The Korean Institute of Information Scientists and Engineers

SIGPL Summer School 2018

Characteristics of Blockchain to consider when writing a Smart Contract

Jae-Yun Kim (Ben)

ben@decipher.ac

jaeykim@altair.snu.ac.kr

Decipher

Agenda

Decipher

Introduction

About Speaker

Jae-Yun Kim (Ben)

Career

- Majoring in Electrical and Computer Engineering at Seoul National University
- Ph.D. candidate in virtual machine & optimization lab at Seoul National University
- Founder and leader of Decipher, Blockchain Research Group at Seoul National University

Work In Progress

- Research in JavaScript app migration
- Edge Computing based on Blockchain
- Git as a blockchain governance protocol

Decipher

Agenda

Introduction to Blockchain

Introduction to Smart Contract

Characteristics to consider

Introduction to Blockchain

1) Distributed Ledger

Definition

- A consensus of replicated, shared, and synchronized digital data over the peer-to-peer(P2P) system.

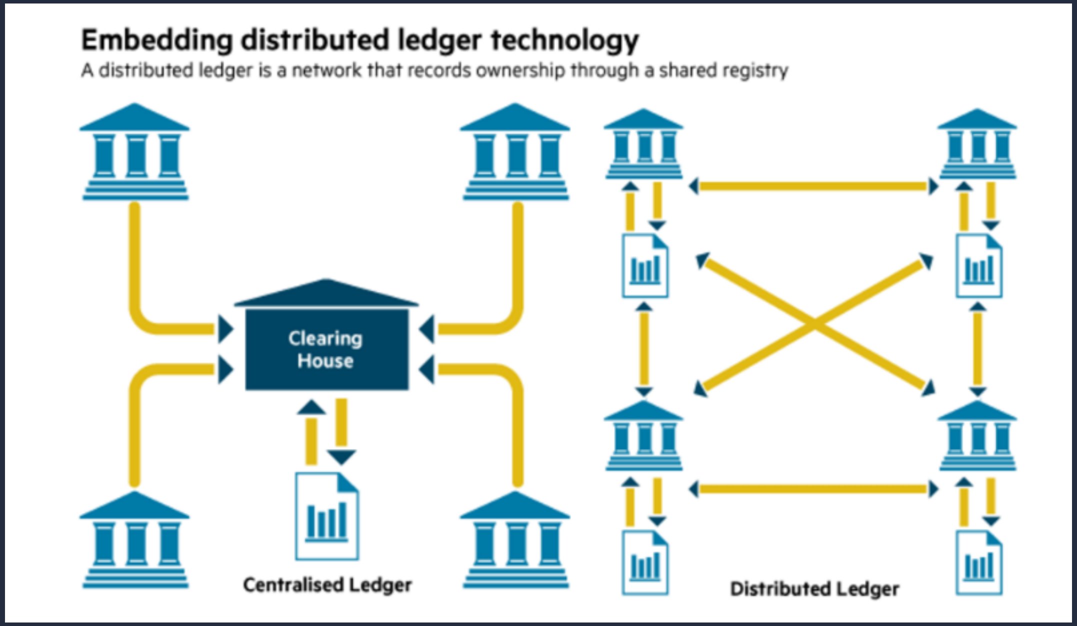

Embedding distributed ledger technology
A distributed ledger is a network that records ownership through a shared registry
Clearing House
Centralised Ledger
Distributed Ledger

Figure 1. Difference between Centralized and Distributed Ledger systems.

Introduction to Blockchain

1) Distributed Ledger

Validation

- Validity of a ledger is guaranteed by all nodes participating in the P2P network.

Ledgers in Blockchain

- Blockchain uses transactions and program states as ledgers.
- All the participants share the transactions and program states by replication and propagation.
- Synchronization requires special techniques.

Introduction to Blockchain

2) Consensus Algorithm

Contradictory Transactions

- Some transactions are incompatible.
  - e.g. Alice generates two value transferring transactions exceeding her balance.
- A system usually takes the prior transaction among the contradictory transactions.
- However, the order of transactions varies from node to node.

Consensus Algorithm

- Because there's no central administrator, the distributed system uses consensus algorithm to determine transaction order.
- Consensus Algorithm determines which node will determine the order.

