HW1: Building your first Dapp

February 12, 2020

1 Introduction

In this homework, you will learn how to develop a smart contract on Ethereum. We will be using the Truffle framework as a development environment. Truffle is a framework for building, testing, and deploying applications on the Ethereum network. The Truffle Framework consists of three primary development frameworks for Ethereum smart contract and decentralized application (dApp) development called Truffle, Ganache, and Drizzle. However, for this homework, we only need Truffle and Ganache. To get started with developing a smart contract, you may need to get used to Solidity, an object-oriented programming language for writing Ethereum's smart contracts. I'd recommend reading this tutorial `https://www.trufflesuite.com/tutorials/pet-shop` (you don't need to get to the part of creating a user interface)

The goal of this homework is to develop a smart contract for open auction. The general idea of the auction contract is that everyone can send their bids during a bidding period. The bids already include sending money / ether in order to bind the bidders to their bid. If the highest bid is raised, the previously highest bidder gets their money back. After the end of the bidding period, the contract has to be called manually for the beneficiary to receive their money - contracts cannot activate themselves.

We have created a skeleton code for the smart contract where some functions were defined. You will have to complete those functions and submit your final code.

2 Setting up the workspace

We strongly recommend doing this task as soon as possible. There could be some errors when setting up the workspace depending on your current system's softwares and libraries.

Requirements

1. NodeJS v8.9.4 or later
2. Windows, Linux or Mac OS X

Install Truffle Truffle is made for building dApps using the Ethereum Virtual Machine (EVM) by providing a development environment, testing framework, and asset pipeline. To install, run the following command:

```
npm install -g truffle
```

On a Unix-based system, you may need to add "sudo" before the command. You can verify the installation by running "truffle version" on a terminal, you should see an output similar to the following:

```
Truffle v5.1.12 (core: 5.1.12)
Solidity v0.5.16 (solc-js)
Node v8.10.0
Web3.js v1.2.1
```

Install Ganache Ganache is a personal blockchain that allows developers to create smart contracts, dApps, and test software that is available as a desktop application and command-line tool for Windows, Mac, and Linux. To install, follow the instructions on `https://www.trufflesuite.com/docs/ganache/quickstart`. Note: please choose version 2.1.2, do not choose any beta versions.

3 Writing the smart contract

The source code is provided in **hw1-source.zip**. The directory containing the file "truffle-config.js" will be referred as the *root directory.* First, make sure that you can compile the source code. Open a terminal in the root directory and run the following command:

```
truffle compile
```

You should see a message saying that the compilation was successful.

Now, you need to complete the following functions in "contracts/Auction.sol":

```
function bid() public payable
function withdraw() public returns (bool)
function auctionEnd() public
```

You can find the guidelines to complete those functions in the code's comment.

Note that a naive implementation of the "withdraw" and "auctionEnd" functions would be subject to the **reentrancy attack** as discussed in class. Your implementation must avoid this attack.

4 Deploy the contract on Ganache

Start the Ganache software and run the following command in the root directory:

```
truffle migrate
```

By default, Ganache provides 10 Ethereum accounts, each has 100 ETH, to interact with the smart contract. The first account is the one deploying the contract and becomes the beneficiary (see the constructor in the source code).

You can interact with your contract using Truffle console. From the root directory, run:

```
truffle console
```

You can make calls or create transactions to the smart contract with some simple Javascript commands. You can find the instructions here `https://www.trufflesuite.com/docs/truffle/getting-started/interacting-with-your-contracts`

You will need to report the amount of gas or transaction fee for the following tasks:

1. Deploying the contract.
2. Triggering each of the implemented functions.

5 Deliverable

You need to submit a zip file **FirstName_LastName_HW1.zip** that contains the following:

1. **hw1-source.zip**: the completed source code
2. Your report in markdown or pdf.

Your report must include the following material:

- Your name
- Brief explanation for each function you wrote and how it works. It would be great to show some experiments on sending/withdrawing bids to show that your contract works correctly.
- The amount of gas or transaction fee needed to deploy the contract and trigger each of the implemented functions. Attach some **screenshots** showing how you obtained those numbers.
- A screenshot from *Truffle console* showing that the beneficiary account has received the money after "auctionEnd" is triggered. You can show the change in the beneficiary account balance before and after the function is triggered. The change must reflect the actual amount of the highest bid minus the transaction fee (show calculation).