CS-3812 HW 02

In=25 March 11:45pm; Out=11 March

*You may partner with up to four others (**a group of max 5**) to submit a single write up in a single group. We encourage discussion with other students in class, even if they are not in your group. Each student must build a good understanding of all the questions even if they discuss and collaborate with others. Merely dividing the problems among group members and putting those together before submission will severely impact your learning, and we **strongly advice against it**. Furthermore, **blatant copying** from online or other resources is **forbidden**. If there are confusions or questions, post those on Piazza or see your TA or Instructor.*

Submission Instructions: *You must submit a PDF on LMS in the appropriate tab. This includes hand-written content (which you may scan and upload). There will be NO late days or extentions. Please write down the names and roll numbers on a front page. Make only one submission per group. We will NOT accept any hard-copy submissions.*

Question 1 [**4+2 marks**]

a) What design decisions went into making Bitcoin?

Solution: Bitcoin has an anarchic model of governance, with open network access and decentralized transaction processing. Bitcoin has internal reference model with intrinsic monetary incentives.

b) What are the consequences of these design decisions with regards to scalability?

Solution: Bitcoin isn't scalable as it can only handle 3 to 4 transactions per second whereas visa can handle more than 40,000 tps. There exist various solutions such as increasing the block size or decreasing block generation time but these solutions come with their own unique challenges.

Question 2 [**3+3 marks**] Suppose Taleef started a blockchain, Dripcoin which is exactly the same as the Bitcoin block (with the same average block generation time), except that the blocksize in Dripcoin is 4MB. Please use 1M = 10^6 for all calculations.

a) Assuming one transaction is 50B, what is the average transaction per second (TPS)?
Solution: 10 mins $\rightarrow$ 60s $\times 10 = 600s$

$4M \rightarrow \frac{4\times10^6 B}{50B}$ = 80000 txns per 10mins
$\rightarrow \frac{80000}{600}$ = 133.33 tps

b) Suppose Bitcoin started on day 0 and Dripcoin started on day 8000. On what day is the size of the Dripcoin blockchain greater than that of Bitcoin? (Bitcoin block size = 1MB)
Solution: Blocks per day in BTC and DRIPCOIN = 1 block per 10 mins

= 6 blocks per 1 hour

$= 144$ blocks per 24 hour
$\rightarrow x(144)(1) < (x - 8000)(144)(4)$
$144x < 576x - 4608000$
$576x - 144x > 4608000$
$432x > 4608000$
$x > 10666.67$
x = 10667^{th} day

Question 3 [3 marks] What class of numbers would you input into ScriptSig to unlock the following ScriptPubKey:
3
OP MUL
5
OP ADD
OP DUP
OP MUL
2
OP MOD

Solution: The thing we are checking is whether $(3x + 5)^2$ mod 2 is even or not. Even squared is even and odd squared is odd. So, we will input even numbers to get an odd number at the end ($(3x + 5)^2 mod 2 == 1$ when x is even).

Question 4 [3 marks] Write a Bitcoin ScriptPubKey that unlocks if you input a number x such that $5x^3 + 3x^2 + 32 = 59$. **Solution:**

OP_DUP
OP_DUP
OP_DUP
OP_DUP
OP_MUL
OP_MUL
$< 5 >$
OP_MUL
OP_SWAP
OP_DUP
OP_MUL
$< 3 >$
OP_MUL
$< 32 >$
OP_ADD

OP_ADD
$< 59 >$
OP_EQUAL

Question 5 [5 marks] Suppose there is a UTXO with a value of 1 BTC that is locked to Ahsan. Ahsan sends Hashim 0.4BTC. Ahsan then sends Hasnain 0.3 BTC. Hasnain further sends 0.2BTC to Hashim. Assuming there were no other transactions (or fees), no other UTXOs, and that they did not send any change to the miner but to themselves:

a) What is the total number of UTXOs at the end? **Solution:**

Tx1:
Inputs:
Ahsan 1BTC
Outputs:
Hashim 0.4 BTC
Ahsan 0.6 BTC (used)

Tx2:
Inputs:
Ahsan 0.6 BTC
Outputs:
Hasnain 0.3 BTC(used)
Ahsan 0.3 BTC

Tx3:
Inputs:
Hasnain 0.3 BTC
Outputs:
Hashim 0.2 BTC
Hasnain 0.1 BTC

Total UTXO's = 4, one for Ahsan and Hasnain. Two for Hashim.