Introduction to Blockchain

3) Nakamoto Consensus

Satoshi Nakamoto's solution - Bitcoin

- Satoshi Nakamoto introduces a block to bundle transactions which are generated during a certain period as a synchronization unit to replace transaction order with block order.
- Nakamoto uses a cost function which is Proof-of-Work algorithm to determine which node will create a current block.
- The order is represented by the previous hash value of the block, which makes a chain.

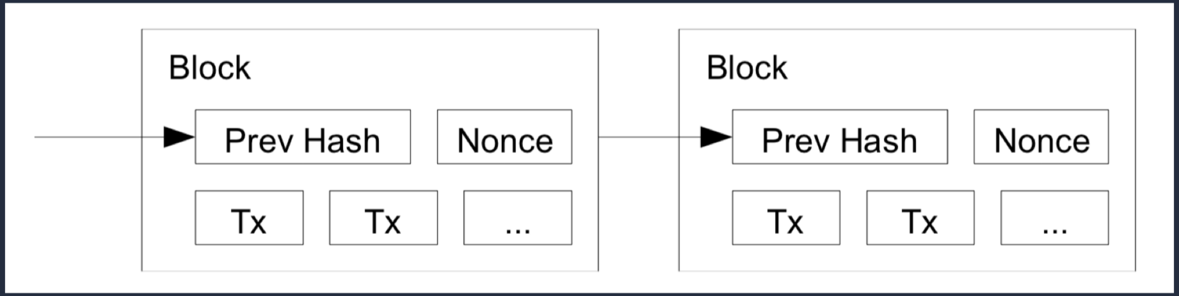


Figure 2. Proof-of-Work of Bitcoin

Introduction to Blockchain

3) Nakamoto Consensus

Longest Chain Rule

- It is possible that more than one blocks are generated simultaneously to make race condition, which is called “fork”.
- Satoshi takes “Longest chain rule” into the protocol to solve the race condition.

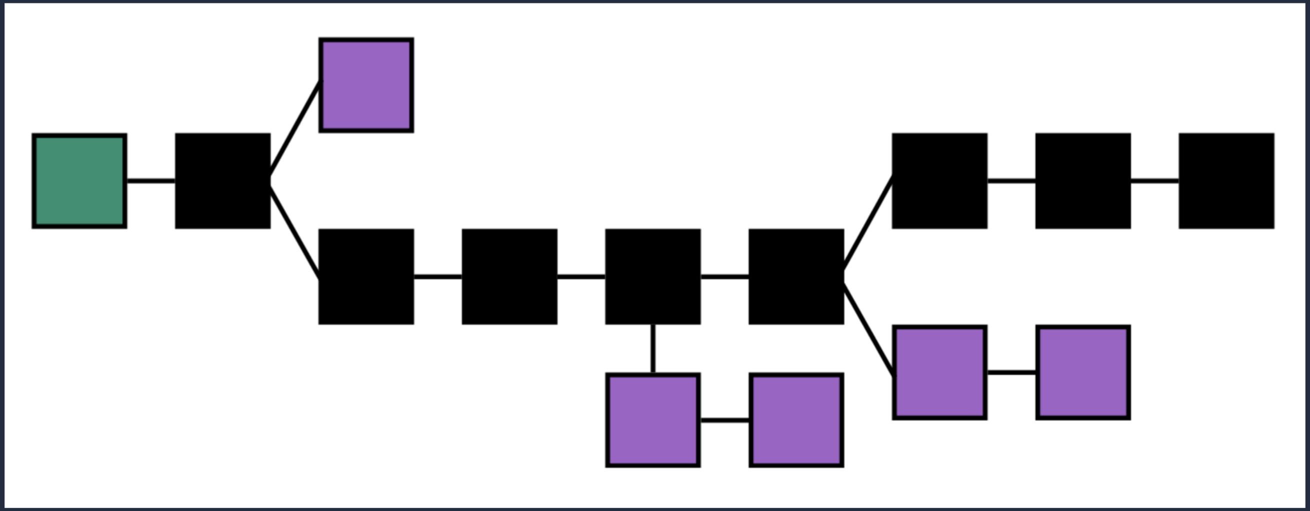


Figure 3. Longest chain rule

Introduction to Blockchain

4) Synchronization

Synchronization is achieved with blocks

- Contradictory transactions cannot be contained to one blockchain to guarantee the integrity of the transactions.
- A the generated block and blockchain are propagated to the blockchain network to be synchronized.
- Synchronization is achieved by consensus algorithm.

Importance of miners(mining nodes)

- Because Nakamoto consensus heavily relies on miners which conduct PoW, the execution of smart contract also heavily depends on miners.

