Tenancy over Distributed Workflows using Blockchain Technology exemplified by Network Slicing

Master-Arbeit

Jordi Bisbal Ansaldo
KOM-M-number

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Fachbereich Elektrotechnik
und Informationstechnik
Fachbereich Informatik (Zweitmitglied)

Fachgebiet Multimedia Kommunikation
Prof. Dr.-Ing. Ralf Steinmetz

Tenancy over Distributed Workflows using Blockchain Technology exemplified by Network Slicing
Master-Arbeit
KOM-M-number

Eingereicht von Jordi Bisbal Ansaldo
Tag der Einreichung: 30.05.2018

Gutachter: Prof. Dr.-Ing. Ralf Steinmetz
Betreuer: Dr.-Ing. Amr Rizk und Prof. Paul Müller

Technische Universität Darmstadt
Fachbereich Elektrotechnik und Informationstechnik
Fachbereich Informatik (Zweitmitglied)

Fachgebiet Multimedia Kommunikation (KOM)
Prof. Dr.-Ing. Ralf Steinmetz

Erklärung zur Abschlussarbeit gemäß § 23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Jordi Bisbal Ansaldo, die vorliegende Master-Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§38 Abs.2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Master-Arbeit stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung überein.

Darmstadt, den 30.05.2018

Jordi Bisbal Ansaldo

Contents

Abstract

The abstract goes here... (BITCOIN influence -> Blockchain). Captivate readers attention.

1 Introduction

Blockchain (BC) has been considered as one of the most promising disruptive technologies during the last years. Many market-leading companies, experts and global innovators have referred it as the "Next Generation of the Internet" [Cla17], succeeding the World Wide Web era. Evaluating its potential benefits, different banks and major enterprises, such as UBS, Microsoft or IBM, have already accomplished important investments in such innovative technologies.

The revolution started in 2008, with a whitepaper publication by Satoshi Nakamoto [Nak08], who introduces a new digital payment protocol called Bitcoin. Satoshi Nakamoto is just a name used by an unknown person or group of people to first reference the implementation. Nowadays, Bitcoin's creator still remains a mystery.

In 2009, a deployed software version based on the paper was launched. Bitcoin uses an alternative virtual currency, to make trusted transactions between different peers. This system relies on a kind of distributed database allocated on the Internet, the blockchain. Blockchain uses a peer-to-peer architecture model combined with secure algorithms, such as public key cryptography, which intentionally removes the presence of middle-parties or intermediaries. Thus, in the case of Bitcoin, it eliminates the agent responsible for transactions: central banks.

At the beginning, the blockchain architecture was restricted to only one application: online payments. However, after observing its advantages and possible use cases, an improvement of Bitcoin emerged: Ethereum [1]. By contrast, Ethereum extends the power of decentralized transactions with a Turing-complete contract system. A Turing-complete system can perform any computation, with just writing a few lines of code, in order to create Ethereum scripts: smart contracts. A smart contract can be generated with non-restrictive and user-friendly programming languages, allowing developers to easily learn and benefit from them. Therefore, it brings to the user the opportunity to develop their own applications. Smart contracts are used to implement decentralized apps, which unlike normal web apps, they may not be allocated in a central server. In other words, they use blockchain technology to retrieve and store data instead of a database.

1.1 Motivation

Nowadays, blockchain is becoming a trending topic in the business world. Thousands of articles, research papers and books, such as: How the technology behind bitcoin is changing money, business and the world [TT16] or Blockchain: Blue print for a new economy [Swa15], are catching the public eye. Nevertheless, as it is an emerging technology, a necessity to look towards new horizons exists. Through the use of the above mentioned Ethereum smart contracts new possibilities to approach existing problems are opened up. Hence, blockchain technology can be used in many applications beyond currency.

Many scenarios are currently investigated from a blockchain perspective, e.g: candidate's voting or asset tracking [AM16]. In the former, voters send signed and encrypted ballots to the blockchain contract, who immediately verifies them. Simultaneously, it also preserves confidentiality, since the ballot can only be emitted from its owner. In the latter, each physical asset could be encoded in the blockchain enabling a fast and transparent tracking. For example, Everledger [2], a startup company from London, tracks diamonds storing each diamond's digital identity on the BC. Thus, diamond theft could be effectively prevented.

After investigating the implementation of blockchain in real scenarios, one top-level application is clear: it can be used as a software connector. Blockchain contributes to face security, storage's prob-

[1] https://www.ethereum.org/

[2] https://www.everledger.ioafa

lem, communication and coordination between users. In this paper, we will focus on enterprises or organizations, suffering from such problem, referenced from now on as supply chain challenges.

1.2 Problem Statement and Contribution

Supply chain management is the process of linking organizations through information flows, in order to achieve a competitive strength or advantage, which will maximise customer value. Supply chain activities go from the design or development of a product, up to its return on investment (ROI). Thus, a good coordination during these activities is extremely needed.

Nowadays in the Big Data era, enterprises must handle a huge amount of information. This leads companies to suffer from considerable issues, such as scalability, data's security or communication. One could think that currently most of these companies rely on third parties, which help them on the mentioned problems. But what if this process could be efficiently accelerated in a secure and decentralized manner? Here, is where Blockchain can play a crucial role.

During the paper, the focus will be on IT companies facing this dilemma. The scenario will include in one side different customers, and in the other organizations acting as providers. For example, eBay, one of the biggest multinational e-commerce corporations, acts as an intermediate for the product's purchase-sale. Thus, eBay is responsible for managing all this data. However, can a user/company always safely trust third-parties? Why not using BC as the main responsible for handling such complex tasks? This will result in a direct customer-provider relation, avoiding the presence of any intermediaries.

For this scenario, a current application example that consists of the embedding of virtual networks (network virtualization) between different Infrastructure Providers (InPs), will be further investigated [DRP15]. This process can also be called network slicing. In this example, Service Providers (SPs) want to embed virtual nodes among different InPs in order to provide wide-area network services. Nevertheless, Infrastructure Providers are not willing to publicly disclose its internal network topology, along with its resources availability and costs. In such cases, brokers, usually known as VN Providers (VNP) try to perform the embedding under the mentioned limited information disclosure (LID) problem. As a result, it can be clearly observed that blockchain can solve this interaction, providing: secure sensitive data storage, customer-provider negotiation without third-parties (without VNP) and finally maintaining a coordinated process. The negotiation between the involved parties will be based on a time-limited auction system, where each virtual network request will be stored as a contract on the blockchain network.

Therefore, a good question for the theses could be: How blockchain takes advantage of distributed workflows providing an agile and secure environment? Exemplifying workflows, with the network slicing example. At the end, a decentralized application approaching the mentioned problem will be deployed. In addition, the app will include a user-friendly front-end in order to guide users through the process.

1.3 Outline

This thesis is structured as follows. In Chapter 2, relevant background on blockchain, network virtualization and auction mechanisms is given. Afterwards, an overview and analysis of existing research about blockchain in supply chain management and multi-provider virtual network embedding is presented in Chapter 3. Based on these investigation, in Chapter 4 an application system design to solve the virtual network embedding problem using blockchain is exposed. Then, the implementation of the proposed design is discussed in Chapter 5. The created application along with its technologies are tested from different perspectives in Chapter 6. Last but not least, in Chapter 7, the conclusions of this thesis are exposed as well as possible improvements for future work.

