Exercise 4

Smart Contracts

Program Analysis for System Security and Reliability 2018
ETH Zurich

March 24, 2018

Problem 1. In this exercise, you will learn about the challenges of implementing a decentralized bank. We encourage you to code your contract in https://remix.ethereum.org, which enables you to compile and run your code.

Task 1 Complete the following contract that enables users to deposit ether (the Ethereum's currency) and withdraw all their ether.

```
contract SimpleBank {

  TODO: Complete

function deposit(uint amount) payable {
  TODO: Complete
}

function withdraw() {
  msg.sender.call.value(TODO: Complete)
  TODO: Complete
}
```

Task 2 Your contract has been uploaded to the blockchain, and it became very popular. One day, your loyal user Alice reports that despite calling `withdraw`, she does not receive her ether. When you read the blockchain, you observe that Bob was able to withdraw all the users' ether that was stored in the contract. Describe a scenario that could lead to this situation.

Task 3 Fix your contract to prevent this situation.

Task 4 After fixing your contract and deploying it on the blockchain, Alice still complains that despite calling `withdraw`, she does not receive her ether. This time when you read the contract, you see that Alice's ether is still in the contract, however your local data structure reports that her balance is zero. Describe a scenario that could lead to this situation.

Task 5 Fix your contract to prevent this situation.

Task 6 After the two unfortunate incidents, your contract has lost its popularity. To regain it, you post a puzzle online and write a contract with the function `submitSolution` that checks a solution and pays the first solver 10 ether. However, since you are low on ether, you do not want to pay 10 ether. Unfortunately, you do not know the solution to the puzzle you solved (you can only verify that a solution is correct). Describe how you can avoid paying the first solver 10 ether, without making it look suspicious.