Agenda

- Introduction to Blockchain
- **Introduction to Smart Contract**
- Characteristics to consider

Introduction to Smart Contract

1) Smart Contract

Definition

- A program of which source code and program state are replicated and shared by all nodes.
- People can see the source code and the execution of the program, which provides transparency.
- The transparency allows a program to be exploited as a contract.

Ledgers(data) of Smart Contract

- Source code
- Program state

Introduction to Smart Contract

2) Virtual Machine

Smart Contract is executed on the Virtual Machine

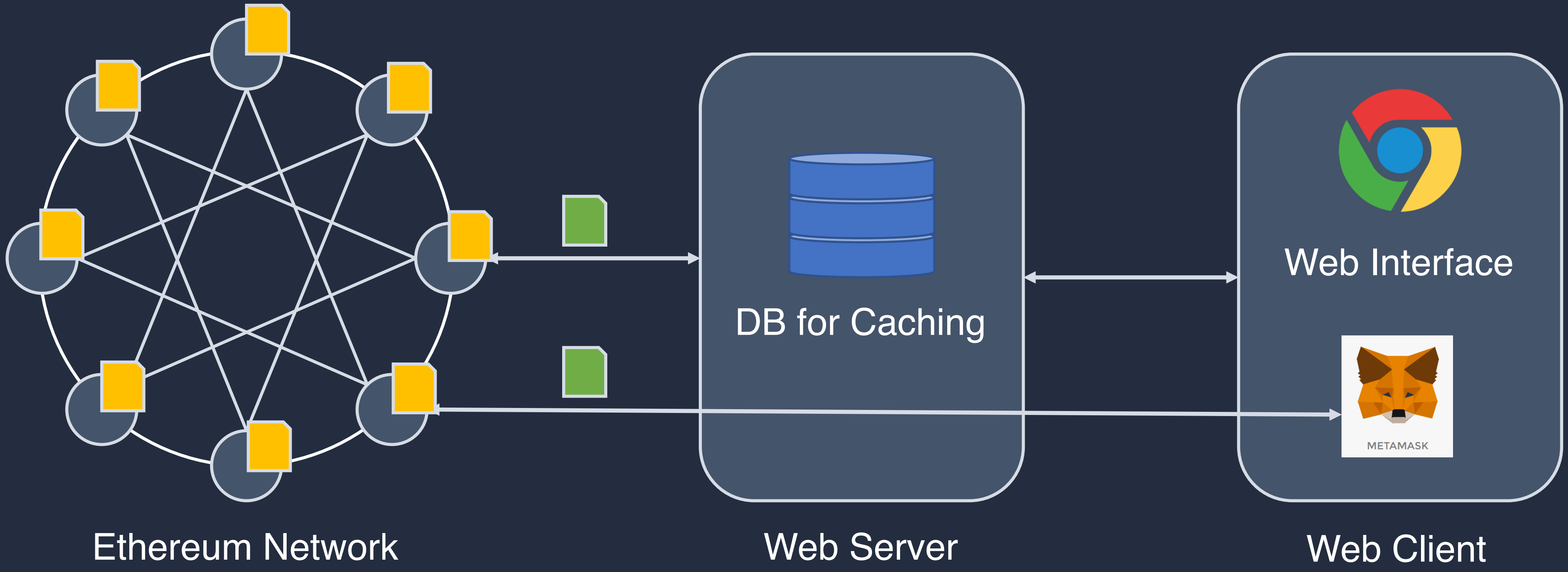
- To guarantee termination of the program.
- The transition of program state should be identical regardless of the execution environment.

Deployment of a Smart Contract

- User writes a source code with Solidity.
- The source code is compiled to a bytecode file like Java.
- The user generates a deploy transaction with the bytecode to get a contract address.