2 Background

In this section, an overview of the blockchain technology evolution will be provided. It starts with the 1st blockchain generation related to cryptocurrencies, with Bitcoin as a leading representative. Then, the second or smart contracts generation will be investigated, where Ethereum extends the idea of money transfers, to any other application that can be writable as a piece of code.

Afterwards, a scenario related to the network virtualization environment will be introduced, in which resource negotiation between customer and providers is crucial. Due to this importance, a well-known public negotiation mechanism will also be presented: the auctions.

2.1 Blockchain: A decentralized and distributed trustless ledger

A blockchain is a decentralized distributed ledger, which stores the entire history of transactions on the network. In other words, it is a simple database distributed among a network of computers, where each computer has an identical copy of this database. This contrasts with traditional (e.g. SQL) databases that are controlled by a single entity. Thus, in a blockchain, there is no central server or agent in the middle of the communication. For example, imagine a user willing to transfer money to another one 2.1. In a centralized system, the transaction will go first to the bank, who will update its internal database and subsequently perform the operation. In contrast, by a decentralized system, each user is able to directly transfer the money, since it possesses an updated copy of the database. Another example to replace a centralize design could be in the healthcare environment. There, patient records are stored in multiple databases, which always leads to a costly exchange of information between them. In this scenario, the blockchain could improve the process, preserving patients confidentiality in a secure and decentralized manner.

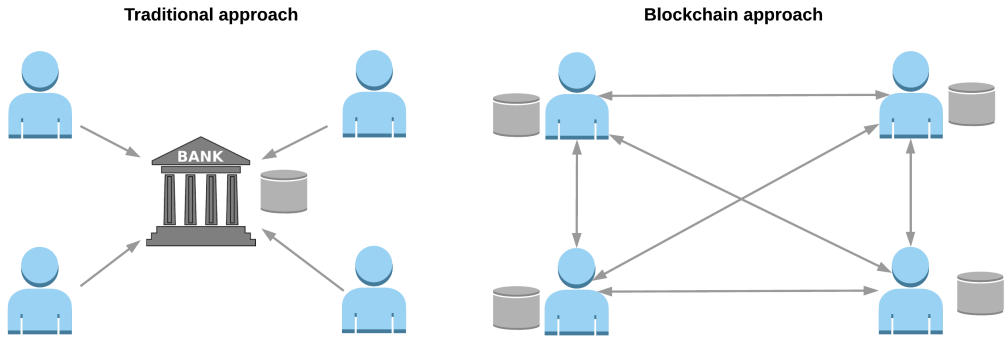

Figure 2.1: Traditional database vs blockchain approach

At the beginning, the terms Bitcoin and blockchain were sometimes interchanged, as these words were used to refer: (i) the technology, (ii) the protocol for making transactions and (iii) the cryptocurrency (money). Therefore, before continuing, one statement that needs to be clear: Bitcoin is a cryptocurrency that uses the blockchain technology. Hence, Bitcoin is just one of the multiple applications that use blockchain.

However, when a new technology appears, the first user's goal is normally to exploit its economic potential. For this reason, money transaction through cryptocurrencies was its first application. In the next subsection, we will understand what cryptocurrencies are, followed by an explanation of the Bitcoin's design architecture.

2.1.1 Blockchain 1.0: Cryptocurrencies

At of the end of January 2018, Coinmarketcap[1], a cryptocurrency market tracker, lists more than 1,400 cryptocurrencies with an aggregate value approaching USD 700bn. But what are cryptocurrencies? Cryptocurrencies are a variety of digital currencies pretending to work as a medium of exchange, such as Euro or USD does. As the name suggests, apart from being a virtual currency, they use cryptography to secure and verify its transactions. The main difference with traditional currencies is that they do not have any physical equivalent in the real world. Nevertheless, they can be used to pay goods and service, with the advantage of not being constrained with geographical or political borders. For example, GMO Internet[2], a Japanese company, will start paying parts of employees salaries in cryptocurrencies (Bitcoin).

In this paper, we will set aside whether they can become true currencies or not, and also its political, social or economic impact. The only focus will be the technology behind it, as blockchain can be extended to much more than digital currencies. Thus, we will start from the genesis of the technology, with Bitcoin as its revolutionary innovator.

Bitcoin

Bitcoin took the world by surprise in 2008, after Satoshi Nakotomo's white paper publication [Nak08] and later its software release. Bitcoin is a cryptocurrency used for making secure transactions across a peer-to-peer (P2P) network. In addition, Bitcoin uses its own protocol that operates in an overlay network, the blockchain. An overlay network is a computer network build in the top of another network, in this case above Internet application's layer, which is controlled by its users (no central authority).

From another point of view, the Bitcoin ledger can be interpreted as a state transition system, where there is an initial state, which after a transition function (money transfer), results in a new state. Imagine an where A wants to send X to B. The first state is A and B current balance and the transition function will take X from A and insert it to B's account, generating a new state. But what happens if A sends exactly the same payment to two different addresses (B and C) at the same time? This scenario is the so-called *double spending attack* and consequently, a transaction always needs to be verified by miners before being confirmed (unconfirmed transaction). Miners are specific blockchain users, responsible for monitoring and verifying all the transactions between users. If they are the first solvers, as it requires a high computational work, they are also economically rewarded. Additionally, if any transaction has a positive balance at the end, it is also destinated to the miner.

And how all these miners cooperate efficiently? The answer to this question is one of the most remarkable Satoshi innovation key factors, which consists of the communication between nodes through a simple decentralized consensus protocol. This consensus protocol consists of multiple algorithms (e.g. Proof of Work), used by the miners in the Bitcoin network.

Mining

Bitcoin needs to combine the state transition system with a consensus protocol, in order to synchronize the order of all transactions among the users. New transactions are stored in the last block of the blockchain, and a new block is mined every ten minutes. Over time, this creates an ever-growing chain of blocks, which are constantly updated. Thus the name: blockchain. Additionally, a complete history of the transactions is kept, so everyone can verify the last money movements. For instance, a blockchain can be compared to an endless domino game, where all the pieces are placed in vertical one after the other. Each of these pieces references a block, and if one block is removed (e.g. transaction error), all the subsequent ones will be affected. Therefore, miner's task is not simple and requires computational power. Figure 2.2 shows how transactions are bundle into blocks, where each block contains a:

