EE 374 - Blockchain Foundations
Final
Mar 21, 2023

1. The exam has 5 questions with a total of 134 points (with 20 bonus points that can take you up to 154 points). All subproblems of a problems with multiple parts are equally weighted. You have 3 hours to take the exam, and a 15 minutes grace period for submissions. Questions have different numbers of points so please allocate your time to each question accordingly.

2. Please write the answer to each question on a separate page and upload a photo or scan to Gradescope.

3. **All answers should be justified, unless otherwise stated.**

4. The exam is open-book, open-notes, open-internet. You may also use artificial intelligence assistance.

5. This exam is an assessment. You agree to abide by the Stanford Honor Code. You must work on this exam alone, with no collaboration with your fellow students whatsoever. You may not discuss the exam questions and answers with any other human while the exam is ongoing and until the deadline for exam submission has elapsed. Any collaboration with other people, inside or outside the class, is a violation of the honor code.

Good luck!

(60 points) Problem 1

For the following questions, choose the one most fitting answer among the four choices. No justifications are required for this problem. 2 points for a correct answer, 0 points for an incorrect answer. 0.5 point for leaving the answer blank. Knowing you don't know something has value.

1. The blockchain and cryptographic protocols we analyzed in class (hash functions, signatures, bitcoin backbone, etc.) can be broken:
   (a) By no adversaries at all.
   (b) By a polynomial probabilistic-time adversary, with non-negligible probability.
   (c) By a polynomial probabilistic-time adversary, with overwhelming probability.
   (d) By an exponential adversary, with probability 1.

2. Consider a second-preimage-resistant hash function $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$. How many collisions $x_1 \neq x_2$ such that $H(x_1) = H(x_2)$ *exist*?
   (a) None.
   (b) One.
   (c) A polynomial number of collisions.
   (d) An infinite number of collisions.

3. Which of the following adversaries would be detrimental to the existential unforgeability of a signature scheme?
   (a) An adversary who, given a tuple (sk, m, σ) of a secret key, message, and a signature such that $\mathrm{Ver}(pk, m, \sigma)$, where pk is the respective public key, finds a new message $m' \neq m$ and signature σ' such that $\mathrm{Ver}(pk, m', \sigma')$.
   (b) An adversary who, given a tuple (pk, m, σ) of a public key, message, and a signature such that $\mathrm{Ver}(pk, m, \sigma)$, finds a new message $m' \neq m$ and signature σ' such that $\mathrm{Ver}(pk, m', \sigma')$.
   (c) An adversary who, given a tuple (pk, m, σ) of a public key, message, and a signature such that $\mathrm{Ver}(pk, m, \sigma)$, finds a new signature $\sigma' \neq \sigma$ such that $\mathrm{Ver}(pk, m, \sigma')$.
   (d) None of the above.

4. For a verifier who knows the index and value of a leaf, what is the size of a proof-of-inclusion in a binary Merkle tree built using the $\kappa = 256$ bit hash function blake2s with 1024 leaves?
   (a) 320 bytes
   (b) 2560 bytes
   (c) 352 bytes

(d) 32768 bits

5. What is the relationship between safety and liveness of ledgers?

   (a) A safe ledger is always live.
   (b) A live ledger is always safe.
   (c) A ledger can be both unsafe and unlive.
   (d) A ledger is safe if and only if it is live.

6. There exists *no* PPT adversary controlling 70% of the mining power in a proof-of-work blockchain who can eventually break:

   (a) All parametrizations of Chain Quality, no matter what constant $\mu > 0$ and $\ell \in \mathbb{N}$ we choose.
   (b) All parametrizations of Common Prefix, no matter what constant $k \in \mathbb{N}$ we choose.
   (c) All parametrizations of Chain Growth, no matter what constant $\tau > 0$ and $s \in \mathbb{N}$ we choose.
   (d) All parametrizations of Liveness, no matter what constant $u \in \mathbb{N}$ we choose.

7. At the moment of a reorg, transactions in the abandoned chain are:

   (a) Discarded.
   (b) All placed in the mempool.
   (c) Placed in the mempool, as long as they can be applied on top of the new chain.
   (d) Placed within the blocks of the newly adopted chain so that they can become immediately confirmed.