Introduction to Smart Contract

3) A Structure of Decentralized Application(DApp)



🟨 : Deployed Smart Contract 🟩 : Transaction

Figure 4. A structure of Dapp

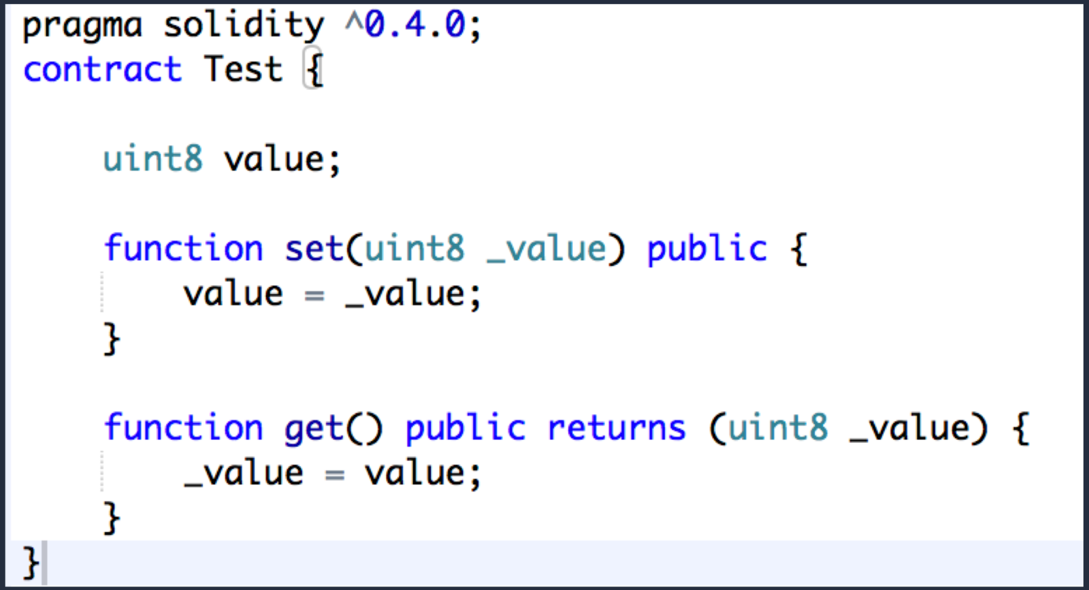
Decipher

Introduction to Smart Contract

4) Execution of Smart Contract

Function-driven(?) programming

- Execution of Smart Contract is not procedural.
- A transaction triggers a function call.
- If a user wants to call a function, he/she has to know the interface of the smart contract.
- A transaction has to contain an address of smart contract, function id, and function parameters.



```
pragma solidity ^0.4.0;
contract Test {

    uint8 value;

    function set(uint8 _value) public {
        value = _value;
    }

    function get() public returns (uint8 _value) {
        _value = value;
    }
}
```

Figure 5. Example Smart Contract

Introduction to Smart Contract

4) Execution of Smart Contract

Two kinds of functions

- A function inducing state transition requires cryptocurrencies(ETH), which is external function call. The transaction cannot get the return value immediately, because the function is not executed until the transaction is included in a block.
- On the other hand, internal function call which is triggered by other function can get the immediate return value.
- The result of the state transition should be represented by ”event” keyword, which logs a result to the corresponding block.
- View function does not change the program state. Therefore, it does not require a transaction, which is free and does not require confirmation delay.

Agenda

- Introduction to Blockchain
- Introduction to Smart Contract
- **Characteristics to consider**

Characteristics to consider

1) Time concept of Blockchain

Time in blockchain is broken into blocks

- Blocks are generated pseudo-randomly.
- Miner generates a block including it's unix timestamp.
- Blockchain protocol adjusts average block interval to be converged into ideal time interval.
  - 10min for Bitcoin, 15s for Ethereum
- Timestamps of previous blocks are used to calculate proper difficulty of PoW to adjust block interval.

