CSCI-GA.3033-107: Cryptography of Blockchains **Spring 2024**

Homework #1

Due: 11:59pm on Sunday, **Feb 11, 2024**
Submit via Brightspace (each answer on a separate page)

Problem 1. Proof-of-Work hash functions. Let $H : X \times Y \to \{0, 1, \ldots, 2^n - 1\}$ be the hash function for a proof of work scheme. Once an $x \in X$ and a difficulty level D are published, it should take an expected D evaluations of the hash function to find a $y \in Y$ such that $H(x, y) < 2^n/D$. Suppose that $X = Y = \{0, 1\}^m$ for some m (say $m = 512$), and consider the hash function

$$H : X \times Y \to \{0, 1, \ldots, 2^{256} - 1\} \qquad \text{defined as } H(x, y) := \text{SHA256}(x \oplus y).$$

Here $\oplus$ denotes a bitwise xor.

a. Show that this H is insecure as a proof of work hash. In particular, suppose D is fixed ahead of time. Show that a clever attacker can find a solution $y \in Y$ with minimal effort once $x \in X$ is published. Hint: the attacker will do most of the work before x is published.

b. Would $H(x, y) = \text{SHA256}(x) \oplus \text{SHA256}(y)$ work? Prove your answer, assuming that SHA256 acts as a random oracle.

c. Is $H(x, y) = x \parallel \text{SHA256}(y)$ a collision-resitant hash function? Prove your answer. (Note that the question is not asking about Proof-of-Work security, but collision-resistance.)

d. Given the hashing power of the Bitcoin network as of January 2024, how long would it take to find x such that $\text{SHA256}(x) = 0^{256}$ if every miner worked towards that goal? (Use reasonable assumptions to get an estimate.)

Problem 2. Binary Merkle Trees and Beyond: Alice can use a binary Merkle tree to commit to a list of elements $S = (v_1, \ldots, v_n)$ so that later she can prove to Bob that $S[i] = v_i$ using an inclusion proof containing at most $\lceil \log_2 n \rceil$ hash values. The binding commitment to S is a single hash value. Let H be the collision-resistant hash function used.

a. A Merkle tree constructed using H is a binding vector commitment if H is collision resistant. Prove this by showing that if an adversary can construct two opening proofs (v_i, i) and (v'_i, i) then we can break the collision resistance of the hash function.

b. Show that Merkle trees satisfy an even stronger property, i.e. that they have unique proofs. That is, given two distinct Merkle proofs π, π' for the same statement (v_i, i) and the same Merkle tree root, we can break the collision-resistant property of the hash scheme.

c. Is such a Merkle tree also a set accumulator? If yes, prove it. If no, explain why not.

d. Consider a k-ary Merkle tree where each node has k children (a binary Merkle tree is where $k = 2$). If the tree contains n elements, What is the length of the inclusion proof as a function of n and k?

e. For large n, if we want to minimize the proof size, is it better to use a binary or a ternary tree? Why?

Problem 3. Merkle Trees and Block Trees

a. Draw a Merkle tree with 5 leaves and describe how each node's value is calculated.

b. Let T be a Merkle tree with n nodes and T' be one with $n' > n$ nodes such that the leaves of T are a prefix of the leaves of T'. How would one prove this? Assume that n is a power of two.

c. What if n is not a power of two? You can give a brief description or a diagram (that is, your answer doesn't have to be too formal). Hint: you'll need to use less than n nodes.

d. To update a Merkle tree T by appending a leaf e, how much information about T would you need? A brief description or diagram will suffice. (You need not formally prove your answer.)

e. Let $h_1 \leftarrow h_2 \leftarrow \ldots \leftarrow h_n$ be a blockchain represented by headers h_i with h_n being the current head. Alice wants to prove to Bob that there exists a block with a certain header h in this chain, but Bob only knows that h_n is the head and knows nothing about the other headers. How can Alice convince Bob? How long is this proof?

f. Instead of a blockchain, imagine that the blocks are arranged in a "blocktree". The blocktree would be a Merkle tree whose leaves are the hashes of the headers of all blocks. State one advantage and one disadvantage of this method. (Hint: use the previous subparts.)

