Peer to Peer Systems and Blockchains
Academic Year 2020/2021
Final Term
Deadline 31 05 2021

Part 1 - Ethereum e Smart Contracts

Sayonara, my Mayor!

The inhabitants of Valadilène elect a mayor every 50 years. A mayor has very important duties such as deciding the decorations of Valadilène during the festivities or organizing the *Maxi-Delirium*, the monthly buffet where Valadilène citizens meet and have fun. However, a mayor has also to resolve the conflicts between rival families, especially those who started a war hundred years ago, and they do not remember why anymore. Many other tasks keep the mayor busy, time goes by and, after 50 years, the citizens need to decide whether to confirm the mayor, or kick them out from Valadilène for the bad job! “Sayonara, my Mayor!” is the typical phrase the folks tell to an unconfirmed mayor while they leave Valadilène.
It is curious how the Valadilène system to elect and confirm was born. It began as a game 3500 years ago, when Valadilène was ruled by annoying people who claimed to be sons of a few Gods and called themselves Pharaohs, or something. Everyone liked the game and stopped considering the Pharaohs, despite their complaints. Nowadays, Pharaohs have a cozy cat café in the city center. They always loved cats.
Basically, anyone can be a mayor. The career begins with the candidate working as a mayor-in-trial for a month. The main challenge of a mayor-in-trial is to organize the *Maxi-Delirium*. At the end of the month the citizens are called to confirm, or not, the mayor-in-trial. If confirmed, the mayor-in-trial becomes mayor for the next 50 years, and the citizens are called again to vote for the confirmation at the end of these 50 years.
Confirming a mayor is a period of creativity at Valadilène. Each citizen creates in secret their own **sigil**, a symbol that they will use to associate a vote to themselves in the future. It is tradition, after the mayor confirmation, to call a competition for the best looking sigil, but this is another story. Each citizen puts inside an envelope their **sigil**, a voting **doblon**, and a portion of their **soul**. The doblon has two sides: a *yay* side, which illustrates a long hat, and a *nay* side, which illustrates a boot in the act of kicking. Anyone can color only one side that represents their vote in favor, yay, or not in favor, nay, to the mayor. The soul represents what someone calls “money”, and during the voting it represents appreciation, or depreciation, to the mayor: the more soul a citizen invests, the more their vote counts.
When the voting period is over Fibonacci, Ada, and Nakamoto, the counting council members, count the votes. They sum the soul attached to the *yay* doblons and to the *nay* doblons. If there is more *yay* soul, the mayor in charge is confirmed for fifty years! If there is

more *nay* soul, the mayor in charge is kicked out from Valadilène and "Sayonara, my Mayor!". What is the purpose of the sigil, you ask? Well, the voters who "lost" can get back their soul if they prove to have a sigil identical to the one that was put inside the envelope. That's why the sigil competition is called only after the confirmation of the mayor, because the sigils have to remain secret, otherwise someone can claim the soul of other citizens!
I hope that your curiosity about Valadilène is now satisfied. Probably many things sound weird to you, a living creature, but you will understand one day, in the future…

The smart contract

A Solidity smart contract implements the Valadilène mayor confirmation system.
The constructor of the smart contract accepts three parameters: the *mayor* address, an *escrow* address, and the *quorum* i.e. the number of envelopes to send, and to open, to terminate the mayor confirmation. The smart contract collects the envelopes, opens them, and computes the outcome.
The main functions are the following:
- *compute_envelope()*: An envelope is computed hashing the byte representation of three inputs: the *sigil* (uint), the *doblon* (bool), the *soul* (uint). The byte representation of an arbitrary input can be obtained with Solidity's *abi.encode(params, ...)*, which returns **bytes**, and the hash is computed with Solidity *keccak256*, which returns **bytes32**.
- *cast_envelope()*: The smart contract collects envelopes until it reaches the quorum.
- *open_envelope()*: After that, each voter can open any envelope they previously casted to officially express their vote. A voter provides the inputs previously used to compute the envelope: the sigil, the doblon, and the soul. In this phase the soul is actually sent to the smart contract as cryptocurrency.
- *mayor_or_sayonara()*: After all envelopes have been opened, the smart contract checks whether the mayor is confirmed or not: the mayor gets confirmed if the *yay* votes are strictly greater (>) than the *nay* votes. In any case, the voters who expressed the "losing vote" (*nay* in case of confirmation, *yay* in case of rejection) get their soul back as refund. If the mayor has been confirmed, the new mayor receives, to their address, the soul associated with the *yay* votes as a contribution to the expenses for the next 50 years; otherwise, the soul associated with the *nay* votes is assigned to the "escrow account" that was declared during the creation of the contract.

