Blockchain Foundations **NTUA, Spring 2026**

Theory Exercise 1

Due: TBD

Please solve all the problems below. The problems are worth equal points. After a genuine attempt to solve the homework problems by yourself, you are free to collaborate with your fellow students to find solutions to the theory homework problems. Regardless of whether you collaborate with other students, you are required to type up or write your own solutions. Copying homework solutions from another student or from existing solutions is a serious violation of the honor code. Please take advantage of the instructors' and TA's office hours. We are here to help you learn, and it never hurts to ask! The assignments should be submitted via Gradescope.

Problem 1

Let $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ be a collision-resistant hash function and define $G(x) = H(H(x))$. Use a reduction to show that G is a collision-resistant hash function.

Problem 2

Given a collision resistant hash function $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$, create a hash function $G : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ with the following properties:

1. G is collision resistant.
2. The first two bits of G are predictable (a bit is predictable if it can be successfully "guessed" with probability $\frac{1}{2}$ + non-negligible when the input is chosen uniformly at random).

Write out explicitly what this G function is, then define and prove the above two properties formally. For the first property, use a reduction.

Problem 3

Construct a correct but insecure signature scheme. Write the full implementation of your signature scheme in pseudocode. Prove its correctness and insecurity.

Problem 4

Consider the complete transaction graph ***accepted and stored in the database of an honest node*** illustrated in Figure 1. The circles represent transactions and the numbers within them are the txids. The outgoing edges are outputs, and the incoming edges are inputs. For each transaction, edges appear ordered from top to bottom. For this graph:

1. Identify the coinbase transactions.

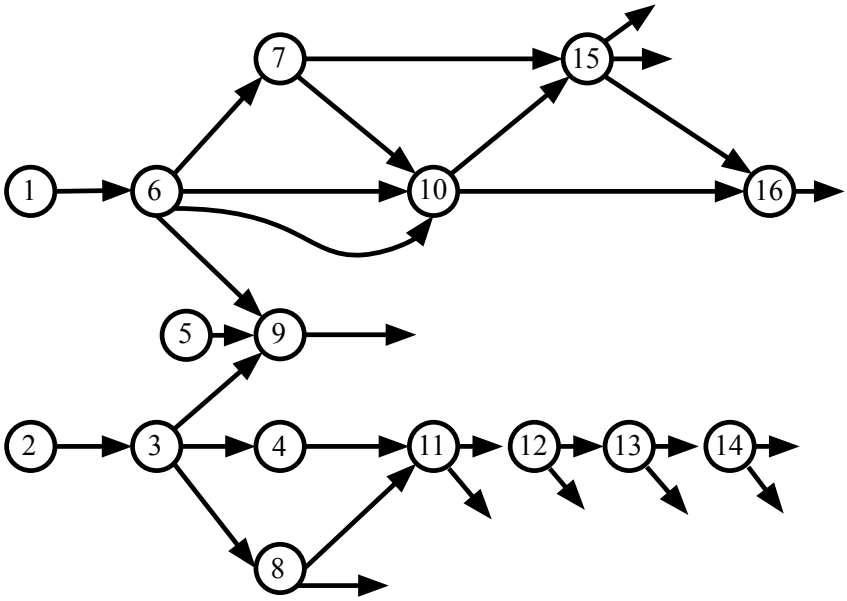

Figure 1: The transaction graph of Problem 4.

2. Identify the double spending transactions. For each double spending transaction, identify the transaction it is conflicting with.
3. Identify the UTXO set.
4. Identify the transactions for which the Weak Conservation Law does not hold.
5. Identify the outputs of transactions that are partially, but not fully, spent.

To identify transactions, use their txids. To identify an output, use outpoint notation. Note that transaction 6 has four outputs.

Problem 5

The honest proof-of-work algorithm begins by sampling a uniformly random κ-bit nonce, then iterates for a polynomial number of repetitions while incrementing the nonce. Prove that, if a constant number of honest parties execute this algorithm a polynomial number of times, the probability that they query the hash using the same nonce is negligible.

Reference

Some helpful definitions are provided below. For the full definitions, consult the lecture notes.

Definition (Collision Resistance)**.** A hash function $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ is *collision resistant* if for all PPT adversaries $\mathcal{A}$,

$$\Pr[\mathsf{collision\text{-}game}_{H,\mathcal{A}(\kappa)} = 1] = \mathrm{negl}(\kappa)\,.$$

The game is defined in Algorithm 1.

Algorithm 1 The collision-finding game for a hash function H.

function COLLISION-GAME$_{H,\mathcal{A}}(\kappa)$
    $x_1, x_2 \leftarrow \mathcal{A}(1^\kappa)$
    return $H_\kappa(x_1) = H_\kappa(x_2) \wedge x_1 \neq x_2$
end function

Definition (Correct Signature)**.** A signature scheme $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is *correct* if, for all $m \in \{0,1\}^*$, whenever $(sk, pk) \leftarrow \mathsf{Gen}(1^\kappa)$, we have that $\mathsf{Ver}(pk, m, \mathsf{Sig}(sk, m)) = 1$.

Definition (Secure Signature)**.** A signature scheme $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is *secure* if for all PPT adversaries $\mathcal{A}$,

$$\Pr[\mathsf{existential\text{-}forgery\text{-}game}_{(\mathsf{Gen},\mathsf{Sig},\mathsf{Ver}),\mathcal{A}}(\kappa) = 1] = \mathrm{negl}(\kappa)\,.$$

The game is defined in Algorithm 2.

Algorithm 2 The existential forgery game for a signature scheme (Gen, Sig, Ver).

function EXISTENTIAL-FORGERY-GAME$_{(\mathrm{GEN},\mathrm{SIG},\mathrm{VER}),\mathcal{A}}(\kappa)$
    $(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $M \leftarrow \emptyset$
    function $\mathcal{O}$(m)
        $M \leftarrow M \cup \{m\}$
        return $\mathsf{Sig}(sk, m)$
    end function
    $m, \sigma \leftarrow \mathcal{A}^{\mathcal{O}}(pk)$
    return $\mathsf{Ver}(pk, \sigma, m) \wedge m \notin M$
end function

Algorithm 3 The proof-of-work algorithm.

function PoW$_{H,T}$
    $\mathsf{ctr} \xleftarrow{\$} \{0,1\}^\kappa$
    while true **do**
        $B \leftarrow \mathsf{ctr}$
        if $H(B) \leq T$ **then**
            return B
        end if
        $\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$
    end while
end function