Decentralized Metering and Billing of energy on Ethereum with respect to scalability and security

Aristotle University of Thessaloniki

Honda R&D Europe

Georgios Konstantopoulos

Contents

1 Abstract

Our design includes

2 Introduction

2.1 Motivation

Blockchain technology blabla

2.2 Problem Statement

How an entity can manage the energy consumed by a complex system of energy meters. The system should be able to bill and perform accounting on the metering data, based on an accounting model The system must be transparent, distributed, decentralized, easy-to-use and secure. Anyone in the network should be able to verify the validity transactions. It also needs to be scalable at reasonable cost.

2.3 Scope

The thesis is limited on blabla, scalability is explored blabla,

2.4 Outline

Chapter X describes Y

3 Blockchain Basics

3.1 Introduction

A blockchain is a database that can be shared by non-trusting individuals without having a central party that maintains the state of the database. Namely, it is a list of *blocks* that grows with time. Each block contains metadata in the form of *blockheaders* and a group of transactions. A block is chained to its previous one by referencing the previous block's hash. As more blocks get added to the chain, previous blocks and their contents are considered to be more secure.

3.2 Bitcoin

In 2009 Satoshi Nakamoto first publishes the Bitcoin whitepaper. There, Nakamoto describes "a purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution." In the beginning, Bitcoin was primarily used for fast and low-cost financial transactions. It was soon realized that its uses could be extended to more than just transfering value from A to B. The concept of colored coins, [1] was introduced, where users were able to embed extra data on a bitcoin, effectively painting a coin with that could represent ownership over a title, digital tokens etc.

3.3 Ethereum

In 2015, Vitalik Buterin authored the Ethereum Whitepaper [3] which was an alternative cryptocurrency to Bitcoin that enabled the creation of *Smart Contracts*. Smart Contracts as a term was first introduced by Nick Szabo in 1996 as a model for verified trustless computation. The Ethereum Network acts as a world computer and smart contracts are code that gets executed trustlessly on every node that is part of the network.

3.4 Blockchain Fundamentals

Before getting into the specifics of blockchains and Ethereum, the next section will be used to explain fundamental terms on cryptograpy and blockchain.

3.4.1 Public Key Cryptography

Also refered to as Assymetric Cryptography, it is a system that uses a pair of keys to encrypt and decrypt data. The two keys are usually called **public** and **private**, due to the private key being known only to its owner while the public key is known to the public. The main advantage of Public Key Cryptography is the lack of need for a secure channel for the initial exchange of keys between any communicating parties.

The security Public Key Cryptography is based on cryptographic algorithms which are not solvable efficiently due to certain mathematical problems, such as the factorization of large integer numbers for RSA or the discrete logarithm problem for ECDSA, being hard.

When a person encrypts a message with one key, its pair can be used to decrypt the same message. If a message gets encrypted with the private key of the sender, any receiver can verify that the message was indeed sent by the sender as they are the only possible owners of the private key used to enrypt the message. This achieves authentication, and the process is often referred to as *signing* of a message.

3.4.2 Cryptographic Hash Functions

A hash function is any function that is used to map arbitrary size data to fixed size. The result of a hash function is often called the *hash* of its input. Cryptographic hash functions are hash functions that satisfy properties which make them useful for cryptograpy

More specifically, a secure cryptographic hash function should satisfy the following properties ($H(x)$ refers to the hash of x):

1. **Collision Resistance**: It should be computationally infeasible to find x and y such that $H(x) = H(y)$.

2. **Pre-Image Resistance**: Given $H(x)$ it should be computationally infeasible to find x.

3. **Second Pre-Image Resistance**: Given $H(x)$ it should be computationally infeasible to find x' so that $H(x') = H(x)$.

Bitcoin uses the SHA-256 cryptographic hash function, while Ethereum uses KECCAK-256. Both functions' outputs are 256 bits long which is considered secure given the document's writing date standards.

3.4.3 Transaction

A transaction is a data structure which at its most general representation should include the following:

1. Transaction Hash
2. From
3. To
4. Amount
5. Extra Data

3.4.4 Block

A block is a data structure which is comprised of headers and transactions. A block in Ethereum which contains 2 transactions can be seen below:

3.4.5 Blockchain

Each block references a previous block by adding its hash to its header. A block needs to have a valid previous block hash in order to be valid itself. This essentially creates a chain of blocks each one linked to its previous one all the way to the genesis block. In Bitcoin new blocks get created every approximately every 10 minutes, while in Ethereum every 12.5 seconds. The process of mining and how consensus is achieved is considered outside the scope of this Master Thesis.