b) What were the total number of (spent + unspent) outputs in this exchange (excluding the initial 1 BTC)?. **Solution:**

Spent + unspent = 4 + 3 = 7

c) Now, suppose Hashim sends 0.35 BTC to Ahsan. What is the total number of UTXO's now? **Solution:**

Ahsan has 0.3 + 0.35, Hassnain has 0.1, Hashim has 0.05 + 0.2
Total of 5 UTXOs

d) Suppose now that an extra output of 0.1 BTC is made for the miner for each transaction. At the end of the above transactions, what will be the amount of BTC that Ahsan, Hashim, and Hasnain have remaining? How much will have the miner gained from just the transaction fees for these? **Solution:**

Ahsan = 1 - 0.4 - 0.1 - 0.3 - 0.1 + 0.35 = 0.45 BTC
Miner = 0.1 + 0.1 + 0.1 + 0.1 = 0.4 BTC
Hasnain = +0.3 - 0.2 - 0.1 = 0 BTC
Hashim = +0.4 + 0.2 - 0.35 - 0.1 = 0.15 BTC

Question 6 [2+3+14 marks] Mustafa is a Bitcoin miner he mined a Bitcoin block with a block number of 500,000 and a bits field of 402,691,653.

a) What is the target hash? How many leading zeros are there?

Solution: : Bits = 402,691,653 = 0x18009645
The target hash is 0x18 = 24 bytes long or 48 nibbles long. The total number of bytes in SHA256 is 32 bytes = 64 nibbles. So, the number of leading zeros are 64-48 = 16 leading zeros.
The target hash is thus 16 leading zeros followed by 48 nibbles. The 48 nibbles begin with 0x009645 (we got this number as the last 6 nibbles of bits in hex)
The final target hash is: 0x0000000000000000009645000000000000000000000000000000000000000000

b) How many tries does he need on average? **Solution:** :

Leading zeros in Bits = $18 * 4$
Hex: 0x9645 to Decimal: 38469
Maximum Hex: 0xFFFF to Decimal: 2^{16}
$2^{18*4} * \frac{2^{16}}{38469} = 2^{72.76186}$

c) What is the block difficulty? **Solution:** :

The difficulty is always measured relative to the difficulty Satoshi originally picked. He had a target hash of $2^{223.9978}$ or average tries = $2^{32.0022}$.
We divide and get $\frac{2^{72.76186}}{2^{32.0022}} = 2^{40.76}$
or Simply $\frac{2^{224}}{2^{183.2314}} = 2^{40.76}$
For the following parts, assume that the average machine can compute 5TH/s. It also has a power consumption of 4000W. Also assume that the average Bitcoin block has a size of 1 MB and takes 10 minutes to be mined on average.

d) How many total machines are there?

Solution:
We know that all the machines must collectively go through $2^{72.76186} = 8.007 * 10^{21}$ hashes every ten minutes (on average).
The total number of Hashes per second must be $\frac{8.007*10^{21}}{600} = 1.3346 * 10^{19}$ Hashes per second.

Each machine computes $5 * 10^{12}$ Hashes per 600 second, so the number of machines is $\frac{1.334*10^{19}}{5*10^{12}} = 2^{21.347} = 2669200$ machines.

e) What is the yearly energy consumption of these machines? (assume all months = 30 days) **Solution:**

First, we find the total power consumption of these machines.
The number of machines is 2669200.
The total power consumption is $4000 \times 2669200 = 1.067 * 10^{10}$W.
The yearly energy consumption is $1.067 * 10^{10} 243012 = 9.2247552 * 10^{13}$Wh per year.

f) Assuming that the average size of a transaction in Bitcoin is 200 bytes, what is the average power consumed to mine a transaction? **Solution:**

First, we find the number of transactions per block. $\frac{1*10^6}{200} = 5000$ transactions per block.
The energy consumed to mine a block is $1.067 * 10^{10}\text{W} \times 600 = 6402$GWs or 1.778GWh.
The energy per transaction is $\frac{1.778*10^9}{5000} = 356$KWh.

g) Assume that the cost of electricity is \$0.1 per KWh around the world. Also, the reward for mining a Bitcoin block is 12.5 bitcoin (transaction fees are negligible). What is the minimum price of bitcoin (in \$) for it to be worthwhile to get a mining rig? **Solution:**

