Midterm

Your Name: _____ **UVa Computing ID:** _____

For this exam, you should **work alone**. You may not use any resources other than your own brain, body, and a simple writing instrument.

For each question, write a correct and clear answer in the space provided. The space provided is more than sufficient for a full-credit answer. If you need to use additional space you may use the backs of pages, but make sure your answers are clearly marked.

Educating Congress Critters

1. Explain, in language and diagrams a typical Congressperson could understand, what it means for a bitcoin transaction to have "2 confirmations"?

The next three questions each gives an excerpt from the Congressional Research Service report on bitcoin. For each excerpt, identify at least one incorrect statement and explain why it is technically incorrect (or at least highly misleading).

2. "With a Bitcoin transaction there is no third-party intermediary. The buyer and seller interact directly (peer to peer), but their identities are encrypted and no personal information is transferred from one to the other. However, unlike a fully anonymous transaction, there is a transaction record."

3. "Cryptographic techniques enable special users on the bitcoin network, known as *miners*, to gather together blocks of new transactions and compete to verify that the transactions are valid—that the buyer has the amount of Bitcoin being spent and has transferred that amount to the seller's account. For providing this service, miners that successfully verify a block of transactions are rewarded by the network's controlling computer algorithm with 25 newly created Bitcoins."

4. "In order to mine and validate a new block of transactions, miners compete to solve a difficult math problem. The miner that solves the problem first validates the transactions in the block and broadcasts his or her proof-of-work to the bitcoin network. Other miners in the network check the successful miner's results. If the miner's work is found to be correct, he or she is rewarded by the system with 25 new bitcoins."

Digital Signatures

Assume H is a cryptographic hash function that provides strong pre-image resistance and collision resistance.

Assume S is cryptographically strong signing algorithm with verification function V. Thus, for any key pair, $(pub, priv)$, the owner of the private key can sign a message X by computing $s = S_{priv}(X)$, and someone with the corresponding key can verify by computing $V_{pub}(s, X)$ which outputs True if and only if s is a valid signature for message X and public key pub.

Consider a transaction where spender Alice who has keys $(pub_A, priv_A)$ sends the coin from output X to recipient Bob who has keys $(pub_B, priv_B)$.

For each question, answer if m would be a reasonable message for Alice to send to a public ledger to record the transaction. If m is not a reasonable way to record the transaction, explain what is wrong with it.

The $||$ operator is bitstring concatenation.

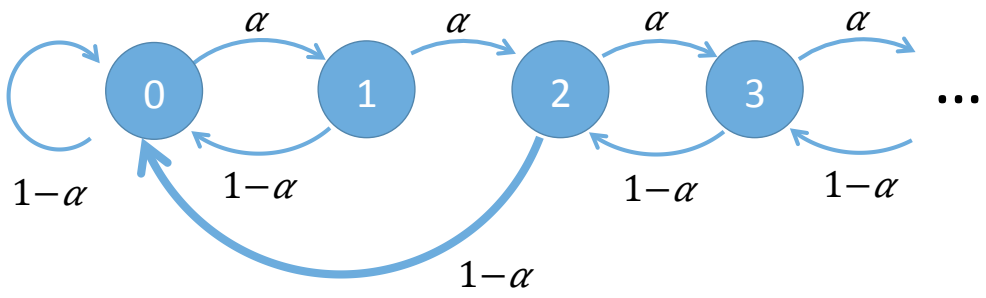
5. $m = H(S_{priv_A}(X||pub_B))||X||pub_B$

6. $m = S_{priv_A}(X||pub_B)||X||pub_B$

7. $m = S_{priv_A}(H(X||pub_B))||H(X||pub_B)$

Selfish Mining

The familiar state machine diagram below models the selfish mining strategy where each forward transition represents the probability α that the selfish mining pool with α-fraction of the network hashing power finds a block.



8. Explain what causes the model to follow the bolded edge from state 2 to state 0.

Transaction Scripts

For each of the following locking scripts: (a) provide an unlocking script that would unlock the protected value, and (b) if the locking script is unreasonable, explain what is wrong with it.

A summary of relevant bitcoin script instructions is given below, and an example.

Bitcoin Script Instructions

Opcode	Input	Output	Description
`OP_DATA` *data*	-	*data*	Pushes *data* on the stack.
`OP_DUP`	*a*	*a a*	Duplicates the top element of the stack
`OP_VERIFY`	*a*	-	If *a* is not **True** (**1**), terminates as Invalid.
`OP_RETURN`	-	-	Terminates as Invalid.
`OP_EQUALVERIFY`	*a b*	-	If *a* and *b* are not equal, terminates as Invalid.
`OP_HASH160`	*a*	H(*a*)	Pushes bitcoin address, RIPEMD160(SHA256(*a*)).
`OP_CHECKSIG`	*publickey sig*	**0** or **1**	(see below)

The `OP_CHECKSIG` pops two items from the stack, *publickey* and *sig*. It verifies that *sig* is a valid signature for the entire transaction for *publickey*. If the signature is valid, pushes **1**; otherwise, **0**.