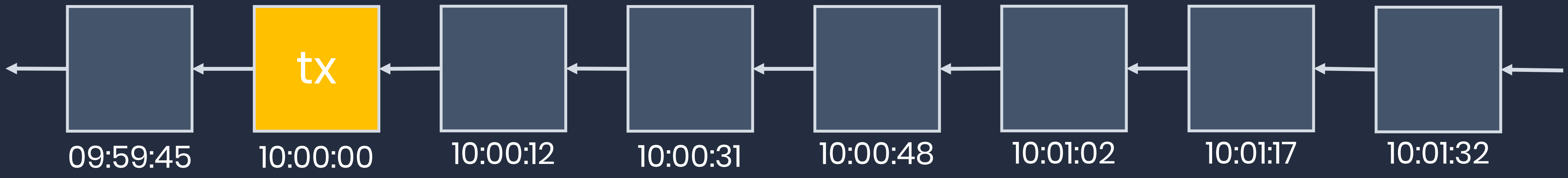


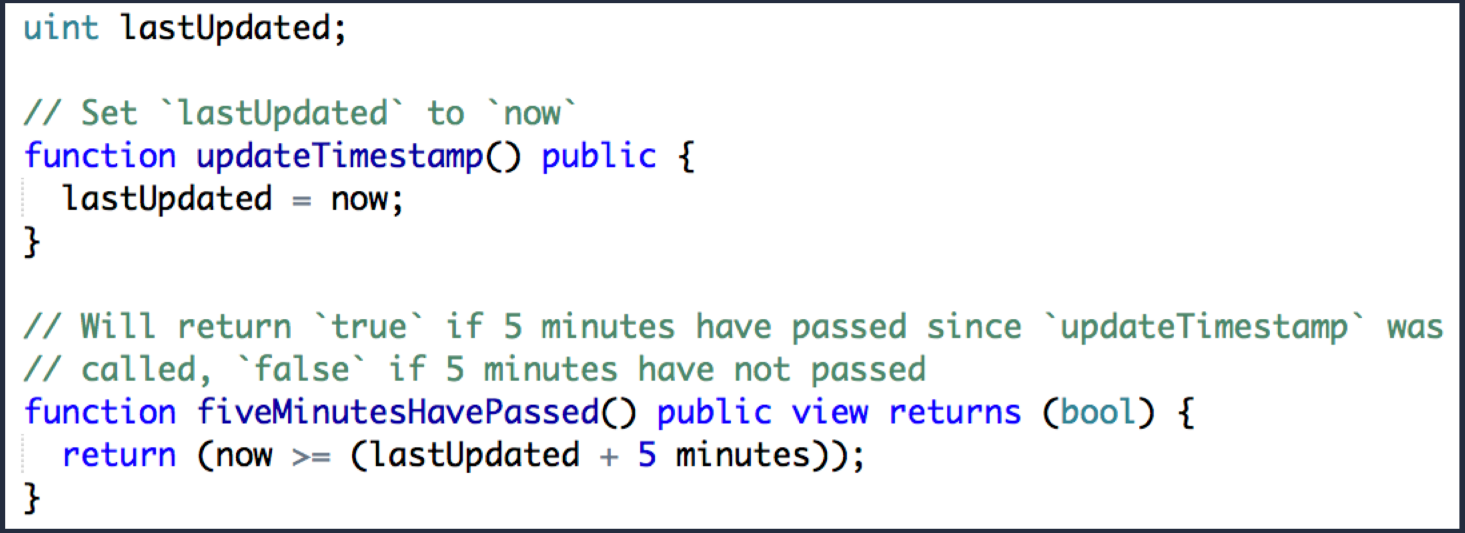
Figure 7. Timestamp of a blockchain

Characteristics to consider

1) Time concept of Blockchain

Unix Timestamp of the block

- “now” keyword returns the current unix timestamp of the latest block.
  - The number of seconds that have passed since January 1st 1970.
- Calculate time with time keywords.
  - “seconds”, “minutes”, “hours”, “days”, “weeks”, “years”



```
uint lastUpdated;

// Set `lastUpdated` to `now`
function updateTimestamp() public {
  lastUpdated = now;
}

// Will return `true` if 5 minutes have passed since `updateTimestamp` was
// called, `false` if 5 minutes have not passed
function fiveMinutesHavePassed() public view returns (bool) {
  return (now >= (lastUpdated + 5 minutes));
}
```

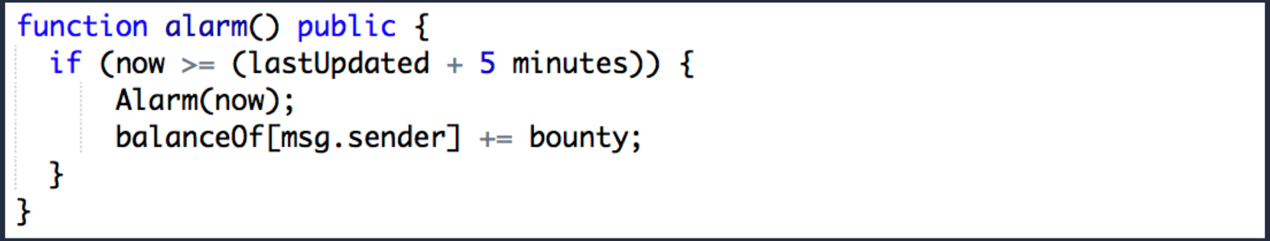
Figure 8. Example of time concept of Solidity