A mining machine will take $2^{72.76186}$ tries to mine a block (on average).
It will be running for $\frac{2^{72.76186}}{5*10^{12}} = 1.610^9$ seconds. The electricity consumed will be $1.610^9 * 4000 = 6.4 * 10^{12}$ Ws worth $0.1\frac{6.4*10^{12}}{36001000} = 177 * 10^3$.
So, 12.5 Bitcoin should be worth more than $177 * 10^3$.
One bitcoin is worth more than $\frac{177*10^3}{12.5} = 14.2 * 10^3$.

h) Suppose that it took the network 1173251 seconds to mine the previous 2016 blocks. The difficulty gets adjusted in block 501984 automatically. What is the new difficulty? **Solution:**

The difficulty gets adjusted every 2016 blocks. So, $501984 - 2016 > 500000$, so the difficulty before that was the same as calculated in part c.
The difficulty was $2^{40.76}$ or $1.862 * 10^{12}$. The new difficulty is $2^{40.76} * \frac{2016*600}{1173251} = 1.9 * 10^{12}$

Question 7 [5 marks] If Mustafa wants to buy some very expensive khokha chai from khokha-Uncle (for 20 BTC). khokha-Uncle is requesting a Bitcoin advance before sending the chai. But, Mustafa is concerned that if he sends the BTC and khokha-Uncle refuses to give the pricey khoka chai, he will lose the BTC he sent. In the event of a dispute, khokha-Uncle and Mustafa both trust Nameer to break the tie and award the BTC to the rightful party. Create a scriptPubKey that allows them to do this.

Solution:

To create a scriptPubKey that allows for Nameer to act as a trusted third-party mediator in the event of a dispute, we can use a multi-signature script. A multi-signature script requires the signature of multiple parties to unlock the funds.
The scriptPubKey can be created as follows:

OP_IF
$< khokha - Uncle's Sig >$
$< khokha - Uncle's PublicKey >$
OP_DUP
OP_HASH160
$< khokha - Uncle's Pubkeyhash >$
OP_EQUALVERIFY
OP_CHECKSIG
OP_ELSE
$< Mustafa's Sig >$
$< Nameer's Sig >$
$< 2 >$
$< Mustafa's PublicKey >$
$< Nameer's PublicKey >$
$< 2 >$
OP_CHECKMULTISIG
OP_ENDIF

In this scriptPubKey, khokha-Uncle's public key is used to verify his signature if he sends the pricey khoka chai as expected. Otherwise, if there is a dispute, the funds can only be spent with the signatures of both Mustafa and Nameer.

Question 8 [5 marks] Public keys are used on the Blockchain to somewhat protect the anonymity of senders and receivers. However, it is often the case that one uses one transaction to extrapolate the identity of multiple users. For example, if you pay Najeeb in Bitcoin to give you extra marks, Najeeb now knows your public key is tied to you in particular and can see what other transactions you have made. He can now start guessing the recipients public keys too, using the information that you were the sender. This is why people try to keep various public keys. Consider the following Bitcoin transaction:

Input	Output
2 BTC - Pk_a	1 BTC - Pk_β
2 BTC - Pk_c	3 BTC - $Pk_ə$
0.5 BTC - Pk_e	0.5 BTC - Pk_a

Let's say you are a Bitcoin detective who wants to identify how Jafri spends his money, and know that Pk_a (Public Key A) surely belongs to Jafri. What are the possible scenarios this transaction is showing you (think of who could be behind the senders and receivers, and why this is the case)? Give at least 5 scenarios. Secondly, do you think there are actor(s) (Jafri or anyone else) with more than one public key in this transaction? Why/why not?

Solution:

1. Jafri owns Pk_a, Pk_c, and Pk_e and is sending to other individuals b and d.
2. Jafri owns Pk_a only, Pk_e and Pk_c are owned by other individuals and they are sending it to other people (and to Jafri too).
3. Jafri owns Pk_a only and Pk_b, Pk_c, Pk_d, Pk_e are all owned by someone else and in the end Jafri receives 0.5 BTC and you cannot tell who owned Pk_b, Pk_c, Pk_d, and Pk_e since they covered their tracks.
4. Jafri owns Pk_a, Pk_d, Pk_e and someone else owns Pk_b and Pk_c. But no money was transferred overall between the two individuals and they just made it difficult to predict who they are.
5. Jafri owns all the public keys and wants to just trick everyone.