The `OP_DATA` instruction can be used to push a bitcoin address, key, or signature on the stack (in the actual scripting language, different opcodes are needed based on the size of the data).

Example. As an example, here is the standard bitcoin pay-to-address script.

`OP_DUP`
`OP_HASH160`
`OP_DATA` *adr*
`OP_EQUALVERIFY`
`OP_CHECKSIG`

Your answer should be:

`OP_DATA` *sig* where *sig* is Sign(private key corresponding to public key *pubkey*, transaction)
`OP_DATA` *pubkey* where *adr* = *H*(*pubkey*)

9. Locking script:
   `OP_DATA` *pubkey*
   `OP_CHECKSIG`

(a) Unlocking script:

(b) Is this a reasonable locking script?

10. Locking script:
    `OP_HASH160`
    `OP_DATA` *adr*
    `OP_EQUALVERIFY`
    `OP_CHECKSIG`

(a) Unlocking script:

(b) Is this a reasonable locking script?

Murky Trees (Bonus Questions)

Bonus questions. You are not expected to answer these, but if you have more time you can impress us by answering the following questions.

To demonstrate its solvency, a bitcoin exchange may need to publish a proof of its total liabilities (that is, the amount of funds it owes to customers). One proposal for doing this is by recording all of the customer accounts in a Merkle tree where each leaf records one account:

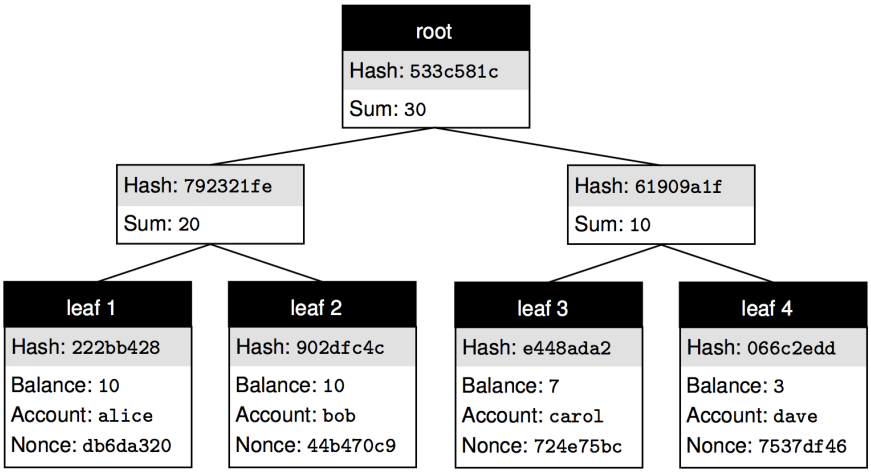
$$v_i = balance_i$$
$$h_i = H(ID_i||balance_i||nonce_i)$$

and each interior node j has a value that is the sum of the balances of its children, and a hash that incorporates that sum and the hashes of its children:

$$v_j = v_{left} + v_{right}$$
$$h_j = H(v_{left} + v_{right}||h_{left}||h_{right})$$

You can assume the bitstring concatenations are safe since each element has a known, fixed length there is no ambiguity about which bits are part of which element. You can also assume account balances must be positive (otherwise the exchange could add a fake account with a negative balance to the tree to reduce its apparent liability).

For example, the tree below records four accounts with total liability of 30:



(In an actual tree, the acounts would be randomly permuted, so no one knows which account corresponds to which leaf node.)

The goal of this protocol is for an exchange to prove that total liability is the value recorded in the root of the tree, without leaking information about individual accounts. The individual account owners should be able to verify that their own account is correctly included in the tree (without learning too much about other accounts). If all of the individual account owners verify their accounts, the exchange cannot reduce its total liabilities without getting caught.

11. In order for carol (leaf 3) to verify her account is included in the tree, in addition to the published root (which includes the hash and sum of the root node) and knowing her own account balance, what does she need to obtain from the exchange?

12. If the exchange wants to hide its true liabilities, what inconsistent but verifiable trees could it send to each client? (Hint: it can hide the value in Bob's account from every verifying client (other than Bob), and hide the value in Alice's account from Bob, so all clients see a total liability of 20 instead of 30.)

13. Suggest a simple way to modify the protocol to fix this problem.