[1] https://coinmarketcap.com/
[2] https://www.gmo.jp/en/

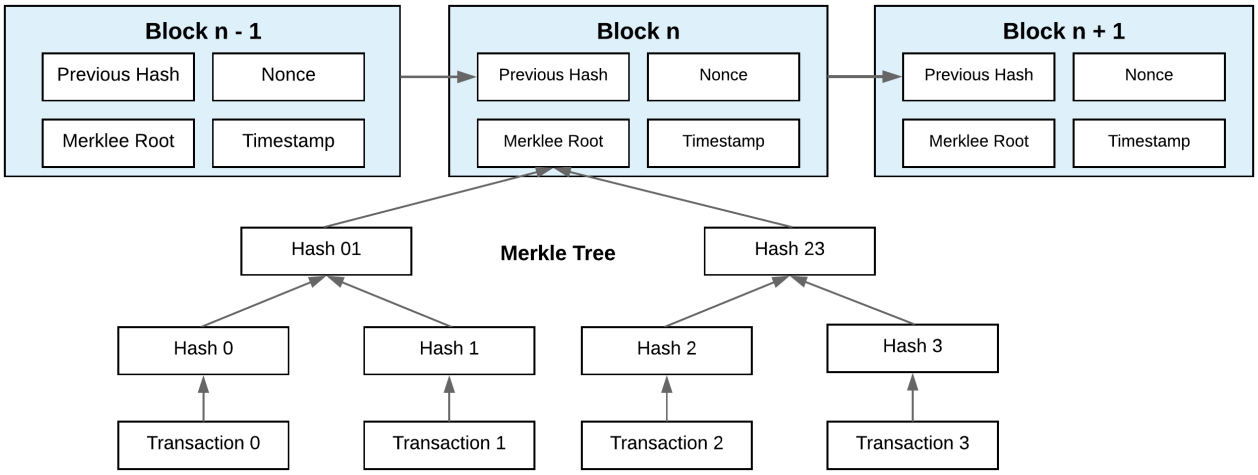

Figure 2.2: Blockchain Mining

- Timestamp, to identify when the event occurs.
- Nonce used to avoid malicious nodes from flooding the blockchain.
- Hash of its previous block, to keep track of already added blocks.
- A list of all the transactions that have been created since the previous block. To save storage, this transaction list is typically stored in a a Merkle root [Mer87].

Furthermore, each block and transaction is restricted to a size of 1MB and 250-300bytes respectively, and for a block being valid it needs to satisfy the following requirements:

- Previous block referenced exists.
- Block's timestamp is greater than previous block one.
- Proof-of-work in this block is valid.
- If any transaction from the transaction lists returns an error, exit.
- If all previous steps confirmed, store state at the end of the block and return true.

In the third step of the block validation process, appears the term Proof-of-work. Bitcoin uses the Hashcash proof of work algorithm to prove that a block miner spent some computational time creating a block. In particular, Bitcoin protocol demands that miners find an input, which is a valid $blockheader$ formed by a random value (c) and nonce (x), whose cryptographic hash (e.g $SHA256$) is less than a determined value. This value is obtained from d, the blockchain difficulty. Then, the only way to create a valid block is simply trial and error until a valid Proof-of-Work is generated. Proof-of-work is also used in other contexts, such as for limiting email spam or denial-of-service attacks (DoS).

$$F_d(\,blockheader\,) = F_d(\,c, x\,) = SHA256(\,SHA256(\,c|x\,)\,) < \frac{2^{224}}{d}$$

Key management

In Bitcoin, each user is identified by a single public address. However, how is this Bitcoin public address generated? This process involves the following steps:

1. A random 256-bit private key is created. Since this key will be used to sign the transactions, it must be kept secret.

2. A 512-bit public key is generated from the private key, using the Elliptic Curve Digital Signature Algorithm (ECDSA)[3]. This key is used for verifying private key signatures.
3. This public key is hashed to 160 bits using SHA-256/RIPEMD. Apart from size constraints, the reason for hashing the public key is that if there is a vulnerability in elliptic curves, user's money can still be safe, since only the hash is known.
4. Finally, the public key is encoded in ASCII using Base58Check. This output is the resulting Bitcoin address.

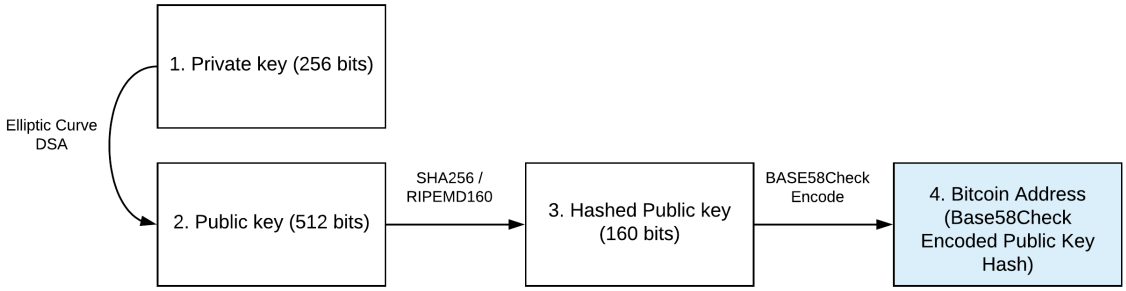

Figure 2.3: Bitcoin keys and addresses relation [Shi14]

After that, users are able to send transactions using Bitcoin. For example, imagine that user A wants to send 2 BTC to user B. Firstly, the user creates a transaction willing to transfer 2 BTC to user address B. Secondly, he signs the transaction with his private key, attaching also his public key, and sends it to the blockchain. There, miners will verify the signature using A's public key and also that its hash matches user A address. If both, the signature and the hash, are correct, the transaction is accepted and added to the next block. Transactions examples can be found in [4].

2.1.2 Blockchain 2.0: Smart contracts

The Blockchain 1.0 had numerous limitations since essentially it only approaches the decentralization of money and payments. However, the architecture implemented by Bitcoin is extensible beyond financial uses cases.

Ethereum

In 2013 Vitalik Buterin, a Russian-Canadian programmer released the Ethereum white paper [B+14], where he describes an alternative platform running in a blockchain that allows building any kind of decentralized application. In 2014 Ethereum started as a crowdfunding project, which collects small amounts of money from a large number of users. Ethereum is currently a running project and the second most valuable cryptocurrency.

Furthermore, it is built on a Turing-complete contract system. As previously explained, a Turing-complete system allows the developer to perform any computation, which runs on the so-called **smart contracts**. These smarts contracts are executed by the Ethereum nodes, each using its Ethereum Virtual Machine (EVM), or in other words, a blockchain with a built-in programming language.

In the end, smart contracts are basically Ethereum scripts that whenever they are executed, runs the function programmed inside of it. A smart contract can also be seen as the logic inside a vending machine. For instance, a buyer wants a Coke bottle that costs 2 €.. Once this buyer inserts 2 € and

[3] https://www.maximintegrated.com/en/app-notes/index.mvp/id/5767
[4] https://blockexplorer.com/

presses the button, a small program (contract) runs inside the vending machine and supplies the user with its Coke. This function can be seen as a smart contract, being for example:

```
if (button_pressed == 'Coke' && rcv_money == '2')
 return coke
```

Therefore, smart contracts allow anyone to write their own functions, in just a few lines of code. Despite Bitcoin and Ethereum have similar features, such as decentralization or transaction-based, Ethereum makes use of smart contracts, which potentially opens up real-world use cases. Bitcoin and Ethereum can be comparable to a calculator (one application) and a smartphone (multiple applications) respectively. In addition to the differences already mentioned (and many more), there are three that need to be highlighted:

- Ethereum's block time is shorter, specifically around 14s. Remember that Bitcoins mining time was around 10 minutes.
- Bitcoin's blocks and transaction sizes are indicated in bytes, however, in Ethereum depends on the contract complexity.
- Ethereum is account-based and not transaction-based.