8. Waiting k blocks before confirmation enables us:

   (a) To use the *Chernoff Bound* to bound the probability of failure to negligible.
   (b) To use the *Law of Large Numbers* to bound the probability of failure to zero.
   (c) To ensure that the honest parties win, in expectation.
   (d) To use the *Pigeonhole Principle* to ensure that the proof-of-work hash function is well-behaved.

9. No transactions are occurring on a blockchain network. A rational party will:

   (a) Stop mining blocks until more transaction traffic appears.
   (b) Keep mining empty blocks to reap the coinbase reward.
   (c) Re-include some previously confirmed transactions into new blocks it mines.
   (d) Create its own high-fee-paying transactions, and include them, to reward itself.

10. A protocol designer proposes replacing the proof-of-work inequality $H(B) \leq T$ with the inequality $H(B) \geq 2^\kappa - T$ everywhere.

    (a) This is fine.

    (b) This will work, but will make the blockchain insecure, as the probability of a successful query is no longer $\frac{T}{2^\kappa}$ in the Random Oracle model.

    (c) This does not make syntactic sense, as the hash $H(B)$ is always out of the designated range.

    (d) This is fine, but the T parameter must be adjusted accordingly to the value $T' = 2^\kappa - T$.

11. The Weak Conservation Law in the UTXO model ensures:

    (a) That money is never double spent.

    (b) That money remains scarce.

    (c) That a party only spends money which rightfully belongs to it.

    (d) That transactions cannot be reverted.

12. If you increase the mining difficulty $\frac{1}{T}$, then:

    (a) You increase liveness, but reduce safety

    (b) You increase safety, but reduce liveness

    (c) You increase both safety and liveness

    (d) You reduce both safety and liveness

13. The Streamlet protocol:

    (a) requires a honest supermajority to be safe because the quorum q has to be set at $2n/3$.

    (b) can relex the honest supermajority safety conditoon but at the expense of decreasing the liveness of the protoccol.

    (c) can relax the honest supermajority safety condition without decreasing the liveness of the protocol.

    (d) can relax the honest supermajority safety condition and increase the liveness of the protocol.

14. An SPV client needs to bootstrap from genesis, and is interested in retrieving *all* coinbase transactions in a chain with length $|\mathcal{C}|$, each block of which has α transactions. It will need communication complexity:

    (a) $\mathcal{O}(|C| + \alpha)$

    (b) $\mathcal{O}(\alpha|C|)$

    (c) $\mathcal{O}(|C| + \log \alpha)$

(d) $\mathcal{O}(|C| \log \alpha)$

15. A majority adversary in the proof-of-work bitcoin backbone model:

    (a) Can break the bounds of the *Patience Lemma*, because she can break the collision-resistance property of the hash function and mine arbitrarily quickly.

    (b) Can break the bounds of the *Patience Lemma*, because she can simulate mining in the future without actually performing any work.

    (c) Is bound by the confines of the *Patience Lemma*, as long as the execution is typical.

    (d) Is bound by the confines of the *Patience Lemma*, even if the execution fails to be typical.

16. The accounts-based model, as compared to the UTXO model, has:

    (a) A larger k parameter for Common Prefix.

    (b) A smaller k parameter for Common Prefix.

    (c) The same k parameter for Common Prefix.

    (d) The Common Prefix parameter set to $k = 1$.

17. During peer discovery in a peer-to-peer protocol, the initial set of peers is:

    (a) Hard-coded into the code of the node as an array of IP addresses.

    (b) Discovered by collecting IP addresses from the issuers of blockchain transactions.

    (c) Derived from the public keys in the coinbase transaction issued by the miners.

    (d) Requested from the user's ISP via an HTTPS exchange.

18. Setting the bitcoin backbone parameter ϵ, representing the acceptable Chernoff error, to 0 would cause:

    (a) Liveness to be lost, because the Chernoff interval λ would need to become infinite.

    (b) Blocks to be produced very slowly, as the block production rate f must also be reduced accordingly.

    (c) The Common Prefix parameter to become very small, allowing for instant confirmation.

    (d) None of the above.

