blocks on the longest chain. Consequently, $1-\beta$ fraction of the blocks on the longest chain are mined by honest nodes. Therefore, the chain quality (CQ) is $1-\beta$.

2. Demonstrate that the lower bound for chain quality is $\frac{1-2\beta}{1-\beta}$, and prove that this bound is attainable. In other words, show that there exists an adversarial strategy through which an average of $\frac{1-2\beta}{1-\beta}$ fraction of the blocks in the final chain will have been mined by honest nodes.

   Solution: This bound and its achievability have been covered in detail in class. For further information, please refer to the lecture notes or this paper: "*Majority is Not Enough: Bitcoin Mining is Vulnerable*".

3. Discuss the implications for mining rewards and transaction throughput when the chain quality is reduced to $\frac{1-2\beta}{1-\beta}$.

   Solution:

   (a) The fair share of mining rewards for an adversary controlling a fraction β of the computational power would be β fraction of the total mining reward. However, the reduced quality of the blockchain results in a larger share of mining rewards for the adversary, creating an unfair incentive structure that discourages honest behavior.

   (b) Chain quality has a direct impact on transaction throughput, as the adversary can mine empty blocks that do not contribute to transaction processing. The diminished chain quality leads to a reduction in transaction throughput.

4. Prove that the growth rate of the longest chain is $\frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta}$ when the adversary (with β fraction of mining power) stops mining and delays every honest block with the maximum delay Δ.

   Solution: Suppose an honest block B is mined by a miner P at time t, and B is the first block at level ℓ. In the worst-case scenario, B is not received by the remaining honest miners until time $t+\Delta$. Given the network's vast number of honest miners, each with infinitesimal mining power, it's highly improbable that miner P will mine another block before time $t+\Delta$. As other miners have not seen B, their block will be mined at the same level as B (with a parent block at level $\ell-1$). Assuming the honest mining process follows a Poisson process with a rate of $(1-\beta)\lambda$, on average, $(1-\beta)\lambda\Delta$ honest blocks are mined in the time interval $[t, t+\Delta]$. Since these blocks are all at level ℓ, they do not increase the length of the longest chain. The first block mined after $t+\Delta$ will increase the length of the longest chain by one. Therefore, on average, only 1 out of $1+(1-\beta)\lambda\Delta$ blocks will contribute to the growth of the longest chain. Hence, the growth rate of the longest chain is given by $\frac{(1-\beta)\lambda}{1+(1-\beta)\lambda\Delta}$. On the other hand, since the adversary stops mining, the growth rate of the longest chain is indeed the growth rate of the honest chain.

Problem 4

4 Block Difficulty Analysis

1. For this Bitcoin block, located at block, what is the difficulty of this block? Based on this information, calculate the number of leading zeros in the mining target in hexadecimal format. To verify your answer, compare it with the block hashes from that day. You can refer to this link for additional details about Bitcoin difficulty.

   Solution: The difficulty of this block is $61,030,681,983,175.59$. According to the provided link, the expected number of hashes needed to find a valid block for a difficulty D, is given by $\frac{D\times 2^{48}}{0xffff}$, which for this

case is approximately 2^{77}. On the other hand, since the output of the hash is a random number, to have k leading zero bits at the Most Significant Bits (MSB) of the result, you need to test 2^k expected hash values to find a valid hash. Combining these considerations, it means that the hash of this block and its neighbors should have around 77 zeros in the MSB, which is indeed true. In particular, for this block, we observe 76 leading zeros in the beginning.

2. Let's consider the following blockchain protocol, where each block B_i is associated with a threshold value Th_i. The miner of each block must solve the puzzle $\mathsf{Hash}(B_i) \leq \mathsf{Th}_i$ by experimenting with different nonces within the block. We define the weight of a block as $W_i = \frac{1}{\mathsf{Th}_i}$. For a chain, the weight of the entire chain is computed as the sum of the weights of all the blocks within that chain. The fork choice rule in this network is based on the heaviest chain, i.e., the chain with the highest weight. In the event of a tie, the choice is the oldest timestamp. The threshold adjustment's rule in this protocol is as follows:

   (a) Within each epoch, which encompasses 2016 blocks, starting from block number 1 and continuing through block 2016 for the first epoch, and then from block 2017 through block 4032 for the second epoch, and so on, the threshold remains constant. In other words, the threshold for the blocks is fixed within each individual epoch. For the epoch e_i, we denote its block threshold by Th_i^e.