Question 9 [5 marks] Mining is a process in which we try to find a hash value that meets a certain criteria (less than the current difficulty of the blockchain) by changing the nonce value which changes the hash value of the block. There are on average 2500 transactions in a Bitcoin block. We can just change the order of those transactions and it will also produce a new hash value (avalanche effect). Then why do we still need the nonce value field? Give an example.

Solution:

A BTC block may have very few transactions; for example, a lot of the early blocks had only one transaction. In situations like this, it may be impossible to rearrange the transactions and get a valid hash for the block.

Question 10 [15 marks] Your TA Ahsan wants to design a ScriptPubKey. He wants to design a ScriptPubKey in a way that takes three numbers that gets unlocked if three numbers in the scriptSig are *pythogorean triples* and in **no** other circumstance.

Note: The input is pushed to the stack (a b c) and the starting state of the stack is c (top of the stack) b a (bottom of the stack).

Solution:

OP_DUP
OP_MUL
OP_SWAP
OP_DUP
OP_MUL

OP_ROT
OP_DUP
OP_MUL
OP_ADD
OP_EQUAL

Question 11 [5 + 5 marks] Bitcoin uses and stores the merkle tree root hash in the block header. You are Bitcoin miner and while mining a block you want have to make a merkle tree and add it's root hash to the header. Suppose you want to add 4 transactions to the block you want to mine (transactions are shown below).

txn1 = "10 Bitcoins from Class to Ahsan"
txn2 = "1 Bitcoin from Ahsan to Humza"
txn3 = "0.5 Bitcoins from Ahsan to Ayesha"
txn4 = "5 Bitcoins from Ahsan to Sara"

a) You are required to make a merkle tree as discussed in class using the above four transactions. Explain every step while making the tree e.g how you are concatenating the hashes. Also write the hashes of every node inside that node (you can shorten the hash such as 123...345). Note: Use SHA256 for calculating the hashes

Solution:

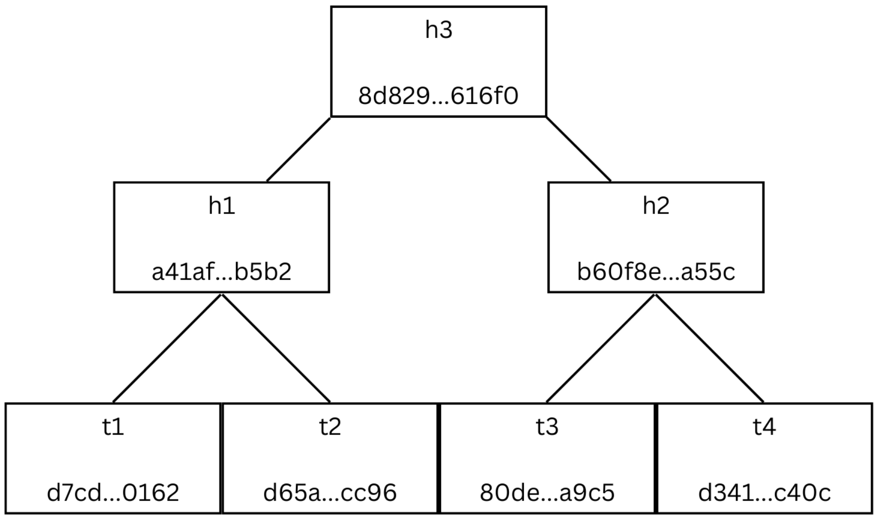
For merkle tree, hash of transactions $\rightarrow$
$hash(tx1) = d7cd...0162$
$hash(tx2) = d65a...cc96$
$hash(tx3) = 80de...a9c5$
$hash(tx4) = d341...c40c$

h_1 = hash(hash(tx1) + hash(tx2))
= a41af...b5b2

h_2 = hash(hash(tx3) + hash(tx4))
= b60f8...a55c

h_3 = hash(h1+h2)
= 8d829...16f0

Given below is the merkle tree $\rightarrow$



b) Suppose another miner Fatima wants to verify a transaction and she has your merkle tree. What is the time complexity of this operation and what hashes will she need to compute to verify that transaction. Give an example from you tree in part a to get full marks.

Solution:

In a Merkle-tree, verifying a transaction involves simply ascending the tree from the leaf node. We have to calculate the hash of two adjacent nodes and verify whether their parent node has a hash equal to the concatenation of the previously calculated hashes. If we reach the Merkle-root, and no discrepancies are found, the transaction is valid. Since the number of checks is $log_2(n)$ we can verify a transaction in O(log(n)).