19. In the UTXO model, a signature on the transaction ensures:

    (a) That the transaction is not a double spend.

    (b) That the transaction was issued by an honest party.

    (c) That the spending party is the rightful owner of the output spent.

(d) That the money has been produced correctly according to the macroeconomic rules of the chain.

20. Consider two independent worlds A and B in which the variable-difficulty proof-of-work protocol is executed. In world A the adversary plays honestly. In world B, the adversary is a selfish miner. The worlds are the same (in terms of the parameters n, t, q, Δ and so on) otherwise.

    (a) World A will tend to have a higher mining target T than world B.

    (b) World B will tend to have a higher mining target T than world A.

    (c) The two worlds will tend to have the same mining target T, as the total mining power devoted to the network is the same.

    (d) The two worlds will tend to have the same mining target T, because the selfish mining attack is undetectable.

21. In a well-configured proof-of-work system with $n = 10$, $t = 5$, an adversary playing honestly will cause the long-term chain quality to be:

    (a) 0

    (b) $\frac{1}{2}$

    (c) $\frac{1}{3}$

    (d) 1

22. Imagine a network for which suddenly, but temporarily, Δ becomes very large because the underwater Internet cable connecting the Americas to Europe is cut. During that temporary outage:

    (a) The proof-of-work protocol remains safe, but temporarily loses liveness.

    (b) The Streamlet protocol remains safe, but temporarily loses liveness.

    (c) The longest-chain proof-of-stake protocol remains safe, but temporarily loses liveness.

    (d) All of the above.

23. Imagine a network for which suddenly, but temporarily, half the parties went offline. The online parties are all honest. During this period:

    (a) The proof-of-work protocol remains safe, but temporarily loses liveness.

    (b) The Streamlet protocol remains safe, but temporarily loses liveness.

    (c) The longest-chain proof-of-stake protocol remains safe, but temporarily loses liveness.

    (d) All of the above.

24. The size of the header of a block:

(a) Is a constant.

(b) Is proportional to the chain size $|\mathcal{C}|$.

(c) Is proportional to the number α of transactions in the block.

(d) Is proportional to the logarithm $\log \alpha$ of the number of transactions in the block.

25. A ξ-*superblock* is a block B that satisfies $H(B) \leq \frac{T}{2^\xi}$ for some $\xi \in \mathbb{N}$. What is the probability that a given block B is a ξ-superblock, given that it is already a good block?

    (a) $\frac{1}{2^{\xi+\kappa}}$

    (b) $\frac{1}{2^\xi}$

    (c) $\frac{T}{2^\xi}$

    (d) $1 - \frac{T}{2^{\xi+\kappa}}$

26. The proof-of-stake longest chain is not accountable because:

    (a) Parties have no identities tied to the proof-of-stake puzzle.

    (b) Safety is based on a synchrony assumption.

    (c) Adversary parties cannot equivocate in the longest-chain protocol.

    (d) All of the above.

27. The proof-of-stake longest chain protocol:

    (a) requires a honest supermajority to be safe because adversary can equivocate.

    (b) requires a honest supermajority to be live because adversary can equivocate.

    (c) None of the above statements is correct.

    (d) Both of the above statements are correct.

28. The backbone proof of the common prefix property of the proof-of-work longest chain protocol cannot be reused for the proof-of-stake longest chain protocol because:

    (a) the chain growth lemma doesn't hold anymore.

    (b) the expected number of honest blocks during an execution is not propertional to the honest stake.

    (c) the expected number of adversary blocks during an execution is not proportional to the adversary stake.

    (d) the Chernoff bound cannot be used to bound the number of honest blocks during an exection.

29. The confirmation latency of a longest chain protocol:

    (a) increases when the honest advantage increases.

    (b) increases when the security parameter increases.

(c) increases when the network delay bound increases.
(d) more than one of the above statements is correct.

30. The confirmation latency of Streamlet:

(a) decreases when the honest advantage increases.
(b) decreases when the security parameter increases.
(c) decreases when the network delay bound increases.
(d) more than one of the above statements is correct.