Therefore, Ethereum introduces a new concept called accounts, which are unique 20-byte addresses in the blockchain. Each of these accounts has an account balance controlled by ethers (ETHs), the digital currency used by Ethereum. In Ethereum, there are two types of accounts (Figure 2.4):

- **Externally Owned Accounts (EOAs):** These accounts normally identify a user, being controlled by their own public and private keys. Only EOA can send transactions to other accounts. A transaction in Ethereum can either send Ethers or call/trigger a contract account
- **Contract accounts:** These are the accounts, where the code (smart contract) is stored. Thus, once triggered (function call from another contract), its defined code is executed.

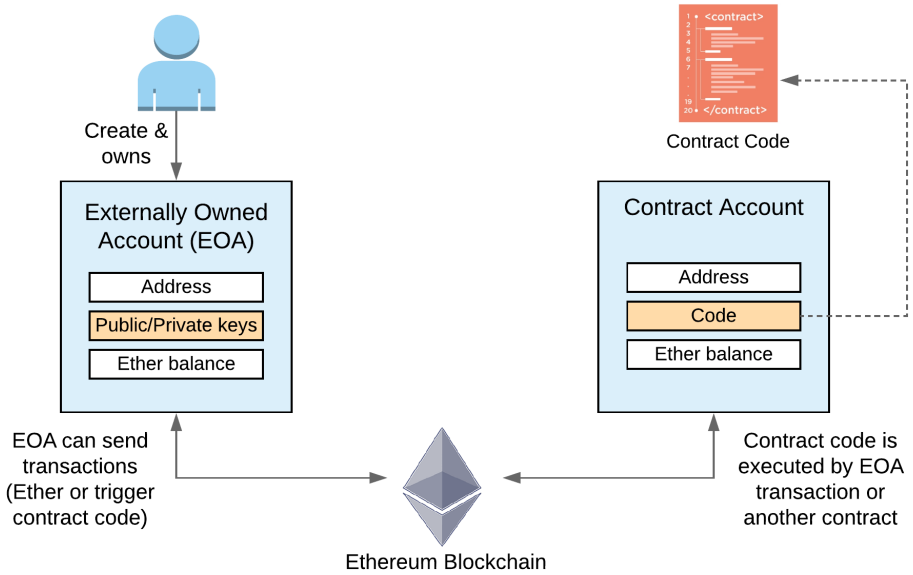

Figure 2.4: Ethereum accounts: Externally owned accounts and contract accounts

However, what happens if a code results in an infinite loop? The node will get stuck the whole time executing this contract. Solution: Ethereum uses the term "gas". Gas is an amount of ether used for

executing contracts in the Ethereum blockchain. For example, if an EOA calls a contract account, which is programmed to make a money transfer to another EOA, apart from the ether transferred, it will also spend an additional amount of ether (gas). Thus, the ether used for a transaction, or transaction fee is:

$$Ether = gasprice \times gaslimit + value$$

Firstly, the gas limit is the maximum amount of gas that a sender is willing to spend on a transaction. Fortunately, all unused gas during a transaction (gas limit - real gas used), is refunded to the sender. Nevertheless, if a transaction gives an error, e.g: a fake transaction, this provided ether (gas limit) will never be refunded. Secondly, the gas price is its relation with real Ether, e.g: 40 Gwei (4e-8 ether). Thirdly, the value is the amount of ether transferred to the other EOA.

In conclusion, a smart contact is just an account containing code, which lives on the blockchain and allows developers to create their own decentralized applications.

Decentralized applications

In the Ethereum white paper [B+14] dapps are split into three types. The first group includes financial applications, where users exchange ether in an efficient and distributed manner. The second group is formed by semi-financial applications, in which money is involved but it is mixed with data from the real world, for example with weather feed. And finally, in the third category, there are non-financial applications. For instance, an online voting platform providing a better transparency into elections, without compromising voter confidentiality.

Despite the blockchain is a secure technology, its created applications are constantly compromised to multiple attacks. This is because the majority of smart contracts are insecure, due to its code being prone to bugs. For example, in June 2016, the DAO [5], a distributed autonomous organization instantiated on the Ethereum blockchain, was hacked. The attack, which was a smart contract bug, had resulted in over 50 million dollars of cryptocurrency theft. Thus, smart contracts need to be protected against malicious attacks before being uploaded on the blockchain.

In the following section, an example application or process currently performed by IT companies will be introduced.

2.2 Network Virtualization

Network virtualization enables simulating hardware networks, in software (Figure 2.5.a). These software networks are called virtual networks, which provide flexibility, high manageability, and a low cost. Network virtualization handles two main concepts: node virtualization and link virtualization. The former implies sharing the node's physical resources located in the substrate network (e.g: CPU, storage, memory) to multiple virtual nodes from the virtual networks. The latter enables the transport of multiple virtual links in a single and shared physical link. The mentioned layers and elements involved in network virtualization, are shown in Figure 2.5.b.

In a real scenario, there are two main actors (and one optional) participating in the network virtualization process (Figure 3.2.a):

- **Infrastructure providers (InPs):** Infrastructure providers (e.g: Deutsche Telekom, Telefonica) deploy and manage the physical nodes, sharing its resources to the virtual network.
- **Service Providers: (SPs)** Service providers lease virtual resources from infrastructure providers to create its own virtual network.

[5] https://ethereum.org/dao

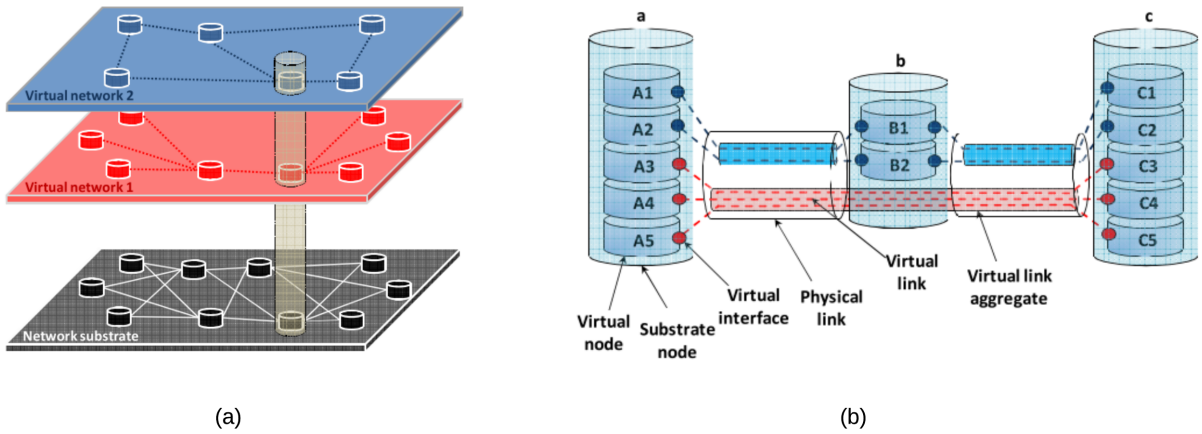

Figure 2.5: Network virtualization model and its basic elements [CJ09]

- **Virtual network providers: (VNPs)** Are the brokers, which can optionally serve as intermediaries between the InPs and the service providers.

In this paper a specific example of network virtualization will be investigated, where SPs are willing to embed resources among one or multiple infrastructure providers, in a constrained scenario.

