CS 491/591 Blockchains, HW1

Prof. Jared Saia, University of New Mexico

Due: Thursday, February 14

Note: Many problems in this hw are taken in whole or in part from the "Cryptocurrencies and Blockchain Technologies" class at Stanford (http://cs251crypto.stanford.edu/ cs251/)

1. In class we defined two security properties for a hash function, one called collision resistance and the other called puzzle-friendly. Show that a collision-resistant hash function may not be puzzle friendly. Hint: Let $H : X \times Y \rightarrow \{0, \ldots 2^n - 1\}$ be a collision-resistant hash function. Construct a new hash function $H' : X \times Y \rightarrow \{0, \ldots 2^m - 1\}$ (for m possibly larger than n), where H' is collision-resistant but not puzzle-friendly. To show H' is collision-resistant, you can show that whenever there is a collision in H, there is also a collision in H'. To show that H' is not puzzle-friendly, you can show that for some difficulty D (say 2^{32}), and for a fixed x, it is computationally easy to find a value y such that $H'(x, y) \leq 2^m/D$.

2. Recall the Fiat-Shamir public-key digital signature scheme we discussed in class. In this problem, you'll do a toy example of that scheme over the multiplicative group $Z_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. Note that the generator for this group is $g = 2$. Assume that Alice has private-key $x = 2$ and public key $y = 2^x$. Let the random target (or commitment) chosen by Alice be $t = g^7 = 7 \pmod{11}$, and let the random challenge chosen by Bob be $c = 2$.

   (a) What is the correct response, r that Alice will choose such that $g^r y^c = t$, and how does she compute it?

   (b) When the group size is large, what makes it hard for someone else to find the correct value for r?

   (c) If Alice wants to prove she knows x without relying on Bob, how should she choose c?

3. k-ary Merkle trees. Alice can use a binary Merkle tree to commit to a set of elements S= $\{T_1, ..., T_n\}$ so that later she can prove to Bob that some T_i is in S using an inclusion proof containing at most $\lceil \log n \rceil$ hash values. The binding commitment to S is a single hash value.

   In this question your goal is to explain how to do the same using a k-ary tree, that is, where every non-leaf node has up to k children. The hash value for every non-leaf node is computed as the hash of the concatenation of the values of all its children.

   (a) Suppose $S = \{T_1, \ldots, T_9\}$. Explain how Alice computes a commitment to S using a 3-ary Merkle tree. How does Alice later prove to Bob that T_4 is in S?

   (b) Suppose S contains n elements. What is the length of the proof that proves that some T_i is in S, as a function of n and k?

   (c) For large n, if we want to minimize the proof size, is it better to use a binary or a 3-ary tree? Why?

4. Can you solve Byzantine agreement in a synchronous setting over three nodes, one of which may be Byzantine? If so, how? If not, why not? Assume you can use digital signatures, but there is no initial setup of a public key infrastructure (PKI).

5. Recall the four properties of blockchains discussed in Section 1.3 of the paper "Analysis of the Blockchain Protocol in Asynchronous Networks". Assume Property 1 ("Consistency") holds. Then, Property 2 states:

   - "*Future self-consistence:* With overwhelming probability (in T), at any two points r, s the chains of any honest player at r and s differ only within the last T blocks".

   What happens if Property 2 is replaced with the following:

   - *Eventual Immutability:* With overwhelming probability (in T), for any honest player, blocks that are T hops away from the front of that player's chain will never be removed or altered in the future."

   Is this new "Eventual Immutability" property equivalent to Property 2 (assuming property 1)? If so, prove it. If not, give a counterexample showing the difference, and discuss which property is more desirable.

6. A signature scheme is said be *malleable* if for all messages m, a valid signature σ on m can be easily transformed into a different valid signature σ' on m. In this problem, you will show that malleable schemes can lead to security problems in a cryptocurrency.

   Consider a simple, newspaper-ad-based cryptocurrency blockchain, where transactions look like this:

$$\begin{aligned} \text{coin}_1 &\leftarrow \{\text{Create 1 Bitcoin (serial\#53401) for } PK_{Fred}\}_{SK_{Fred}} \\ \text{coin}_2 &\leftarrow \{\text{Pay H}(coin_1) \text{ to } PK_{David}\}_{SK_{Fred}} \\ \text{coin}_3 &\leftarrow \{\text{Pay H}(coin_2) \text{ to } PK_{Dan}\}_{SK_{David}} \end{aligned}$$

   Here the notation m_{SK} denotes a pair (m, σ) where σ is a signature on m generated using the key SK. Every day the paper publishes a classified ad that looks like: new-coins, H(yesterday's ad).

   Everyone using the system builds a database containing a hash of every coin ever published, along with one bit saying if the coin has already been spent. Nodes ignore blocks that contain a coin whose hash is already in the database (a duplicate hash) or spending a coin containing a spent or non-existent hash. For instance, an attempt to republish $coin_1$ would fail, as would an attempt to re-spend $coin_2$ such as:

$$\text{coin}_4 \leftarrow \{\text{Pay H}(coin_2) \text{ to } PK_{David}\}_{SK_{David}}$$

   Nevertheless, malleable signatures introduce a vulnerability. Show how someone might attack this system when we use malleable digital signatures. Hint: Assume a node may see a transaction before it is published in the newspaper.