(16 points) Problem 2

Consider a longest chain blockchain in the UTXO model. The protocol follows a confirmation rule with $k = 1$ (a transaction is considered *confirmed* if it is included in $\mathcal{C}[:-1]$). Consider the transaction graph in Figure 3. The coinbase reward is 50 units. The amount of each transaction's output is denoted as a number above it. Imaginary txids are denoted within each transaction circle. Note that transaction 7 has two outputs, transaction 10 has one input, transaction 10 has two outputs, transaction 11 has two inputs, and transaction 8 has one input.

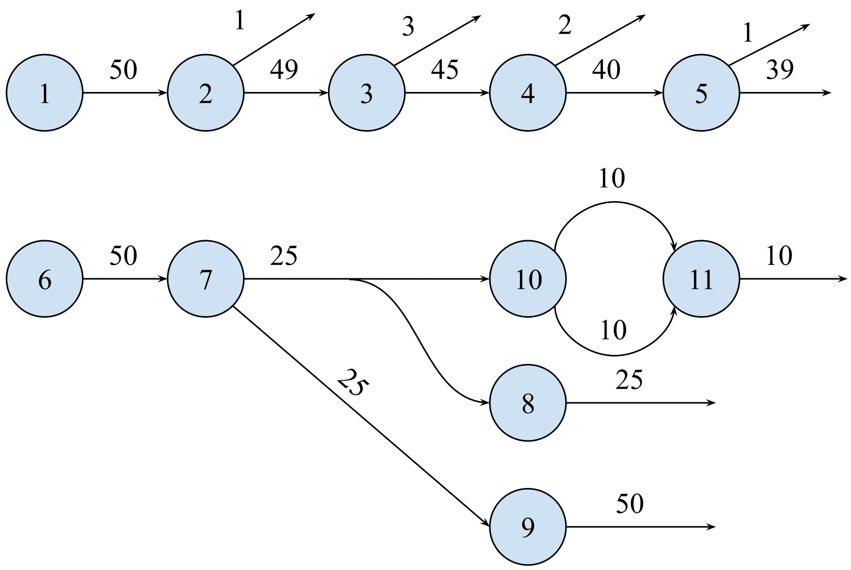

Figure 1: The transaction graph for Problem 2.

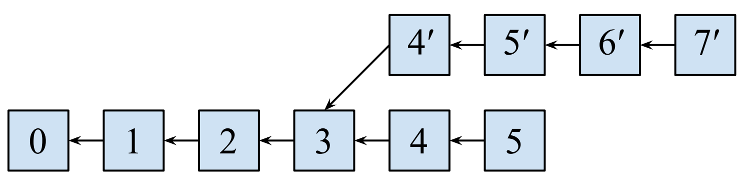

Figure 2: The block tree for Problem 2.

Consider the blocktree in Figure 2. In this system, blocks can be completely empty and coinbase transactions are not mandatory. The block contents are exactly the following:

- The genesis block 0 contains no transactions.
- Block 1 contains transaction 1.
- Block 2 contains transaction 6.

- Block 3 contains transactions $2, 3$.
- Block 4 contains transactions $4, 5$.
- Block 5 contains transactions $7, 8$.
- Block $4'$ contains transaction 7.
- Block $5'$ is empty.
- Block $6'$ contains transactions $10, 11$.
- Block $7'$ contains transactions $8, 5$.

Our timeline concerns an honest party P and is as follows:
Answer the following questions with regard to the timeline of a full node honest party P:

1. P has adopted the chain $[0, 1, 2]$ and has received transactions 1 and 6 by round 1. Subsequently, during round 1, he receives all transactions in the following order: $2, 3, 4, 5, 7, 8, 9, 10, 11$. What is his mempool at the end of round 1?
2. What is his UTXO set (according to the mempool) at the end of round 1?
3. In round 2, he receives blocks 3, 4 and 5. What is his mempool at the end of round 2?
4. What is his UTXO set (according to the mempool) at the end of round 2?
5. In round 3 he receives blocks $4'$, $5'$, $6'$ and $7'$. What his chain at the end of round 3?
6. What is his mempool at the end of round 3?
7. What is his UTXO set (according to the mempool) at the end of round 3?
8. What is his ledger at the end of round 3?

