Reichmann University: Blockchains and Cryptocurrencies

Exercise 3

Assigned: January 06, 2023 Due: January 20, 2023

General Instructions

How to Submit The homework exercise has two parts:

1. Question 1 requires you to write a formal proof, and its answer should be submitted via the course Moodle by uploading a PDF file. **The submitted homework must be typed** (not hand-written and scanned).

   You will receive a 2 point bonus for the exercise if your submission was prepared in LaTeX, and contains the LaTeX macro in a footnote. You can download a latex homework template to get you started. If you don't want to install latex locally, there are online latex editors such as Overleaf that you can use.

2. Questions 2 to 4 are coding questions and should be submitted via the Inginious Server.

Pair Submission You are encouraged to solve the **theory part** of this exercise in pairs. To register your pair, write your partner's name and inginious username at the top of the page you submit. When you solve in a pair, **each member of the pair should submit the solution independently**, but you and your partner may submit identical solutions.

Note that the *Inginious* part of the assignment should not be submitted in pairs (you are welcome to discuss solutions with your classmates, but don't copy code).

Homework Questions

1. **(Cryptographic Sortition)** In this question, we consider committee-sampling algorithms that define a distribution of the form

   $$(C_0, \ldots, C_{k-1}) \leftarrow \mathsf{Sample}(k, n)$$

   where k is the size of the committee, n the total number of miners and the underlying probability space consists of the random coins flipped by the algorithm.

   We're interested in the resilience of the sampling algorithm to an adversary that can corrupt some subset $S \subset \{0, \ldots, n-1\}$ of the miners (before the sampling algorithm is executed). We assume there exists a constant q such that the corrupting adversary cannot corrupt more than a q fraction of miners — that is, $|S| \leq q \cdot n$.

   The event we're trying to avoid, which we denote $\mathbf{bad}_{k,n}$, is that $k/2$ or more of the chosen committee members are corrupted (belong to S). Note that if a miner is chosen more than once to the committee, it counts more than once.

   (a) (20 points) Suppose the committee members are sampled uniformly and independently at random (i.e., $(C_1, \ldots, C_k)$ is a product distribution and for all $i < k$ and $j < n$ it holds $\Pr[C_i = j] = 1/n$). Prove that for every constant $q < 1/2$ and every corruption strategy for the adversary, $\Pr[\mathbf{bad}_{k,n}] = 2^{-\Omega(k)}$. (Note that since q is constant, $\Pr[\mathbf{bad}_{k,n}]$ can depend on q; however, it cannot be a function of n.)

Hint: you may find the *Chernoff bound* useful. The following version is one of the simpler ones and should suffice:

Chernoff Bound If $X_1, \ldots, X_m$ are independent random variables taking values in $\{0, 1\}$. Let X denote their sum and let $\mu = E[X]$ denote the sum's expected value. Then for any $\delta \in [0, 1]$,

$$\Pr\{X \leq (1-\delta)\mu\} \leq e^{-\frac{\delta^2\mu}{2}} .$$

(b) (15 points) The result above assumes members are sampled *independently*. Prove that sampling committee members uniformly is *not* sufficient by itself. Describe an explicit randomized algorithm $\mathsf{Sample}(k, n)$ that satisfies:

1. For all $j < n$ and $i < k$:
$$\Pr[C_i = j] = \frac{1}{n}$$
(i.e., every committee member is uniformly selected) but
2. For all $q < 1/2$ and $k > 0$, there exists an adversary that corrupts $q \cdot n$ miners such that
$$\Pr[\mathbf{bad}_{k,n}] \geq q$$
(i.e., the probability that $k/2$ or more of the committee members are corrupt miners is at least q.)

Prove that your algorithm satisfies the conditions.

For bonus points, your algorithm should satisfy $\Pr[\mathbf{bad}_{k,n}] \geq 2q$ (you can assume that k is even and $q \cdot n$ is an integer).

2. (25 points) **(VRFs, Signatures and Seeds)** Solve the VRFs, Signatures and Seeds task on Inginious.

3. (25 points) **(Rewards and Incentive Compatibility)** Solve the Rewards and Incentive Compatibility task on Inginious.

4. (25 points) **(Nakamoto Consensus)** Solve the Nakamoto Consensus task on Inginious.

   To help with debugging, I have provided a testing framework that you can use offline in your favorite IDE. To use it, clone the repository: `https://vcs.ap.runi.ac.il/blockchain/hw3` (your Inginious credentials should work). Write your code in a new file named `student_code.py`.

   Note that submission is still via the Inginious server.