3.4.6 Blockchain Types

Data on blockchains are public and readable by anyone. This is one of the main benefits of using a blockchain, transparency. However this does not apply to all business use-cases and Enterprises are looking at using their own permissioned (or private) blockchain for their internal business processes.

Private blockchain implementations are JP Morgan's Quorum, IBM Hyperledger or R3 Enterprise. As described in [2], explain differences advantages disadvantages of private vs public.

3.4.7 Ethereum Virtual Machine

The Ethereum Virtual Machine (EVM) is the runtime environment for smart contracts on Ethereum.

3.4.8 Transactions as State Transitions

3.4.9 Gas

3.5 Programming in Ethereum

3.5.1 Programming Languages

Many ones, most popular being Solidity. Contract oriented, compiles to bytecode, explain ABI

3.5.2 Tooling

Docker

Docker is used for isolated processes that interoperate with each other, same each time, no it works on my machine, launch network with docker-compose.

Geth - Parity - Ganache

Blockchian clients, Geth PoW with custom genesis, Parity can run PoA, Ganache for testnet locally. Compare Parity Genesis vs Geth Genesis blocks

Truffle

Truffle development framework.

4 Blockchain Scalability

4.1 Bottlenecks in Scalability

Explain proof of work and how it limits transactions per second. A variety of Scalability Solutions have been thought of such as Plasma, blabla, Ethermint? We argue that there are two levels of scalability. Scalability on Contract and on Network Level. While the solutions described above are aiming at solving the Network level scalability, it is argued that scalability cna be also be improved by proper contract design, i.e better algorithm

4.2 Network Level Scalability

Long term solutions include. Due to the use case we will focus on shorter term solutions are shardign and casper pos Shorter term solutions include payment channels, sidechains and alternative consensus algorithms. Due to the corporate setup, we choose to use proof of authority networks and further analyze them along with a minimal implementation of payment channels

4.3 Contract Level Scalability

4.3.1 Gas Costs

4.3.2 Optimizing for gas

4.3.3 Encoding

5 Smart Contract Security

Smart contracts immutable, lack of tooling, and developer mistakes

5.1 Past Vulnerabilities

Past vulnerabilities had lots of money locked, business logic of application broken. Brief summary of some vulns. Refer to relevant literature for more in depth. Online platforms for training and exploiting have been developed.

5.2 Security of deployed Smart Contracts

Talk about the literature which has already mapped through the deployed contracts. Which tools did they use? Oyente, Mythril etc

5.3 Best Practices

Contract oriented, keep it simple, no tx.origin, ktl as described in literature

5.4 Access Control

There is NO private data, just functions that can be called by certain individuals A proper access control model needs to be implemented so that only authorized users can access certain functions.

6 Blockchain and the Energy Market

Price of energy, consumer does not know always what they pay, or what they gain from their renewables

List relevant projects in energy sector

6.1 Advantages of Blockchain

Transparency, full history of meter readings, price calculation, billing of inhouse energy departments. This can be extended for EV car payment microtransactions and so on.

6.2 Our Use-case

Describe meters, billing and so on

7 Design and Implementation

7.1 Business Logic

Explain company structure

7.2 Smart Contracts

Explain the Smart Contracts suite

```
pragma solidity ^0.4.16;

contract TestContract {

  string private myString = "foo";
  uint private lastUpdated = now;

  function getString() view external returns (string, uint) {
    return (myString, lastUpdated);
  }

  function setString (string _string) public {
    myString = _string;
    lastUpdated = block.timestamp;
  }
}
```

7.3 Monitoring Server

Explain monitoring server

7.3.1 REST API

Explain rest api usage

7.3.2 Python Client

Explain python implementation of rest api

7.3.3 web3.py interaction

Explain how web3.py interacts with monitoring server and sends data to Smart Contracts

8 Conclusion

8.1 Results

8.2 Future Work

Final remarks include.

Bibliography

[1] Colored coins, 2013.

[2] Vitalik Buterin. On public and private blockchains. `https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains`, 2015.

[3] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 2014.