Use outpoints to designate the items in your UTXO set. Use a calculator such as Python and round your numbers to three significant digits.

(20 points) Problem 3

Consider the safety violation of Streamlet as shown in the figure below (the numbers within the blocks indicate their epochs). The upper fork has blocks notarized in three epochs $5, 6, 7$. The lower fork has a block from epoch 10 at the same height of the block from epoch 6.

1. Define what it means by the quorum of the Streamlet protocol.
2. Assuming the quorum of Streamlet is set to be $3n/4$. Argue that the safety violation cannot occur if the number of adversary parties is less than t, for the largest possible t you can provide an argument for.
3. Assuming the quorum of Streamlet is set to be $3n/4$. Argue that if the safety violation occurs, then the number of adversary parties that can be provably found to violate the protocol has to be at least t, and find the largest possible t.
4. Assuming the quorum of Streamlet is set to be $3n/4$. What is the largest number of adversary parties for which liveness of the protocol can be guaranteed? Explain.

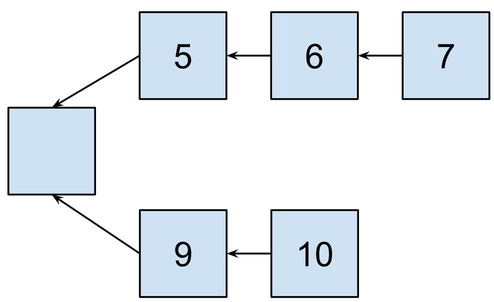

Figure 3: Safety violation. Upper fork has blocks from epochs $5, 6, 7$. Lower fork has block from epoch 10 at the same height as the block from epoch 6.

(18 points) Problem 4

In this class we studied three blockchain protocols: (i) proof-of-work longest chain, (ii) proof-of-stake longest chain, (iii) Streamlet. We have also introduced three protocol properties: (i) partition tolerance, (ii) dynamic availability, (iii) accountability.

1. Define the three protocol properties in terms of the basic properties of safety and liveness.

2. Explain which of the three protocols satisfies which properties.

3. Consider all the pairings of the three properties

   (a) partition tolerance and dynamic availability,

   (b) partition tolerance and accountability,

   (c) dynamic availability and accountability.

   For each pairing of the properties, argue either why it is impossible to have *any* protocol (not only the three we studied) that satisfies both properties, or give an example of a protocol that simultaneously achieves both properties.

(20 points) Problem 5

Consider a bitcoin backbone execution with $n = 10$, $t = 2$, $q = 1000$, $f = 0.01$, and $\kappa = 256$.

1. Calculate the honest advantage δ for the given parametrization.
2. Calculate the probability p that a query is successful.
3. Calculate the target T.
4. Use the balancing equation to calculate a reasonable ϵ.
5. Suppose the probability of an execution being typical is $1 - 2^{-\left(\frac{\epsilon^2\lambda}{3}+\kappa-327\right)}$. Setting the acceptable probability of failure to 2^{-256}, calculate a λ that satisfies this.
6. What is the probability that round 5 is successful?
7. What is the probability that round 5 is a convergence opportunity?
8. What is the probability that round 5 has exactly two adversarial, but no honest, successful queries?
9. What is a safe k for the confirmation rule?
10. What is a parameter for the velocity τ that guarantees chain growth for intervals at least $s = \lambda$?

Show the formula that you used for each calculation. Use a calculator such as Python and round your numbers to three significant digits.

(20 bonus points) Problem 6

Consider a *typical* Bitcoin Backbone execution correctly parameterized under honest majority in the static proof-of-work model.

1. An honest party P_1 broadcasts a new transaction tx_1 at round r_1 and another honest party P_2 broadcasts a new transaction tx_2 at round $r_2 \geq r_1$. Both transactions are included in the *same* block B, which eventually becomes stable and adopted by all honest parties. You correctly reason that, since the two transactions were both confirmed in the *same* block, they must have been broadcast closely together in time.

   Calculate an upper bound $d \in \mathbb{N}$ for the interval $r_2 - r_1$ such that we can rest assured that $r_2 - r_1 \leq d$. Your bound does not need to be tight, but needs to hold always for typical executions. Your bound can be a function of all the execution parameters $(\epsilon, \lambda, f, p, T, n, t, k, u, \mu, \ell, s, \tau)$. Prove that your bound holds.