2.2.1 Multi-Provider Virtual Network Embedding

To deploy wide-area networks, virtual network resources need to be embedded across different infrastructure providers instead of just one InP. As a result, the system avoids being restricted to just a single provider topology, providing at the same time better efficiency. This process is composed of two main tasks:

- **VN request partitioning**: Divide the virtual network into groups of virtual nodes in order to minimize the cost of the VN setup.
- **VN segment mapping**: Map the virtual resources to the InPs physical nodes.

The last two functions are performed by the VNPs and InPs respectively. Furthermore, as reported by [DRP15], the partitioning suffers mainly from the limited information disclosure problem (LID), in which VNPs have a very limited knowledge of InPs internal architecture. The reason for such a problem is that the InPs are not willing to broadcast their resources information (e.g: nodes availability, cost) or their network topology to the outside world. And if that was not enough, this information is crucial for the partitioning of resources in order to obtain a fair negotiation between the SPs and InPs. Solving this issue will be one of the main focuses of our thesis.

On the other hand, in such scenarios exists two types of communication:

- **Horizontal communication** is the one established between infrastructure providers to guarantee the most efficient end-services. These relations stated in [ZXB10], can be: *public relations* established using a market mechanism (e.g: an auction), or *private relations* already arranged. In both cases, the relation arises from the need to negotiate and cooperate to serve the SPs.
- **Vertical communication** emerges from the negotiation between SPs and InPs, the former willing to lease some virtual resources from the latter. This communication is facilitated by a third party (VNP), an intermediary that forwards SPs request to the relevant InPs.

In addition, in order to fulfill this public services negotiation between the SPs and InPs, an open mechanism will also be presented: the auctions.

2.3 Auction Mechanisms

An auction is a public negotiation method involving buyers and sellers that enables resource trading between P2P users. While buyer's goal is to find the desired service at the lowest price, providers or seller's goal is to catch the attention of buyers who are able to pay the highest price for its offered services. Thus, an auction must provide fairness in terms of technical and economic efficiency, for both the buyer and the seller.

Firstly, auctions can be divided into (i) *one-sided auctions* where only the buyers submit their bids (e.g: a painting auctioned in an art auction), or (ii) *two-sided auctions* in which buyers and sellers submit their bids. Normally, when any of the two sides (buyer or seller) cannot perform a good estimation of the service, a one-side auction is preferred. Secondly, the auctions are (i) *open cry* auctions, where the bids are broadcasted to all users (traditional type), or (ii) *sealed-bid* auctions in which bidders first commit bid values in secret that are later revealed (e.g: physical envelopes).

2.3.1 Vickrey Auction Model

Many types of auction models exist, such as the English or Dutch auctions [CST80], however as previously discussed a fair-price system is extremely needed. The Vickrey auction model [Vic61] matches these requirements, achieving a reasonable price for the buyer by motivating bidders to bid truthfully. One of the Vickrey auction model important aspects is that corresponds to a *second-price* auction. In a second-price auction the bidder will offer a price for the service, and among all the bids, the higgest one will win the auction. Nevertheless, unlikely standard auctions, the service will be rendered at the second highest bid. For example, if the winning bidder bids 15 €and the second highest bid is 10 €, the winner pays only 10 €. From the bidder point of view means that he can safely bid the true value, knowing that if he wins, he will pay less than his bid. However, the second-price concept needs to be thoroughly investigated as long as a bidder can submit multiple bids. Imagine an scenario, where a single bidder A submits two bids: {1 €,9 €} for a service with a real cost 5 €. In this case, this buyer will end up paying 1 for the service. Therefore, the second-price auction model needs always to be compared between bidders and not between bids. Suppose the last example, with a second bidder B submitting {5 €, 6 €}. At the end user A, will win the bid but paying 6 €, risking that if he would have been the only user bidding, he would have payed 9 €.

2.4 Summary

The blockchain is a transparent, secure and robust system where users are in charge of their own accounts and transactions. Beyond money transfer, which starting with Bitcoin, the blockchain technology can be used as a software connector in multiple scenarios. Due to smart contracts, any kind of application could be implemented, enabling developers using the BC concept in their own environment. In the next chapters, the already presented network virtualization scenario, will be further investigated.

3 Related Work

In the following, we provide a comprehensive overview of supply chain management, in particular about a blockchain application, in order to demonstrate BC technology potential benefits. Then, we present related work on already implemented frameworks for virtual network embbedding across multiple InPs, along with the auctions mechanisms used. Finally, we will conduct an analysis of the prior VNE related work, arguing why blockchain could solve and improve the problem described.

3.1 Blockchain in Supply Chain Management

In the last years, managing the information during the lifecycle of a product represents a major challenge for all companies ([KARF03], [Tut02], [KH02]). This product information has to be most of the time accessible and alterable among the parties of a supply chain. Thus, a shared platform supplying data distribution and storage is extremely needed.

However, normally most companies store and maintain its data into their company-specific or public infrastructures (AWS[1]). Accordingly, each enterprise creates its own copy of the product data, using their own protocols and procedures. As a consequence, lots of information asymmetries are found, which ends up destroying the benefits of knowledge or data sharing. In [LB92], [Fia05],[LZ08] inadequate definition of customer services, poor coordination or organizational barriers are named to be some of the pitfalls that customer and providers suffer from.

The following section explores an example scenarios using blockchain technology. Firstly, a blockchain application in the supply chain for a manufacturing system will be presented. Afterwards, we will enumerate the lessons learned from the prior use case.

3.1.1 Blockchain Ready Manufacturing Supply Chain

In this part, the blockchain example [AM16] will be discussed, which stores and distributes specific product information during its lifecycle. This task also named Product Lifecycle Management (PLM [Sta15]), involves different stages in which multiple parties modify the product data. In Figure 3.1.b shows the typical stages on the cycle. In this scenario, each actor interacts with a user interface, which is connected to the product's data from the blockchain. Each product is created in form of rules (smart contract) so that only specific users can access or modify this information. To access or insert data into the contract, parties must authenticate themselves signing the request with their private keys. During the lifecycle management, a product is owned by an entity (e.g. distributors). Moreover, when a product goes to another actor (e.g. from distributors to a consumer), both parties will sign a contract that updates product's ownership. Thus, the system guarantees that only the granted user can modify the contract in each phase.

In [AM16], an example is also presented to better explain the mentioned approach. The application consists of the product lifecycle of a cardboard box, starting from the trees cut down until the box is recycled, in which the involved parties are previously registered to the system. In Figure 3.1.b an overview of the implemented design is presented. In this business process, different actors are entailed. For instance, the box manufacturer first receives the physical object and then signs a contract to gain the product's access. Afterwards, he will be allowed to alter the product data through the software application (e.g. box quality). In the end, he will transfer the package (signed) to the next actor, in particular, the product filler.

