CS 601.441/641 – Blockchains and Cryptocurrencies

Homework 1

Instructor: Abhishek Jain *Deadline: February 26; 2018, 11:59 PM*

1. (15 Points) Let H_1 and H_2 be hash functions out of which at least one is collision-resistant. Are the following constructions necessarily collision resistant? Prove your answer. Specifically, if you think it is not collision-resistant, give a counterexample. And if you think it is collision resistant, show that a collision in H would imply a collision in H_1 and H_2.

   (a) $H(x) = H_1(x)||H_2(x)$

   (b) $H(x) = H_1(H_2(x))||H_2(H_1(x))$

2. (15 Points) Let $h : \{0,1\}^{n+t} \to \{0,1\}^n$ be a fixed-length compression function. Suppose we forget a few of the important features of the Merkle-Damgård transformation, and construct a hash function H from h as follows:

   - Let x be the input.
   - Split x into pieces $y_0, x_1, x_2, \ldots, x_k$, where y_0 is n bits, and each x_i is t bits. The last piece x_k should be padded with zeroes if necessary.
   - For $i = 1$ to k, set $y_i = h(y_{i-1}||x_i)$.
   - Output y_k.

   Basically, it is similar to the Merkle-Damgård transformation, except we loose the IV and the final padding block.

   (a) Describe an easy way to find two messages that are broken up into the same number of pieces, which have the same hash value under H.

   (b) Describe an easy way to find two messages that are broken up into different number of pieces, which have the same hash value under H.

   Hint: Pick any string of length $n + 2t$, then find a shorter string that collides with it.

   Neither of your collisions above should involve finding a collision in h.

3. (15 Points) We say that a hash function $H : P \times S \to \{0,1\}^n$ is proof of work secure with difficulty d (say, $d = 2^{50}$) if for a randomly chosen puzzle $p \in P$, it is difficult to find a solution $s \in S$ such that $H(p||s) < \frac{2^n}{d}$ in time significantly less than $\frac{2^n}{d}$. You can verify on your own that if we view a hash function as a Random Oracle, then it indeed satisfies the proof of work security for any suitable choice of parameters.

   In this question we explore the relation between collision resistance and proof of work security. Show that a collision resistant hash function may not be proof of work secure.

   Specifically, let $H : P \times S \to \{0,1\}^n$ be a collision resistant hash function. Construct a new hash function $H' : P \times S \to \{0,1\}^{n'}$ (where n' may be greater than n) that is also collision resistant, but for a fixed difficulty d is not proof of work secure with difficulty d. This is despite H' being collision resistant. Also, explain why H' is collision resistant, that is, why a collision on H' would yield a collision on H.

4. (15 Points) Let H be a collision-resistant hash-function. We can use a Merkle tree to commit to a set $S = \{T_1, \ldots, T_n\}$ as follows:

   - **Commit Phase:** Set each element T_i as a leaf node and compute a Merkle hash tree (using hash pointers as discussed in class). Finally, the commitment to S is a single hash value (the root node).
   - **Open Phase:** We can prove to someone that some T_i is in S using a proof containing at most $\log_2(n)$ nodes. The proof consists of all the nodes on the path from the root to the leaf node containing T_i.

   Prove that this is a binding commitment to set S, i.e., once a polynomial-time prover commits $\mathsf{Com}(S) = C$ to a set S, he cannot open it to another set S' such that $\mathsf{Com}(S') = C$.

   Further, explain how would you construct a commitment scheme of the above form using a k-ary tree (that is, where every non leaf node has up to k children) instead of a binary tree. Now, suppose that we commit to a set S as defined above using this new commitment scheme. What would be size of the proof for proving that an element T_i is in S?

5. (15 Points) Let q be an N-bit prime. Let p be an L-bit prime such that $p - 1$ is a multiple of q. Choose g, a number whose multiplicative order modulo p is q. This means that q is the smallest positive integer such that $g^q = 1 \mod p$. Let H be a hash function. The DSA signature scheme is a $\mathsf{UF\text{-}CMA}$ secure signature scheme (in the Random Oracle model) that consists of a tuple of algorithms $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ that are defined as follows:

   - $\mathsf{Gen}(g, p, q)$: Sample a secret key $x \xleftarrow{\$} \{1, \ldots, q\}$. Set $sk = x$. Compute $y = g^x \mod p$, set $pk = y$. Output (sk, pk)
   - $\mathsf{Sign}(sk, m)$: Parse $sk = x$. Sample a random $k \xleftarrow{\$} \{2, \ldots, p-1\}$. Compute $r = (g^k \mod p) \mod q$. Compute $s = k^{-1}(H(m) + xr) \mod q$. Output $\sigma = (r, s)$
   - $\mathsf{Verify}(pk, \sigma)$: Parse $pk = y, \sigma = (r, s)$. Compute $w = (s^{-1} \mod q)$, $u_1 = (H(m) \cdot w \mod q)$, $u_2 = (r \cdot w \mod q)$ and $v = (g^{u_1} y^{u_2} \mod p) \mod q$. Accept if $v = r$.

   Consider a deterministic variant of the DSA signature scheme where the random string k is generated as a part of the secret key i.e., $(sk = (x, k))$. As a result, in the signing algorithm, the same k is used each time. Show that this variant of DSA is not $\mathsf{UF\text{-}CMA}$ secure.