2. An honest party P_1 broadcasts a new transaction tx_1 at round r and an honest party P_2 broadcasts a new transaction tx_2 at the *same* round r. However, the transactions make it on different blocks B_1 with height h_1 and B_2 with height $h_2 > h_1$ of the same chain that eventually becomes stable and adopted by all honest parties. You correctly reason that, since the two transactions were both broadcast at the *same* time, they must be included close together on the chain.

   Calculate an upper bound $d \in \mathbb{N}$ for the distance $h_2 - h_1$ such that we can rest assured that $h_2 - h_1 \leq d$. Your bound does not need to be tight, but needs to hold always for typical executions. Your bound can be a function of all the execution parameters $(\epsilon, \lambda, f, p, T, n, t, k, u, \mu, \ell, s, \tau)$. Prove that your bound holds.

Reference

Variables

- κ: The security parameter
- $\mathcal{A}$: The adversary
- Π: The honest protocol
- $\mathcal{G}$: The genesis block
- Δ: The network delay (in backbone, $\Delta = 1$)
- H: The hash function
- n: The total number of parties
- t: The adversarial number of parties
- q: In proof-of-work, the hash rate of one party per round; in proof-of-stake, the quorum.
- T: The mining target
- p: Probability of a successful query
- δ: The honest advantage
- k: Common prefix parameter
- μ: Chain quality parameter (the honest ratio of blocks)
- ℓ: Chain quality chunk length (in blocks)
- τ: Chain growth rate (in blocks per round)
- s: Chain growth duration (in rounds)
- f: Probability of successful round
- ϵ: Chernoff bound error
- λ: Chernoff bound duration
- X: Successful round indicator
- Y: Convergence opportunity indicator
- Z: Adversarially successful query indicator

Formulae

- The honest majority assumption: $t < (1-\delta)(n-t)$.
- The balancing inequality: $3f + 3\epsilon \leq \delta$.
- The proof-of-work inequality: $H(B) \leq T$.
- The proof-of-stake inequality: $H(s_0 \,\|\, pk \,\|\, r) \leq T_p$.

Security Definitions

Algorithm 1 The collision resistance game.

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

Definition 1 (Collision Resistant Hash Function). *A hash function* $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ *is* collision resistant *if for any PPT adversary* $\mathcal{A}$*:*

$$\Pr[\mathit{collision}_{H,\mathcal{A}}(\kappa) = 1] < \mathrm{negl}(\kappa)$$

Algorithm 2 The preimage resistance game.

function $\textsc{Preimage}_{H,\mathcal{A}}(\kappa)$
 $x \xleftarrow{\$} \{0,1\}^{2\kappa}$
 $y \leftarrow H_\kappa(x)$
 $x^* \leftarrow \mathcal{A}(y)$
 return $H_\kappa(x^*) = H_\kappa(x)$
end function

Definition 2 (Preimage Resistant Hash Function). *A hash function* $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ *is* preimage resistant *if for any PPT adversary* $\mathcal{A}$*:*

$$\Pr[\mathit{preimage}_{H,\mathcal{A}}(\kappa) = 1] < \mathrm{negl}(\kappa)$$

Algorithm 3 The second preimage resistance game.

function $\textsc{2nd-Preimage}_{H,\mathcal{A}}(\kappa)$
 $x \xleftarrow{\$} \{0,1\}^{2\kappa}$
 $x' \leftarrow \mathcal{A}(x)$
 return $H_\kappa(x) = H_\kappa(x') \wedge x \neq x'$
end function

Definition 3 (Second Preimage Resistant Hash Function). *A hash function* $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ *is* second preimage resistant *if for any PPT adversary* $\mathcal{A}$*:*

$$\Pr[\textit{2nd-preimage}_{H,\mathcal{A}}(\kappa) = 1] < \mathrm{negl}(\kappa)$$

Algorithm 4 The existential forgery game for a signature scheme (Gen, Sig, Ver).