[1] https://aws.amazon.com/

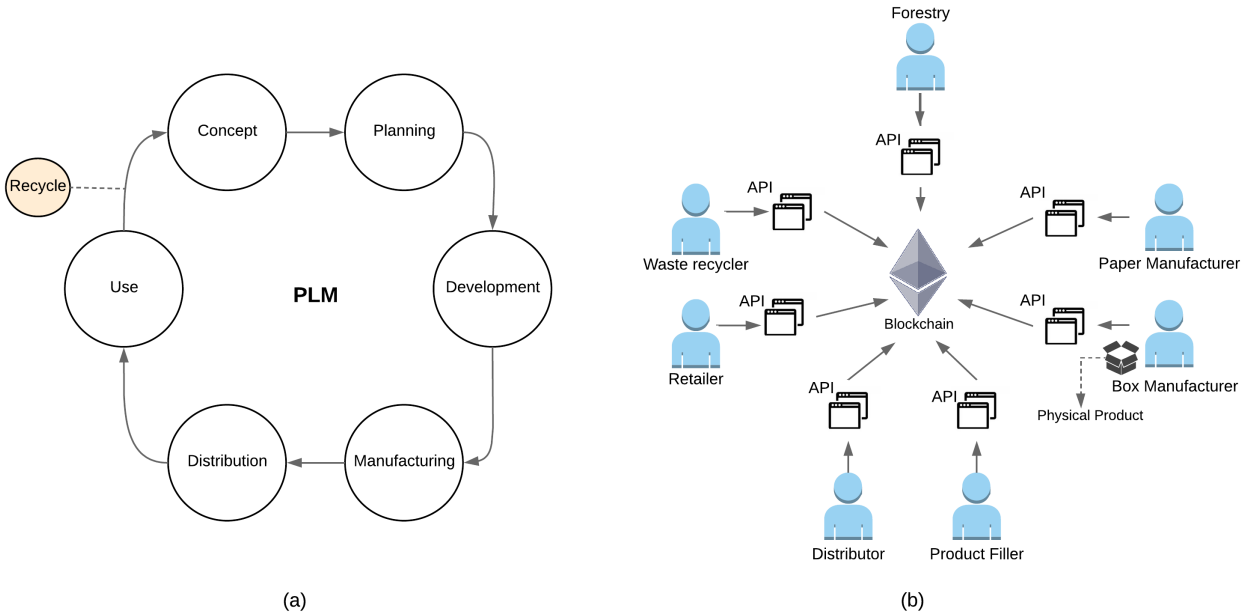

Figure 3.1: PLM stages and cardbox blockchain application example [Sta15]

3.1.2 Lessons Learned

Through relevant examples in supply chain management, as the above presented, we can observe the extraordinary potential of blockchain in multiple areas. Despite the last paper does not include detailed specifications, such as the blockchain's type or mining strategy, a useful system design has been provided, which help us in defining our possible architecture. In addition, some of the main benefits compared to centralized technologies are observed:

- **No trusted third-party:** By using the blockchain, all the significant costs of involving a middle-party are suppressed[MS15]. Intermediaries are not required for realizing transactions or simply reaching an agreement. For instance, in the last example, users were able to transfer ownership, by signing the product's contract and without the need of a third party.
- **Autonomy:** After an owner deploys a smart contract on the BC, it is then maintained by all the participating users, and not only by its creator.
- **Data management:** Blockchain helps us in coordinating, validating and storing the distributed data in a decentralized manner.
- **Technology possibilities** As actors can define and establish their own rules (smart contracts), new business models open up.
- **Privacy:** Thanks to cryptography, users can interact with unknown partners. For example, in the above example, all actors can coordinate its services through the PLM process, without knowing the next actor involved.
- **Transparency and immutability:** All the changes that are made to the contract data are directly visible to the other users. In addition, once a transaction is executed, it cannot be altered or deleted.

Nevertheless, the technology still faces lots of challenges, such as scalability or incorporating external information, which will be later investigated.

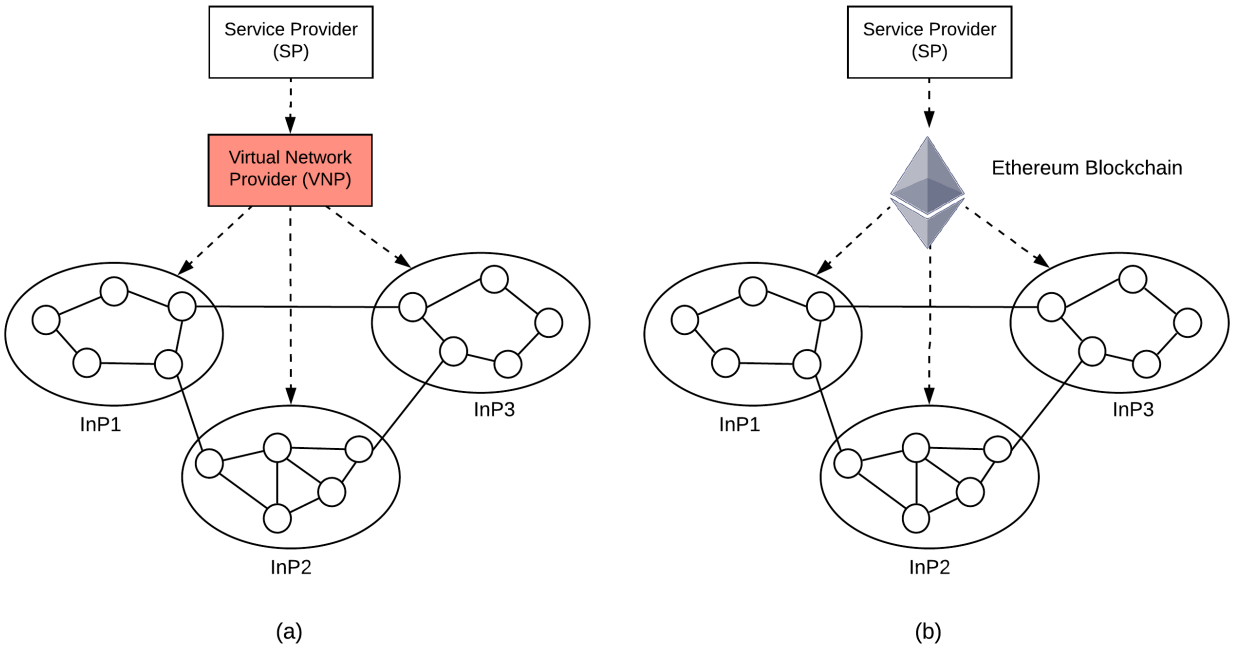

Figure 3.2: Virtual network request accross multiple InPs with and without VNP [DRP15]

3.2 Multi-provider Virtual Network Embedding

In the last years, many investigations in VNE have been conducted by different authors [HLAZ11], [ZZSRR08], [CRB09]. Among these, it is important to highlight topics such as the trade-offs between single [CRB09], [HLZ08] and multi-provider VNE [DARP17], or the different techniques used for achieving VN efficiency, which decompose VNE into two tasks: VN partitioning and VN segment mapping [FBB+13]. Each of these studies proposes new interesting aspects or features compared to the prior research. Nevertheless, in this paper, we will enter in the discussion of the different scenarios. The main goal of this work will be to explore existing solutions, in particular, frameworks for multi-provider VNE applications that suffer from LID [DARP17], [ZXB10], [EDPM13], [CSB10].

3.2.1 Centralized and Decentralized approaches

UNTIL HERE!