   (b) For each epoch e_i, we determine its threshold based on the average inter-block time from the previous epoch. Additionally, we ensure that the threshold is adjusted with a maximum factor of 4 per epoch. Specifically, assume that we aim to achieve an average inter-block time to a target value T, such as the 10-minute target in Bitcoin. For the epoch e_i, we calculate the threshold as

$$\mathsf{Th}_i^e = \mathsf{Th}_{i-1}^e \times \max(\frac{\bar{t}_{i-1}}{T}, \frac{1}{4}), \tag{10}$$

   where $\bar{t}_{i-1}$ represents the average inter-block time from the previous epoch e_{i-1}.

      i. Assume that instead of (10), we use the formula $\mathsf{Th}_i^e = \mathsf{Th}_{i-1}^e$, for all $i > 1$, which implies that threshold is always a fixed number, from the day one! What is the cons of using this mechanism?
      Solution: If the difficulty remains unchanged since the genesis block, with the total hash rate on the rise, the block generation rate increases accordingly. However, this poses a security challenge for the network, as the probability of natural forks occurring also increases.

      ii. Consider a scenario where, for each block, the miner has the flexibility to select an arbitrary threshold, yet the main chain remains the heaviest one. What are the drawbacks of this mechanism? Assuming the presence of an adversary with $\beta = \frac{1}{3}$, describe an attack by this adversary that enables the double spending of a transaction. This attack should remain effective regardless of the choice of parameter k for the k deep rule.
      Solution: Suppose honest miners stick to the initial mining weight set in the genesis block, aiming for an average time of 10 minutes between blocks. Let's use 10 minutes as our time unit and consider the initial weight as the weight unit. In this scenario, it takes an average of k units of time to mine an honest chain with k blocks. Now, imagine the adversarial mining power is half of the honest mining power (or 1/3 of the total mining power). For the adversary to mine a heavier chain, they only need to mine one block with the weight equivalent to k honest blocks within k units of time. The adversarial mining process for a block with the weight of k, follows a Poisson point process with a rate of $1/2k$, and the number of adversarial blocks with the weight of k, mined in k units of time follows the Poisson distribution $Poiss(1/2)$. As a result, the success probability of this attack is given by $P(\text{attack succeeds}) = P(\text{Poiss}(1/2) \geq 1) = 1 - e^{-1/2} \approx 39.3\%$. Importantly, this success probability is a constant independent of k, meaning that any k-deep confirmation rule will fail.

      iii. Assume that instead of (10), we use the formula $\mathsf{Th}_i^e = \mathsf{Th}_{i-1}^e \times \frac{\bar{t}_{i-1}}{T}$, for all $i > 1$, which implies that there is no bound for threshold adjustment. What are the cons of using this mechanism? Assuming the presence of an adversary with $\beta = \frac{1}{3}$, describe an attack by this adversary that

enables the double spending of a transaction. This attack should remain effective regardless of the choice of parameter k for the k deep rule.

Solution: The attacker has the flexibility to set any timestamp in its private blocks, allowing for an arbitrary weight for the second epoch in its private chain, as long as the adversary completes the first epoch. Assume that in the first epoch, adversary produces 2016 blocks with the wight of 1. Let B, with weight X, be the first block of the second epoch in the private chain. Then, the chain including B, has a chain weight of $2016 + X$. To mine an honest chain with a chain weight of $2016 + X$, it takes an average of $2016 + X$ units of time. On the other hand, for the adversary, it takes an average of 4032 units of time to complete the first epoch in its private chain (Since it has the half power of the honest nodes). Therefore, for the attack to succeed, the adversary needs to mine block B within $2016 + X - 4032 = X - 2016$ units of time, which happens with the probability:

$$P(\text{attack succeeds}) = P(\text{Poiss}(\frac{X - 2016}{2X}) \geq 1) = 1 - e^{-\frac{X-2016}{2X}} \approx 1 - e^{-\frac{1}{2}} \approx 39.3\%,$$

if $X \gg 2016$. Note that the success probability is independent of the length of the public longest chain; hence, any k-deep confirmation rule will fail again.