Question 12 [12 marks] Bitcoin does not allow immediate transaction finality. That means that you can't be sure any transaction your receive is valid until it is in a block with 4-5 confirmations.

However, in certain circumstances, it is possible to have *instantaneous* transaction finality *after* one transaction has been confirmed. Moreover, all of this happens off-chain (except the initial transaction). Read up on the bitcoin lightning network and explain how it allows this? You should write down the script and explain how Alice and Bob may do instantaneous transactions.

Solution: Lightning network transactions involve using Hashed Time-Lock contracts. The basic idea is that Alice and Bob both supply a certain amount of Bitcoin as the output for a transaction. This total amount becomes the capacity of the channel. Once this transaction is on the chain, we can do instantaneous transactions by using *script* to punish anyone trying to be malicious.

This link has details about the script which you may use and I will not replicate here.
`https://blog.scottlogic.com/2016/06/16/bitcoin-redeem-scripts.html`

Question 13 [12 marks] Bitcoin allows you to define the script that must be satisfied by using only its hash. This is a scheme called Pay-to-Script-Hash. Explain how this scheme works and give an example of a case in which Alice wants to lock 5 BTC to Bob's pubkey.

Solution: Let's suppose Alice wants to transfer 5 BTC to Bob. Alice makes a transactions in which the output doesn't specify the pubkey-hash etc directly. Instead, she makes a simple output which verifies whether the corresponding scriptSig has the right redeem script *and* the inputs to unlock the same redeem script.

Here would be the scriptSig, scriptPubKey and redeemScript for the example asked:
redeemScript:
OP_DUP OP_HASH160 <Bob's pubkey hash> OP_EQUALVERIFY OP_CHECKSIG
scriptSig:
<Bob's signature> <Bob's Pubkey>
scriptPubKey:
OP_HASH160 <Redeem Script hash> OP_EQUAL

Question 14 [4+4+4+8 marks] These questions are related to bitcoin signatures:

a) When computing the signature, why is signing *just* the parent transaction a bad idea?

Solution: The output of *this* transaction must also be guarded by the signature. Otherwise, someone may modify the output of a transaction while keeping it valid, potentially stealing someone's BTC.
Bitcoin allows you to pick which parts of a transaction should be signed. These are specified with particular "sighash" flags.

b) What does $SIGHASH\_SINGLE$ do and when is it useful?

Solution: This scheme ensures that all the inputs of a transaction are guarded as well as *one* output (the one with the same index as the input). This scheme can be useful when you are making a multi-input transaction and only want to transfer a small fraction of the BTC to yourself and do not care where the remaining BTC goes.

c) What does $SIGHASH\_NONE$ do and when is it useful?

Solution: This scheme ensures that all the inputs of a transaction are guarded. This means that once you create a transaction, anyone can create the output for this transaction and publish it. This scheme can be useful if you are trying to be charitable.

d) You want to gather 50BTC in order to buy food for your entire workspace. You know the (very public) bitcoin address of a popular restaurant that all your colleagues would be willing to eat from. However, you do not have 50BTC – you must ask your colleagues (such as Bob) to donate. Bob, however, is worried that he will donate the BTC to the restaurant and he will lose his bitcoin if the total 50BTC required are not collected. This is because the transaction is irreversible and the restuarant may not be trusted to return the Bitcoin.

Considering this, how will you ensure that Bob gets his bitcoin back if the total goal of 50BTC is not reached?

Solution: I would use the "ALL—ANYONECANPAY" flag. The singular output would be BTC going out to the restaurant (worth 50 BTC) while the input would just be my contribution. Everyone in your workspace may make their individual contributions in the same transactions by creating inputs to the transaction. If the sum of contributions reaches 50 BTC, this transaction becomes valid but only the restaurant may have the BTC. Otherwise, this transaction is invalid and everyone may use their BTC elsewhere.

Question 15 [5 marks] In bitcoin, the difficulty is the ratio of the average number of tries required for a block compared to the average number of tries required for the genesis block. What would be an analogue in Ethereum (which relies on Proof-of-Stake)?

Solution: The analogue would be the ratio of the total Ethers being staked by all the validators right now compared to the total Ethers being staked by all the validators when the blockchain switched over to Proof-Of-Stake.