CS251
Cryptocurrencies and Blockchain Technologies
Final Exam – Monday, December 10, 2018

OPEN BOOK, OPEN NOTES, OPEN LAPTOP, CLOSED WiFi

Your Name: _____

SUNet ID: _____ @stanford.edu

Check if you require expedited grading: ☐
Check if you would like the exam returned to you via SCPD: ☐

In accordance with both the letter and the spirit of the Stanford Honor Code, I neither received nor provided any assistance on this exam.

Signature: _____

- The exam has 9 questions totaling 100 points.
- You have 150 minutes to complete them.
- Some questions may be much harder than others.
- All questions require you to justify your answer to receive full credit, even multiple choice questions for which you circle the correct answer(s).
- Keep your answers concise. We will deduct points for a correct answer that also includes incorrect or irrelevant information.

1	/15
2	/10
3	/10
4	/5
5	/10
6	/15
7	/10
8	/15
9	/10
Total	/100

1. [15 points]:

Short questions from all over.

A) Briefly explain what is an Ethereum reentrancy attack and why it can lead to loss of funds.

B) In class we presented an HD wallet as a way to generate many addresses that are unlinkable on the blockchain. The online wallet maintains a key – denoted in the lecture by (h, k_2) – that lets it construct all these addresses and calculate the user's overall balance. Recall that public key number i is defined as $\mathrm{p}k_i = h \cdot g^{\mathrm{HMAC}(k_2, i)}$ where $h = g^{k_1}$. If this key (h, k_2) is stolen, can the funds associated with any of these addresses be stolen?

Circle your answer: **yes** **no**

Brief justification:

C) In delegated proof-of-stake (DPOS) systems such as Tendermint, EOS, and BitShares, coin holders vote to elect a council that runs a closed consensus protocol and splits the mining rewards. What factors do rational coin holders typically weigh most heavily in deciding which candidate council members to vote for?

D) Zcash vs. Grin (Mimblewimble). Can an observer who sees a single snapshot of the shielded portion of the Zcash blockchain construct a transaction graph – a graph where the nodes are UTXOs and there is an edge from UTXO A to B if A was spent to create B? Also answer the same question, but with respect to Grin instead of Zcash.

Circle your answer: Zcash: **yes** **no** Grin: **yes** **no**

Brief justification:

E) Grin vs. Zcash continued. Same question as above, but with respect to an observer who collects all transactions as they are added to the mempool.

Circle your answer: Zcash: **yes** **no** Grin: **yes** **no**

Brief justification:

2. [10 points]:

Suppose that in Maker DAO, the pricing oracle (elected by MKR token holders) temporarily malfunctions and advertises that the price of ETH is $1,000, when in reality it is only $100.

A) How might an attacker exploit this situation to make money?

B) Assuming the error is corrected quickly enough not to destroy MakerDAO, who would bear the losses from such an attack and through what mechanism?

3. [10 points]:

Recall that the Howey test for a security asks: 1) is there an investment of money, 2) in a common enterprise, 3) with an expectation of profit, 4) relying on the efforts of a promoter or third party?

A) Suppose the owner of an orange grove decides to create 1,000 ERC-20 tokens, each entitled to 0.1% of the profits from the orange grove after expenses such as salaries for those picking the fruit. Is the SEC likely to consider this ERC-20 token a security?

Circle the best answer.

A Yes, it will be deemed a security. **If you circle this, briefly explain your answer.**

B No, it is unlikely to be deemed a security. **If you circle this, explain which parts of the Howey test it will fail.**

B) Suppose instead that the owner of an orange grove creates a bunch of non-fungible ERC-721 tokens, each corresponding to one orange tree. Owning the token for a particular tree grants you ownership of the tree, including both the responsibility of caring for the tree and the right to pick the fruit on the tree. Would the SEC consider one of these ERC-721 tokens a security?

Circle the best answer.

A Yes, it will be deemed a security. **If you circle this, briefly explain your answer.**

B No, it is unlikely to be deemed a security. **If you circle this, explain which parts of the Howey test it will fail.**

4. [5 points]:

Suppose that around a week ago the great firewall of China somehow partitioned the Bitcoin network and also undermined NTP (the network time protocol), leading some miners to mine blocks with timestamps several months in the future. As a result, there was a fork and the following two branches were created:

- Chain A has 500 blocks since the fork. Because of incorrect timestamps at the time of the last difficulty adjustment, the last 498 of these were mined with a target of 2^{182}.
- Chain B has 1,000 blocks since the fork. All have plausible timestamps, and all were mined with a target of 2^{184}.

Once the network partition heals and miners reset their clocks, which of the two chains will they continue to mine on?

Circle the best answer.

A Chain A will be accepted by all miners.

B Chain B will be accepted by all miners.

C Different miners will continue to mine on both chains.

Justify your answer:

5. [10 points]:

Recall that a selfish miner temporarily refrains from publishing mined blocks in an effort to get several blocks ahead of other miners, thereby causing other miners to waste effort mining orphan blocks. When a selfish miner is only one block S ahead of the public chain, if another miner mines and publishes a block O at the same height as S, the selfish miner immediately publishes S. Let γ be the probability that, when this happens, an honest miner will try to mine the next block on S instead of on O.

A) How can an attacker ensure that $\gamma \approx 1$ (in other words, S almost always overrides O) on today's Bitcoin network?

B) What's a backwards-compatible change in honest miners' behavior that would result in $\gamma \approx 0.5$.

6. [15 points]:

Ethereum lottery. Suppose the state of California decides to implement its lottery system as an Ethereum contract. The contract should support the following methods:

- buyTicket: any user can send the price of a ticket in Ether to the contract and the contract will record that user's address.
- doLottery: this method is called by the state lottery system once a week to randomly select that week's winner, if any. If there is a winner, the contract sends 90% of the pot to the winner's address and the remaining 10% rolls over to the following week. If there is no winner, the entire pot rolls over to the following week. Either way, the set of users resets to the empty set.

