Background Assignment

N V SAI LIKHITH
nvsailikhith@gmail.com

April 2022

Background Assignment : Link to Assignment.

1 Conceptual Knowledge

1.1 What is Smart Contract

Smart contract can be thought as analogous to classes in object oriented paradigms(C++/Java). It is basically a code written in Solidity programming language that comprises of local variables, functions, modifiers etc., These are deployed on blockchain associated to a address. Each contract has a sender. Once a blockchain is deployed, it can't be manipulated.

Steps to Deploy :

1. Initially the smart contract gets compiled. This compiled version is called as bytecode.
2. This bytecode is sent to the network via a transaction.
3. They are deployed using hardhat.
   Command for deployment : `$ npx hardhat run scripts/deploy.js`
4. `deploy.js` is a javascript that contain functions which deploys the smart contract into a network (which can be specified in the cmd with `- network`
5. We can also use online IDEs like remix for deployment (without any cmd line args).

In this way we can deploy a smart contract on a network.

1.2 What is Gas

→ Gas can be said as a fee that is required to conduct a transaction (basically deploying a smart contract) on Etherum Blockchain.

→ Cost of gas is 1 gwei ($= 10^{-9}$ ethers).

→ One must build the smart contract so that it consumes as minimum gas as possible.

→ Gas is heavily consumed in bigger while loops and multiple recursions. Where as functions (mainly view) do not consume much gas.

→ View functions that simply return the local variables (like balance query) do not at all need GAS.

→ Thus it is always better to optimize the gas usage. This can be thought analogous to the running time in Algorithms.

1.3 What is hash

→ Hash is a function which give distinct outputs for each input given.

→ Output is always fixed size and changes with change in input (even a single bit change gives an unpredictable output).

→ Another main property of hash is that inverse of this function doesn't exist concretely. That is, output cannot be passed back to the input. So, the input is forever hidden

→ Evidently, it is impossible to generate back the input (brute force always exists but computationally hard), it is very useful to encode the user information like passwords etc.,

1.4 A ZK-Proof

Here are the steps how one can prove a colour-blind person that two colours are different.

→ Assume the two colours are given in two buckets. Let the verifier to pick any two completely identical handy objects (which he can confirm by touch) and apply different colours on each of them.

→ Verifier can now beleieve that there are two identical objects of diiferent colours in his hands. Prover(Me) shall ask him to select one object and remember it. I would also see the object he picked.

→ Now the verifier shuffles the objects behind me so that he only knows the hand which had his selected object.

→ After shuffling, he can ask me to pick the object which he had selected. I can easily do it as I remember the colour of object.

→ *Completeness :* If this proof is repeated multiple times, the verifier can get convinced that objects are of different colours i.e., the buckets have different colours in them.
Soundness : If the colours are same, then I wouldn't be able to pick the

correct object each time with a probability higher than 0.5.
Zero-Knowledge : The verifier never gets to know the colours of objects as we are not saying it to him at any point of time.

→ As *Completeness, Soundness, Zero-Knowledge* are satisfied this is a valid ZK-proof.

This is a cool example of ZK-proofs

2 Practical Knowledge

Solutions are present in Github Repo.

- Here are the attached screenshots of outputs after running on remix.
- Fig 1 shows the output of `HelloWorld.sol` which is storing 125.
- Fig 2 shows the winner after one voter voting to him. (only one vote is casted).
- Fig 3 shows that the contract reverted back as the time is up. Here the second vote is getting casted after 5 mins the contract was deployed.

Figure 1: Output of HelloWorld.sol

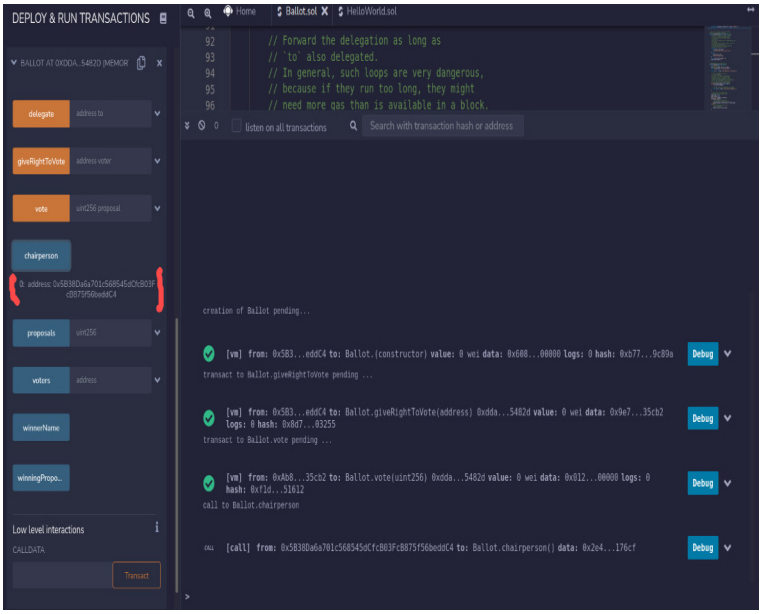

Figure 2: Output of Ballot.sol

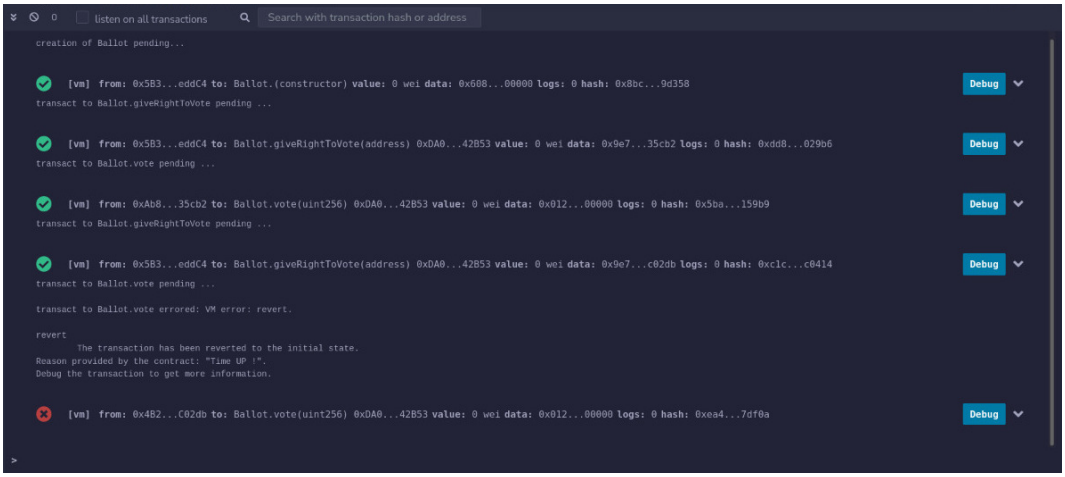

Figure 3: Not accepting vote after 5 minutes