Characteristics to consider

2) Blockchain is not a daemon

How to implement daemon on Smart Contract

- Because Smart Contract is function-driven program, blockchain does not execute a logic without a transaction.
- If someone wants to confirm the passage of time, he/she has to execute a transaction or call a view function.
- So the external server has to generate a transaction periodically.
- If you run an external server, Smart Contract is useless.
- Setting a bounty incentivizes someone to generate the transaction.



```
function alarm() public {
  if (now >= (lastUpdated + 5 minutes)) {
      Alarm(now);
      balanceOf[msg.sender] += bounty;
  }
}
```

Figure 9. Set a bounty to incentivize people to call the function

Characteristics to consider

3) Reorganization

A block could be reverted

- Sometimes, reorganization occurs due to the Longest chain rule.
- If reorganization occurs, some nodes discard their blocks and receive longest blockchain.
- Therefore, a transaction in block does not fully guarantee validity.

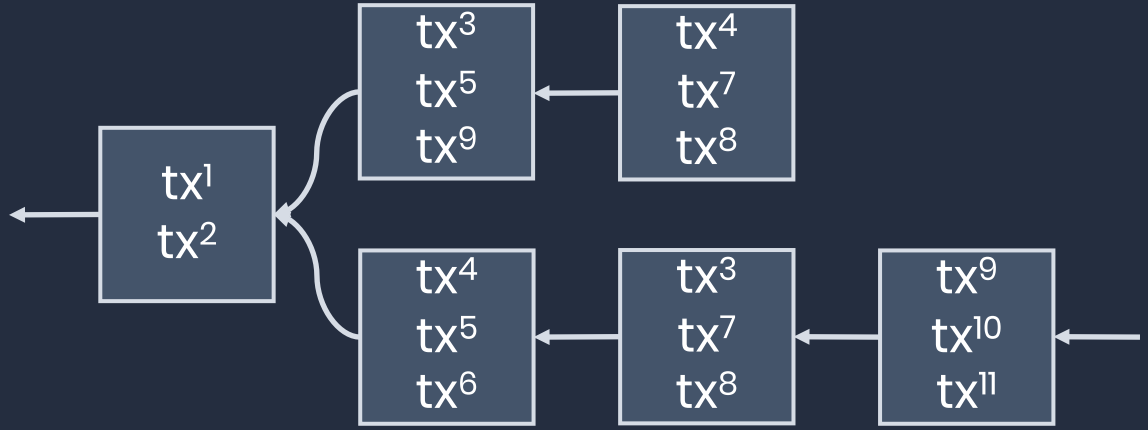


Figure 10. Reorganization

Characteristics to consider

4) Settlement Time

A certain time is required

- Transaction is not executed immediately.
- Transaction is executed when it is contained to a block.
- To avoid reversion of transaction by reorganization, we have to wait until confirmation confidence decreases to a certain level.

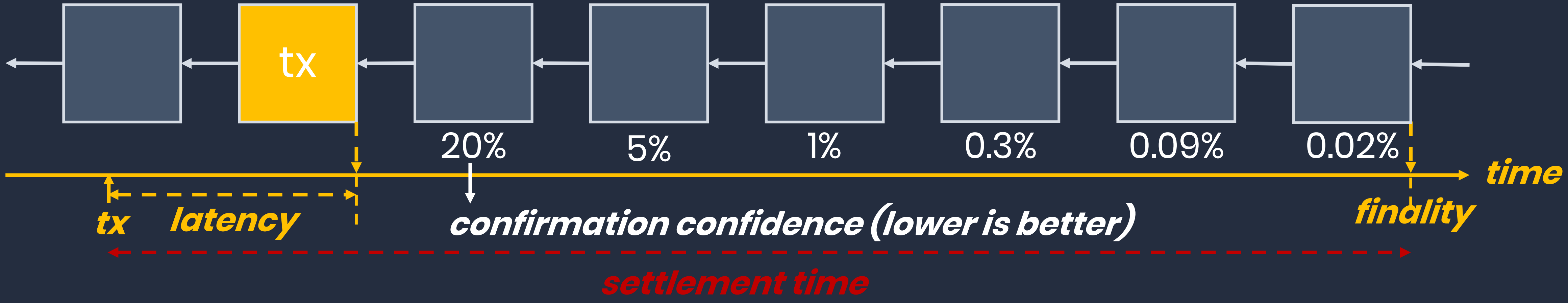


Figure 11. Settlement Time

Thank You

Q & A

Decipher