The contract proceeds in epochs, where each epoch is seven days, starting from the moment that the lottery contract is created. Say n users participate in a particular epoch. Each user is assigned an ID between 0 and $n-1$ in sequential order.

The doLottery method can only be called by the state of California within a 10 minute window after the end of each epoch. The method selects a winner by computing the current block hash modulo $2n$, and if the number matches a user ID, that user is the winner. Otherwise there is no winner, which happens with probability 1/2. The block hash modulo $2n$ can be computed as `(blockhash(block.number)%(2*n))`.

A) Write the solidity code to implement this contract. Use the back of this sheet.

B) Is this a good idea? Are there parties that can manipulate the lottery to greatly increase their chances of winning? If so explain how, if not explain why not.

7. [10 points]:

Suppose you want to create a compact proof that a particular address controls at least a certain number of coins (allowing those who verify the proof to enjoy a similar level security to what they would get by running a full node). As a function of chain length and number of addresses, can such a proof be more compact on Bitcoin or on Ethereum?

Circle the best answer.

A The proof can be smaller for Bitcoin.

B The proof can be smaller for Ethereum.

C They should be the same down to constant factors such as the size of particular data structures.

Justify your answer

8. [15 points]:

Multisig vs. threshold sigs. Let $\mathbb{G} = \{1, g, g^2, \ldots, g^{q-1}\}$ be a finite cyclic group of prime order q with generator g. Let H be a hash function $H : \mathcal{M} \to \mathbb{G}$. In a signature scheme called BLS the secret key x is a random number in $\{0, 1, \ldots, q-1\}$ and the public key is $pk = g^x \in \mathbb{G}$. A signature on message $m \in \mathcal{M}$ is simply $\sigma = H(m)^x \in \mathbb{G}$.

A) 3-out-of-3 signing. To protect its signing key x, Bob can split up x into three shares and distribute the shares among three parties as follows: choose three random numbers $x_1, x_2, x_3 \in \{0, 1, \ldots, q-1\}$ such that $x = x_1 + x_2 + x_3 \bmod q$. Bob deletes its local copy of x. To sign a message m each party computes $\sigma_i = H(m)^{x_i}$ for $i = 1, 2, 3$ and sends this partial signature to Bob. Explain how Bob obtains the signature on m from these partial signatures.

B) Let's generalize the approach above to 3-out-of-5 signing with five parties called P_1, P_2, P_3, P_4, P_5. As in Bitcoin's multisig, we want the signature to identify the subset of three parties that signed the transaction.

Bob does the following: he generates $\binom{5}{3} = 10$ public keys $\text{pk}_i = g^{x_i}$ for $i = 1, \ldots, 10$, one public key for each subset of three parties. Say, pk_1 corresponds to the subset $\{P_1, P_2, P_3\}$ and say pk_8 corresponds to the subset $\{P_2, P_4, P_5\}$.

Next, for $i = 1, \ldots, 10$ Bob applies 3-out-of-3 sharing as in part (A) to each of the secret keys x_i to get $x_i = x_{i,1} + x_{i,2} + x_{i,3} \bmod q$. He gives the share $x_{i,j}$ to party number j (for $j = 1, 2, 3$) in subset number i (so, for pk_8 party P_2 gets $x_{8,1}$, P_4 gets $x_{8,2}$, and P_5 gets $x_{8,3}$). Each of the five parties now has one share for each subset of size 3 that it belongs to (each party has 6 shares total). When three parties want to sign a message m, they sign using the three shares that correspond to their subset. For example, P_2, P_4, P_5, will sign using $x_{8,1}, x_{8,2}, x_{8,3}$.

What information needs to be written to the blockchain to use this 3-out-of-5 signing approach as a replacement for Bitcoin's multisig?

Hint: observe that the funding transaction can contain the Merkle root of a Merkle tree that has the 10 public keys $\text{pk}_1, \ldots, \text{pk}_{10}$ as its leaves. What signature data is needed in the spending transaction to validate the transaction?

C) Suppose the Merkle tree is implemented using SH256 (32 byte hashes), that each ECDSA or BLS public key is 32 bytes, and that each ECDSA or BLS signature is 48 bytes. Does the method used in part (B) result in more or less data written to the blockchain compared to Bitcoin's multisig method for 3-out-of-5 signing?

D) Does your conclusion from part (C) hold for t-out-of-n signing for all t and n? Or is there a t and n where your conclusion would change?

E) When using the method from Part (B), what is the running time for creating the funding transaction in terms of n and t? Is this practical for, say, $n = 100$ and $t = 50$?

9. [10 points]:

Synchronizing the mempool across miners. Miners A and B each have a set of transactions in their mempool. Suppose that miner A's set is a superset of miner B's. Miner A wants to send to B the transactions that B is missing. The problem is that A does not know which transactions B is missing.

A) Suppose B is only missing one transaction. Show that A can send a single 32-byte message to B that quickly lets B identify the missing transaction hash. B will send the missing transaction hash to A and A will send back the transaction data. Hint: think of computing the xor of all the transaction hashes in A's mempool.

B) Suppose B is missing k transactions, for some small k. There is a simple algebraic solution for sending the k missing transactions to B, but here we will look at an interactive solution instead. Show that A and B can engage in a multi-round protocol, where n is the total number of transactions in A's mempool, to identify the k missing transaction hashes. In each round A sends k hash values to B and B responds with k hash values.
Hint: In the first round try applying the method from part (A) twice – once to A's transactions in the lower half of the hash range and once to A's transactions in the upper half.