Multi-provider virtual network embedding consists of the allocation of virtual resources, in a constrained and limited scenario. However, as previously introduced in the background, InPs are not willing to cooperate or disclose internal information. In [DRP15] the LID problem is approached regarding:

- **Virtual resources availability**: In this case, virtual resources examples are obtained from Amazon EC2 [ama], which announces the attributes of different instances types (CPU, memory, cost) to give the user facility in choosing the virtual resource that better fits in its application.
- **Network topology**: Most of network topology information is treated confidentially for the InPs. Nevertheless, there are certain aspects of the network which are not considered private, such as InPs peerings (including location) and its related link cost.

Thus, in the mentioned research, VNPs use above information to perform the resource partitioning. Despite the results in embedding costs are not much higher than the ideal case scenario (InPs announcing

all required information), they could be further improved enabling the InPs directly interacting in the process.

However, what if the SPs and InPs are not willing to trust a centralized broker? Here, is when blockchain could play a crucial role (Figure 3.2.b). Thanks to the blockchain, more precisely to Ethereum and smart contracts, the untrustworthy VNP could be replaced with a secure, flexible and coordinated end-system. Hence, InPs disclosed information will be kept confidential (public/private key cryptography) in the blockchain and just accessible for its desired users.

VN embedding framework Linear programming formulation used is nice, but we have the problem of the LID suboptimality. How to solve it? Auctions mechanism maybe... VNE framework.

[DARP17] graph topology maybe...

[ZXB10] VNE framework

Service negotation has already been investigated in [HS05], [OV02], who have worked with diverse auctions platforms.

the pros and cons of different types of auctions will also be presented. which could be accomplished through a limited time auction.

The idea of using auctions for a distributed slice allocation has been floated before: in V-Mart [13], InPs submit their bids on a subset of the slice to the auctioneer SP that repeats the auctions for a second round to a selected set of InPs. V-Mart ensures a fair market but does not guarantee performance.

Accordingly, a *sealed one-side auction* is the one which best meets our system specifications, since the InPs do not want to disclose their network topology or nodes information. Our auction procedure will start with the SP requesting a virtual network (a graph of virtual nodes) along with an upper bound, the last being the maximum amount of money the SP is willing to pay. Then, the InPs will estimate the request requirements (e.g: nodes availability or costs) in their infrastructure and thereafter submit their bids. In other words, the InPs will be sellers submitting bids and SPs the consumers requesting a service. Despite it can be seen as a fair system, how can the SP ensure that the InPs are offering prices proportional to the current market costs?

It is important to note that in our scenario the **lowest auction** will win and not the highest. The reason is that at the end the SPs is who will pay the bid value. For instance, suppose that all the InPs make an offer close to the upper bound specified by the SP. If the highest bid wins, the SPs will always end-up paying approximately its upper bound (unfair). However, if the lowest bid wins, the system ensures that the InPs submits a bid proportional to the real cost since it will be the minimum amount of money that they could earn.

Ultimately, after judging the preceding analysis along with the pros and cons of each auction type and model, the ***second-price sealed-bid*** or simply the ***Vickrey auction*** fulfills the application requirements, providing the highest aggregate profit among sellers and buyers. [EDPM13] super example with consensus protocol! GOOD images

During this paper, the Ethereum blockchain will be used to perform a service negotiation (Vickrey auction) and later to track its possession throughout the supply chain process. This will be exemplified by a web application implementation, focused on network slicing. Thus, blockchain strengths and weaknesses will be further investigated.

In [DARP17], [ZXB10], the actors rely on a central entity or server, called the virtual network provider. Therefore, we assume that this process could be enhanced by using a decentralized technology, e.g. blockchain. In addition, already deployed decentralized approaches regarding VN embedding [EDPM13], [CSB10] will also be in our spotlight.

3.2.2 Auction Mechanisms

Several authors, notable Wellman and Wurman [11], have pointed out the difficulties of truly simulation open cry auctions in the unreliable and insecure environment of the internet. Study [AM+06] with equations and everything...

3.3 Analysis of Related Work

Reference	Description	Findings	Limitations
A	...	...	...
A	...	...	...
A	...	...	...
A	...	...	...
A	...	...	...
A	...	...	...
A	...	...	...

Table 3.1: Comparison of implemented approaches for VNE

We do not claim that our approach is the best or the only way of performing VN embedding. On the contrary, it is more an add-on feature that can be merged with most of the existing solutions. In this scenario, the main goals are to remove the middleware or virtual network provider, by using a new prominent technology such blockchain. Thus, a new decentralized, automated and secure system will be created between the SPs and InPs, which benefits from the blockchain potentials. Furthermore, to the best of our knowledge, this is the first approach using blockchain in this scenario, and thanks to smart contract's flexibility, it can serve as a starting point for all upcoming investigations.

Hint:

This chapter should give a comprehensive overview on the related work done by other authors followed by an analysis why the existing related work is not capable of solving the problem described in the introduction. The chapter should have a length of about three to five pages!

The blockchain technology, in particular, secure smart contracts on the Ethereum platform, will be used to approach a real-world scenario. A decentralized application (third category) that enhances supply chain performance, will be implemented. More precisely, network virtualization providers and customers will improve its communication, through the use of a web application (front-end) connected to the blockchain (back-end). In the following section this concept will be further investigated.

3.4 Summary

4 Design

Hint:

This chapter should describe the design of the own approach on a conceptional level without mentioning the implementation details. The section should have a length of about five pages.

1. Types of blockchain and BC decision (graphs: Why private and not traditional? economical reasons: cost, maintenance, computing power... 2. Scenario, architecture 3. Notification system (P/S) 4. Mining 5. Authentication system

1. Invite People to the private BC. Users fixed amount of Ether (Proof of Stake). Garzik, J. 2015. "Public Versus Private Blockchains.

2.A DApp has its backend code running on a decentralized peer-to-peer network. Contrast this with an app where the backend code is running on centralized servers. A DApp can have frontend code and user interfaces written in any language (just like an app) that can make calls to its backend. Furthermore, its frontend can be hosted on decentralized storage such as Swarm or IPFS.

What can be externally extracted from a contract? Since value transfers cannot be blinded in Ethereum, anyone can see the value and therefore the highestBid. Sol: We can just see the address. That's a problem if there are just a few users. Do research on amount of users needed to solve the problem.

4. Mining from users or supernode? Start with supernodes (like in Bitcoin at the beginning), when more users maybe a user say he want to become a miner (gets the privileges and anonimity increased)

And why is a blockchain better than a traditional database? The truth of it is that there is no better technology than the others, it just depends on user's requirements. However, blockchain provide multiple advantages such as:

- **Disintermediation:** Running without a central administrator. This is possible because it uses its own consensus mechanisms, such as the Proof-of-Work (later explained).
- **Privacy:** Data is encrypted and stored on the blockchain, and it also requires user authentication.
- **Robustness:** Different databases do not need to be synchronized across different boundaries. All the information is stored in a single blockchain.
- **Transparency:** All the transactions are logged on the system.

4.1 Requirements and Assumptions

e.g: Flexibility, fair price negotiation, trustworthy, transparent ... And explain each point Assumption that intra-domain virtual links cost is almost 0. Having the virtual link cost from EC2 Amazon and the BW we could determine it...