function existential-forgery-game$_{Gen,Sig,Ver,\mathcal{A}}(\kappa)$
    $(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $M \leftarrow \emptyset$
    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

Definition 4 (Existentially Unforgeable Signature Scheme). *A signature scheme* Gen, Sig, Ver *is* existentially unforgeable *if for any PPT adversary* $\mathcal{A}$*:*

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

Algorithms

Algorithm 5 The Random Oracle

$r \leftarrow 0$
$\mathcal{T} \leftarrow \{\}$ ▷ Initiate Cache
$Q \leftarrow 0$ ▷ q for honest parties, qt for adversary
function $H_\kappa(x)$
    if $x \notin \mathcal{T}$ **then** ▷ First time being queried
        if $Q = 0$ **then** ▷ Out of Queries
            return $\perp$
        end if
        $Q \leftarrow Q - 1$
        $\mathcal{T}[x] \xleftarrow{\$} \{0,1\}^\kappa$
    end if
    return $\mathcal{T}[x]$ ▷ Return value from Cache
end function

Algorithm 6 The environment.

$r \leftarrow 0$
function $\mathcal{Z}^{n,t}_{\Pi,\mathcal{A}}(1^\kappa)$
    $\mathcal{G} \xleftarrow{\$} \{0,1\}^\kappa$ ▷ Genesis block
    for $i \leftarrow 1$ to $n - t$ **do** ▷ Boot stateful honest parties
        $P_i \leftarrow \mathsf{new}\ \ \Pi(\mathcal{G})$
    end for
    $A \leftarrow \mathsf{new}\ \ \mathcal{A}(\mathcal{G}, n, t)$ ▷ Boot stateful adversary
    $\overline{M} \leftarrow []$ ▷ 2D array of messages
    for $i \leftarrow 1$ to $n - t$ **do**
        $\overline{M}[i] \leftarrow []$ ▷ Each honest party has an array of messages
    end for
    while $r < \mathsf{poly}(\kappa)$ **do** ▷ Number of rounds
        $r \leftarrow r + 1$
        $M \leftarrow \emptyset$
        for $i \leftarrow 1$ to $n - t$ **do** ▷ Execute honest party i for round r
            $Q \leftarrow q$ ▷ Maximum number of oracle queries per honest party (Section 2)
            $M \leftarrow M \cup \{P_i.\mathsf{execute}^H(\overline{M}[i])\}$ ▷ Adversary collects all messages
        end for
        $Q \leftarrow tq$ ▷ Max number of Adversarial oracle queries
        $\overline{M} \leftarrow A.\mathsf{execute}^H(M)$ ▷ Execute rushing adversary for round r
        for $m \in M$ **do** ▷ Ensure all parties will receive message m
            for $i \leftarrow 1$ to $n - t$ **do**
                $\mathsf{assert}(m \in \overline{M}[i])$ ▷ Non-eclipsing assumption
            end for
        end for
    end while
end function

Algorithm 7 The honest party

$\mathcal{G} \leftarrow \epsilon$
function CONSTRUCTOR($\mathcal{G}'$)
    $\mathcal{G} \leftarrow \mathcal{G}'$ ▷ Select Genesis Block
    $\mathcal{C} \leftarrow [\mathcal{G}]$ ▷ Add Genesis Block to start of chain
    round $\leftarrow 1$
end function
function EXECUTE(1^κ)
    $\tilde{\mathcal{C}} \leftarrow \text{maxvalid}(\mathcal{C}, \bar{M}[i])$ ▷ Adopt Longest Chain in the network
    if $\tilde{\mathcal{C}} \neq \mathcal{C}$ **then**
        $\mathcal{C} \leftarrow \tilde{\mathcal{C}}$
        BROADCAST($\mathcal{C}$) ▷ Gossip Protocol
    end if
    $x \leftarrow$ INPUT() ▷ Take all transactions in mempool
    $B \leftarrow$ POW($x, H(\mathcal{C}[-1])$)
    if $B \neq \perp$ **then** ▷ Successful Mining
        $\mathcal{C} \leftarrow \mathcal{C} || B$ ▷ Add block to current longest chain
        BROADCAST($\mathcal{C}$) ▷ Gossip protocol
    end if
    round $\leftarrow$ round$+1$
end function
function READ
    $x \leftarrow \epsilon$ ▷ Instantiate transactions
    for $B \in \mathcal{C}$ **do**
        $x \leftarrow x || B.x$ ▷ Extract all transactions from each block in the chain
    end for
    return x
end function