Task

Given the smart contract **Mayor.sol** associated to this assignment, the student is asked to answer to the following questions:

1. Implement the *open_envelope()* and *mayor_or_sayonara()* functions. Feel free to make any change to the smart contract state attributes;
2. Evaluate the cost in gas of the functions provided by the smart contract;

   a. Provide 2 or 3 cost variations of the *mayor_or_sayonara()* function, for example by fixing the quorum and varying the number of losing voters, or varying the quorum, or any other method;
3. What are the security considerations and potential vulnerabilities related to the *mayor_or_sayonara()* function? List and explain them.
4. The *compute_envelope()* function is an helper to compute an envelope. Why is its presence an issue if the smart contract would be deployed to the Ethereum network?

Note1: the smart contract can be easily exploited casting "useless" envelopes (random bytes32), or never opening them. Consider the conditions to go from a phase to the next (i.e. the modifiers in Mayor.sol) as representative conditions.
Note2: since the vote, envelope, is secret, someone could send an envelope with 0 soul, thus reaching the quorum faster. To simplify the exercise, do not consider this as a weakness (for the reason in Note1).

The tasks of the assignment can be done on Remix (https://remix.ethereum.org), or any platform the student prefers.

"That Fibonacci guy wants to open too many envelopes all together. And why 5 and then 8?"
Ada Lovelace
"I did not lose my sigil! I did not lose my sigil, it's true! I did not lose it, I DID NOT!"
Johnny, average employee
"It took me a while to adapt to this system, I am not gonna lie, but now I am fine with my life at the café."
Ramses II

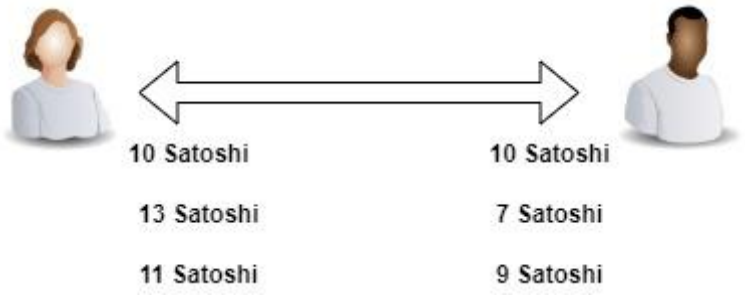
Parte 2 - BITCOIN and the Lightning network

Answer the following questions:

1) One of the main problem that the inventors of Bitcoin faced is that of "double spending". Imagine that you have 1 BTC in your wallet, you decide to buy a car, go to the car dealer, and you pay with your BTC. After having received the car, you decide to buy a sailboat, but you do not have any more money. Would you be able to buy the sailboat by somehow 'doubles spending' the 1 BTC you had earlier?
   Show how an attacker can perform a **double spending attack** and which are the countermeasures that Bitcoin defines to protect the system from this attack.

2) Alice and Bob have several trade relations and decide to open a channel of the **Bitcoin's Lightning network**. Initially each of them decides to fund the channel with

10 Satoshi. Then, they perform some transactions changing their balances in the channel, as shown in the following picture:



After the last transaction, Alice tries to cheat Bob, by publishing the second transaction, which is more favorable to her, because the state of the channel, after the second transaction, assigns 13 Satoshi to Alice and 11 Satoshi to Bob, while, after the last transaction, 11 are assigned Satoshi to Alice and 9 Satoshi to Bob. Describe how Bob can avoid that the scam is successful, highlighting the functionalities of the blockchain that are exploited to prevent the scam.