4.2 System Overview

4.2.1 Component 1

4.2.2 Component 2

4.3 Summary

5 Implementation

1. How a contract is compiled in to the Ethereum BC. 2. Technologies used (GETH vs ETH NW, web3, Solidity, truffle). (The Genesis block is the first block in a blockchain. It is always hardcoded in the blockchain). 3. Authentication 4. Component relations in the source code 5. Code examples

3.Investigate web3.eth.sign(web3.eth.accounts[0], address), the address should be unlocked. How this is possible?

Hint:

This chapter should describe the details of the implementation addressing the following questions:

1. What are the design decisions made?
2. What is the environment the approach is developed in?
3. How are components mapped to classes of the source code?
4. How do the components interact with each other?
5. What are limitations of the implementation?

The section should have a length of about five pages.

5.1 Design Decisions

5.2 Architecture

5.3 Interaction of Components

5.4 Summary

6 Evaluation

Hint:

This chapter should describe how the evaluation of the implemented mechanism was done.

1. Which evaluation method is used and why? Simulations, prototype?
2. What is the goal of the evaluation? Comparison? Proof of concept?
3. Wich metrics are used for characterizing the performance, costs, fairness, and efficiency of the system?
4. What are the parameter settings used in the evaluation and why? If possible always justify why a certain threshold has been chose for a particular parameter.
5. What is the outcome of the evaluation?

The section should have a length of about five to ten pages.

Test scenario with Real users using our application.

6.1 Goal and Methodology

6.2 Evaluation Setup

6.3 Evaluation Results

6.4 Analysis of Results

7 Conclusions

Hint:

This chapter should summarize the thesis and describe the main contributions of the thesis. Subsequently, it should describe possible future work in the context of the thesis. What are limitations of the developed solutions? Which things can be improved? The section should have a length of about three pages.

7.1 Summary

7.2 Contributions

7.3 Future Work

IOTA-TANGLE The Front-end application could be hosted in a decentralized storage such as IPFS or SWARM. BC storing chunks of data

7.4 Final Remarks

Bibliography

[AM+06] Lawrence M Ausubel, Paul Milgrom, et al. The lovely but lonely vickrey auction. *Combinatorial auctions*, 17:22–26, 2006.

[AM16] Saveen A Abeyratne and Radmehr P Monfared. Blockchain ready manufacturing supply chain using distributed ledger. 2016.

[ama] Amazon ec2 instance types. `http://aws.amazon.com/ec2/instance-types`. Accessed: 2018-01-16.

[B+14] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 2014.

[CJ09] Jorge Carapinha and Javier Jiménez. Network virtualization: a view from the bottom. In *Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*, pages 73–80. ACM, 2009.

[Cla17] Jen Clarck. Blockchain Technology: the Next Generation of the Internet, March 2017. accessed:2017-11-27.

[CRB09] NM Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, and Raouf Boutaba. Virtual network embedding with coordinated node and link mapping. In *INFOCOM 2009, IEEE*, pages 783–791. IEEE, 2009.

[CSB10] Mosharaf Chowdhury, Fady Samuel, and Raouf Boutaba. Polyvine: policy-based virtual network embedding across multiple domains. In *Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures*, pages 49–56. ACM, 2010.

[CST80] Vicki M Coppinger, Vernon L Smith, and Jon A Titus. Incentives and behavior in english, dutch and sealed-bid auctions. *Economic inquiry*, 18(1):1–22, 1980.

[DARP17] David Dietrich, Ahmed Abujoda, Amr Rizk, and Panagiotis Papadimitriou. Multi-provider service chain embedding with nestor. *IEEE Transactions on Network and Service Management*, 14(1):91–105, 2017.

[DRP15] David Dietrich, Amr Rizk, and Panagiotis Papadimitriou. Multi-provider virtual network embedding with limited information disclosure. *IEEE Transactions on Network and Service Management*, 12(2):188–201, 2015.

[EDPM13] Flavio Esposito, Donato Di Paola, and Ibrahim Matta. A general distributed approach to slice embedding with guarantees. In *IFIP Networking Conference, 2013*, pages 1–9. IEEE, 2013.

[FBB+13] Andreas Fischer, Juan Felipe Botero, Michael Till Beck, Hermann De Meer, and Xavier Hesselbach. Virtual network embedding: A survey. *IEEE Communications Surveys & Tutorials*, 15(4):1888–1906, 2013.

[Fia05] Petr Fiala. Information sharing in supply chains. *Omega*, 33(5):419–423, 2005.

[HLAZ11] Ines Houidi, Wajdi Louati, Walid Ben Ameur, and Djamal Zeghlache. Virtual network provisioning across multiple substrate networks. *Computer Networks*, 55(4):1011–1023, 2011.

[HLZ08] Ines Houidi, Wajdi Louati, and Djamal Zeghlache. A distributed virtual network mapping algorithm. In *Communications, 2008. ICC'08. IEEE International Conference on*, pages 5634–5640. IEEE, 2008.

[HS05] David Hausheer and Burkhard Stiller. Peermart: The technology for a distributed auction-based market for peer-to-peer services. In *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*, volume 3, pages 1583–1587. IEEE, 2005.

[KARF03] Mikko Kärkkäinen, Timo Ala-Risku, and Kary Främling. The product centric approach: a solution to supply network information management problems? *Computers in Industry*, 52(2):147–159, 2003.

[KH02] Mikko Kärkkäinen and Jan Holmström. Wireless product identification: enabler for handling efficiency, customisation and information sharing. *Supply chain management: an International journal*, 7(4):242–252, 2002.

[LB92] Hau L Lee and Corey Billington. Managing supply chain inventory: pitfalls and opportunities. *Sloan management review*, 33(3):65, 1992.

[LZ08] Lode Li and Hongtao Zhang. Confidentiality and information sharing in supply chain coordination. *Management science*, 54(8):1467–1481, 2008.

[Mer87] Ralph C Merkle. A digital signature based on a conventional encryption function. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 369–378. Springer, 1987.

[MS15] Michael Mainelli and Mike Smith. Sharing ledgers for sharing economies: an exploration of mutual distributed ledgers (aka blockchain technology). 2015.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[OV02] Elth Ogston and Stamatis Vassiliadis. A peer-to-peer agent auction. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*, pages 151–159. ACM, 2002.

[Shi14] Ken Shirriff. Bitcoins the hard way: Using the raw bitcoin protocol, February 2014. accessed:2018-02-01.

[Sta15] John Stark. Product lifecycle management. In *Product Lifecycle Management (Volume 1)*, pages 1–29. Springer, 2015.

[Swa15] Melanie Swan. *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.", 2015.

[TT16] Don Tapscott and Alex Tapscott. *Blockchain Revolution: How the technology behind Bitcoin is changing money, business, and the world*. Penguin, 2016.

[Tut02] A Tuttle. Who do you trust. *Industrial Distribution*, 91(3):17, 2002.

[Vic61] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1):8–37, 1961.

[ZXB10] Fida-E Zaheer, Jin Xiao, and Raouf Boutaba. Multi-provider service negotiation and contracting in network virtualization. In *Network operations and management symposium (NOMS), 2010 IEEE*, pages 471–478. IEEE, 2010.

[ZZSRR08] Yaping Zhu, Rui Zhang-Shen, Sampath Rangarajan, and Jennifer Rexford. Cabernet: Connectivity architecture for better network services. In *Proceedings of the 2008 ACM CoNEXT Conference*, page 64. ACM, 2008.