EE 374 Homework 1

Lucas Masterson August 7, 2025

1 Problem

Problem 1.a.

The main problem with trust is that it can be broken. For example, we entrust banks to authorize and conduct our transactions, but they can simply refuse to execute a transfer as they please, or they can act maliciously by simply stealing your funds and removing your account.

Problem 1.b.

A secure digital signature scheme doesn't rely on an external third party to verify transactions. This is instead entrusted to how good some cryptographic scheme is.

Problem 1.c.

No, Nakamoto's proof-of-work longest chain protocol trusts that a majority 51% network is honest. Otherwise, malicious actors can control the verification of transactions.

2 Problem

Problem 2.a.

Mining Bitcoin is a process where some user tries finding nonce until a block b satisfies the condition $H(b) \leq \texttt{threshold}$, where H represents some hash function (SHA-256 in the case of Bitcoin). Trials can be modeled as independent, and the first number of trials tend to follow a geometric distribution. However, as success probability decreases and the number of trials increases, the approximate time a block is found can be modeled using an exponential distribution [1]. The probability density function of an exponential distribution is as follows:

$$f(x; \lambda) = \lambda e^{-\lambda x} \tag{1}$$

where λ represents the rate parameter and some random variable x has an exponential distribution.

With Bitcoin, the rate λ is the average time to find a block, which is about once every 10 minutes. Thus, $1/\lambda = 1/600$. With this in mind, we can conclude that the probability density function is (where x is time in seconds):

$$f(x; \lambda) = \frac{1}{600} e^{-x/600} \tag{2}$$

Problem 2.b.

In exponential distributions, the variance of x is given by:

$$\text{Var}[x] = \frac{1}{\lambda^2} \tag{3}$$

Thus, the standard deviation is equal to the mean. It follows that the ratio of the standard deviation over the mean is 1.

Problem 2.c.

Let $T_1, ..., T_10$ be the inter-block mining times for 10 consecutive blocks. We can model each T_i as $\text{Exp}(\lambda)$ with mean

$$\mathbb{E}[T_i] = \frac{1}{\lambda} = 10 \text{ minutes}. \tag{4}$$

Thus, $\text{Var}(T_i) = 1/\lambda^2 = (10)^2$ minutes^2

Assuming independence (which is reasonable given the mining model), for the sum $S = \sum_{i=1}^{10} T_i$,

$$\mathbb{E}[S] = \sum_{i=1}^{10} \mathbb{E}[T_i] = 10 \cdot 10 = 100 \text{ minutes} \tag{5}$$

$$\text{Var}(S) = \sum_{i=1}^{10} \text{Var}(T_i) = 10 \cdot 10^2 = 1000 \text{ minutes}^2. \tag{6}$$

Therefore

$$SD(S) = \sqrt{1000} = 10\sqrt{10} \approx 31.6228 \text{ minutes} \approx 1897.37 \text{ seconds}. \tag{7}$$

Problem 2.d. *Skip (can't access site)*

3 Problem

Problem 3.a.

Let the total network hashrate be $H = 14.4$ EH/s $= 14.4 \times 10^{18}$ hashes/s. In 10 minutes, there are 600 seconds, so the network performs

$$N = H \cdot 600 = 14.4 \times 10^{18} \cdot 600 = 8.64 \times 10^{21} \tag{8}$$

hashes on average per 10 minute interval.

If on average one hash out of those N succeeds (i.e., yields a block in 10 minutes), the threshold T in the inequality $H(\text{block}) \leq T$ is approximate the reciprocal of N:

$$T \approx \frac{1}{N} = \frac{1}{8.64 \times 10^{21}} \approx 1.1574 \times 10^{-22}. \tag{9}$$

We can represent this as a power of two (numbering of leading zero bits):

$$\log_2 N = \log_2(8.64 \times 10^{21}) \approx 72.8715, \tag{10}$$

so

$$T \approx 2^{-72.8715} \approx 2^{-73}. \tag{11}$$

Thus, the threshold corresponds to about 73 leading bits, or roughly $73/4 \approx 18$ hex digits, i.e. about 18 leading zero hex digits.

As fpr discrepancies, the difficulty is adjusted in the network every so often, so the difficulty we assumed there could change depending on the certain time period you look at the network.

Problem 3.b. *Skip*

4 Problem

Sources: [2]

Problem 4.a.

As per the signature algorithm, the signature for the message $m = (1, 1, 1, 0)$ would be $(x_{1,1}, x_{2,1}, x_{3,1}, x_{4,0})$.

To verify the image, we assume the verifier has:

$$m = (1, 1, 1, 0)$$
$$pk = \{y_{i,0}, y_{i,1}\}$$
$$\sigma = (x_{1,1}, x_{2,1}, x_{3,1}, x_{4,0}).$$

The verifier will then iterate and check that:

$$H(x_{1,1}) = y_{1,1}$$
$$H(x_{2,1}) = y_{2,1}$$
$$H(x_{3,1}) = y_{3,1}$$
$$H(x_{4,0}) = y_{4,0}$$

So, yes, this signature verifies.

Problem 4.b.

Yes, this signature scheme is correct. For any messsage $m \in \{0, 1\}^l$, the signer releases x_{i,b_i} where $H(x_{i,b_i}) = y_{i,b_i}$ by construction of the key pair. The verifier checks the exact equality for each i. Since all values are computed honestly and the hash function is deterministic, the verifier always outputs 1. Thus, correctness holds.

Problem 4.c.

No, this signature is not secure. An adversary can break exisential unforgeability under chosen messsage attack by requesting a signature on two messages m and m' such that, between them, the message bits cover both 0 and 1 in each position. This reveals both preimages $x_{i,0}$ and $x_{i,1}$ for some (or all) i, enabling the adversary to forge signatures on arbitrary new messages. After enough queries, the adversary reconstructs the entire secret key.

Problem 4.d.

Yes, the scheme is secure if each key pair is used to sign only a single message. In the one time setting, the adversary sees only one preimage per x_{i,b_i} per bit position. To forge a signature on a different message $m \neq m'$, the adversary would need to produce a preimage $x_{i,1-b_i}$ for some i, which they cannot do unless they invert the hash function. Assuming H is preimage resistant, this is infeasible, so the scheme is secure.

References

[1] Exponential distribution - Wikipedia — en.wikipedia.org. https://en.wikipedia.org/wiki/Exponential_distribution. [Accessed 08-08-2025].

[2] Lamport signature - Wikipedia — en.wikipedia.org. https://en.wikipedia.org/wiki/Lamport_signature. [Accessed 08-08-2025].