For the next two problems it's useful to recall the security definitions of signatures that was presented in class (note that we will discuss a weakness of this definition in Problem 5:

Definition 1 (Secure Signatures) *A Signature Scheme* Σ *is a tripple of algorithms* $\Sigma = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify})$. *We say the signature scheme is secure if for all polynomial time and query adversaries* $\mathcal{A}$

$$P\left[\begin{array}{c} \mathsf{Verify}(pk, m, \sigma) = \text{"}accepts\text{"} \\ \wedge \\ m \notin \mathcal{O}.M \end{array} \middle| \begin{array}{c} (pk, sk) \leftarrow \mathsf{Setup}(\lambda) \\ (M, \sigma) \leftarrow \mathcal{A}^{\mathcal{O}(sk)}(2^\lambda, pk) \end{array}\right]$$

Here $\mathcal{O}$ *denotes an oracle that on input* m *outputs a valid signature on that message. and* $\mathcal{O}.M$ *denotes the set of messages that the oracle was queried on.*
A one-time signature is a signature scheme where the adversary does not have access to $\mathcal{O}$.

Problem 4. One-time Schnorr: Consider the following modification of the Schnorr signature scheme:

- $sk = (x, r)$ and $pk = (Y = x \cdot G, R = r \cdot G)$
- $\mathsf{Sign}(sk, M)$ outputs $s = r + c \cdot x$, where $c = H(pk, M)$
- $\mathsf{Verify}(pk = (Y, R), \sigma = s, M)$ outputs $s \cdot G \stackrel{?}{=} R + c \cdot Y$, where $c = H(pk, M)$

a. Show that the scheme is a secure one-time signature scheme. Concretely, show that if there exists an adversary A_{OTS} that can forge one-time Schnorr, there exists an adversary A_{Schnorr} that can create arbitrary forgeries for the normal Schnorr signature scheme.

b. Show that given any two signatures on two different messages, you can extract the private key.

c. Assume you have two parties with public keys pk_1, pk_2 and a message M. Design a way for the parties to produce a signature σ_a that is the size of only one signature but shows that both parties (identified by their keys pk_1, pk_2) signed M. Hint: You might need to generate an additional challenge c'.

d. Prove that the signature scheme in part (c) is secure. The proof requires a special notion of "special soundness" and starts with the assumption that you have two aggregate signatures σ_a, σ'_a on the same message M but distinct challenges c'_1, c'_2. Using these and pk_1, pk_2, show that you can compute σ_1, σ_2 where each σ_i is a valid signature for message M and public key pk_i.

e. Is such a one-time signature scheme useful in a blockchain setting? Can you modify Bitcoin such that it still works using only a one-time signature scheme? What about Ethereum?

Problem 5. Randomized BLS: Let Sign be a signature scheme with private key x and public key $Y = x \cdot G \in \mathbb{G}$ with $\mathsf{Sign}(sk, M) := \sigma = (r, (x \cdot r) \cdot H_{\mathbb{G}}(M))$ where $r \leftarrow \mathbb{Z}_p \backslash \{0\}$, and $\mathsf{Verify}(pk, M, \sigma) := e(\sigma, G) \stackrel{?}{=} (r \cdot H_{\mathbb{G}}(M), y) \wedge r \neq 0$. Here $H_{\mathbb{G}}$ is a hash function that maps to group elements (as in BLS).

Consider the following pseudocode describing an exchange's withdrawal function. (Note that this is a function running on the exchange's server and *not* on a blockchain smart contract.)

```
request_withdrawal(amount,account,withdrawaladdr)
  If account.balance>=amount:
      lock account
      create tx sending amount to withdrawaladdr
      sig = sign_BLSR(sk,tx)
      txid = H(tx,sig)
      wait(timeout)
      Check blockchain
      if txid on blockchain then
           account.balance-=amount
           unlock account
      else
           notify user that tx failed
           unlock account
```

txid= $H(tx, \sigma)$ is the transaction id on the blockchin.

a. First, show that the signature scheme is secure assuming BLS is secure

b. Assume that the exchange uses this randomized BLS signature scheme. Can you construct an attack that allows a user to steal money from the exchange? (This really happened).

c. There are two possible mitigations. For the first, how would you strengthen the security definition of signature schemes such that vulnerable signature schemes are no longer considered secure?

d. For the second, how can we change the blockchain so that this attack does not occur anymore?