Algorithm 8 Mining

function $\textsc{pow}_{H,T,q}(x, s)$
  $ctr \xleftarrow{\$} \{0,1\}^{\kappa}$ ▷ Randomly sample Nonce
  for $i \leftarrow 1$ to q **do** ▷ Number of available queries per party
    $B \leftarrow s||x||ctr$ ▷ Create block
    if $H(B) \leq T$ **then** ▷ Successful Mining
      return B
    end if
    ctr $\leftarrow$ ctr $+1$
  end for
  return $\perp$ ▷ Unsuccessful Mining
end function

Algorithm 9 The longest chain rule

function $\textsc{maxvalid}_{\mathcal{G},\delta(\cdot)}(\overline{C})$
  $C_{\mathsf{max}} \leftarrow [\mathcal{G}]$ ▷ Start with current adopted chain
  for $C \in \overline{C}$ **do** ▷ Iterate for every chain received through gossip network
    if $\mathsf{validate}_{\mathcal{G},\delta(\cdot)}(C) \wedge |C| > |C_{\mathsf{max}}|$ **then** ▷ Longest Chain Rule
      $C_{\mathsf{max}} \leftarrow C$
    end if
  end for
  return C_{max}
end function

Algorithm 10 Chain Validation

function $\text{VALIDATE}_{\mathcal{G},\delta(\cdot)}(C)$
  if $C[0] \neq \mathcal{G}$ **then** ▷ Check that first block is Genesis
    return false
  end if
  $\text{st} \leftarrow \text{st}_0$ ▷ Start at Genesis state
  $h \leftarrow H(C[0])$
  $\text{st} \leftarrow \delta^*(st, C[0].x)$
  for $B \in C[1{:}]$ **do** ▷ Iterate for every block in the chain
    $(s, x, ctr) \leftarrow B$
    if $H(B) > T \vee s \neq h$ **then** ▷ PoW check and Ancestry check
      return false
    end if
    $\text{st} \leftarrow \delta^*(st, B.x)$ ▷ Application Layer: update UTXO & validate transactions
    if $st = \perp$ **then**
      return false ▷ Invalid state transition
    end if
    $h \leftarrow H(B)$
  end for
  return true
end function

Chain Virtues

1. **Common Prefix** $(k \in \mathbb{N})$**.** $\forall$ honest parties P_1, P_2 adopting chains $\mathcal{C}_1, \mathcal{C}_2$ at any rounds $r_1 \leq r_2$ respectively, $\mathcal{C}_1[{:}-k] \preceq \mathcal{C}_2$ holds.

2. **Chain Quality** $(\mu \in [0,1], \ell \in \mathbb{N})$**.** $\forall$ honest party P with adopted chain $\mathcal{C}$, $\forall i$ any chunk $\mathcal{C}[i{:}i+\ell]$ of length $\ell > 0$ has a ratio of honest blocks μ.

3. **Chain Growth** $(\tau \in \mathbb{R}^+, s \in \mathbb{N})$**.** $\forall$ honest party P, $\forall r_1, r_2$ with adopted chain C_1 at round r_1 and adopted chain $\mathcal{C}_2$ at round $r_2 \geq r_1 + s$, it holds that $|\mathcal{C}_2| \geq |\mathcal{C}_1| + \tau s$.

Ledger Virtues

- **Safety:** For all honest parties P_1, P_2, and rounds r_1, r_2, $L_{r_1}^{P_1}$ is a prefix of $L_{r_2}^{P_2}$ or vice versa.

- **Liveness**(u): If all honest parties attempt to inject a transaction tx at rounds $r, ..., r + u$, then for all honest parties P, tx will appear in L_{r+u}^{P}.

Theorems

Lemma 5 (Patience Lemma)**.** *In typical executions, any $k \geq 2\lambda f$ blocks have been computed in at least $\frac{k}{2f}$ rounds.*