Storing digital certificates using blockchain

Fredrik Wegelid

DIVISION OF INNOVATION ENGINEERING | DEPARTMENT OF DESIGN SCIENCES
FACULTY OF ENGINEERING LTH | LUND UNIVERSITY
2019

MASTER THESIS

Storing digital certificates using blockchain

Fredrik Wegelid

LUND
UNIVERSITY

Storing digital certificates using blockchain

Published by
Department of Design Sciences
Faculty of Engineering LTH, Lund University
P.O. Box 118, SE-221 00 Lund, Sweden

Subject: Innovation Engineering (INTM01)
Division: Innovation Engineering
Supervisor: Jessica Wadin, Division of Innovation Engineering, Department of Design Sciences, Faculty of Engineering LTH, Lund University
Co-supervisor: Emil Åkesson, Division of Innovation Engineering, Department of Design Sciences, Faculty of Engineering LTH, Lund University
Examiner: Lars Bengtsson, Division of Innovation Engineering, Department of Design Sciences, Faculty of Engineering LTH, Lund University

Abstract

The purpose of this thesis is to investigate what the blockchain technology is and how it can be used to store digital certificates. Blockchain has gotten lots of attention since it was presented in 2008 for the digital currency Bitcoin. Certificates are present in many different industries.

The work done in this thesis has an exploratory approach. The work performed consists of a case study and the development of an application based on the blockchain technology. The goal of the case study is to understand how certificates are stored and accessed today. The application was built with Hyperledger Sawtooth, an open source project hosted by the Linux Foundation. In addition to this a literature study, focused on the blockchain technology, has been conducted.

The results of the case study indicate that there is no single way to access certificates today. It depends on the industry, the body that issued the certificate and the underlying standard. There are attempts to gather certificates from different sources in one place, but these are not complete.

The application developed attempts to solve the problems identified in the case study. Thanks to the blockchain technology it is transparent, secure and decentralized. This shows that the blockchain technology can be used to store digital certificates and can potentially create value if the application is developed further.

Keywords: Blockchain, Digital certificates, Hyperledger Sawtooth, Certification

Sammanfattning

Syftet med denna rapport är att undersöka vad blockkedjeteknologin är och hur den kan användas för att lagra digitala certifikat. Blockkedjor har fått mycket uppmärksamhet sedan det först presenterades 2008 för den digitala valutan Bitcoin. Certifikat finns inom många olika industrier.

Arbetet i denna rapport har en explorativ ansats. Arbetet innefattar en fallstudie och utvecklingen av en applikation baserad på blockkedjeteknologin. Målet med fallstudien är att förstå hur certifikat lagras och används idag. Applikation har byggts med hjälp av Hyperledger Sawtooth, en öppen programvara som görs tillgänglig av Linux Foundation. Utöver detta har en litteraturstudie på blockkedjeteknologin gjorts.

Resultaten från fallstudien visar på att det inte finns ett sätt att komma åt certifikat idag. Det beror på industrin, organet som utgett certifikatet och den underliggande standarden. Det finns försök att samla certifikat från olika källor, men dessa är inte kompletta.

Applikationen som utvecklats försöker lösa de problem som identifierats i fallstudien. Tack vare blockkedjeteknologin är den transparent, säker och decentraliserad. Detta visar på att blockkedjeteknologin kan användas för att lagra digitala certifikat och kan eventuellt skapa värde om applikationen vidareutvecklas.

Nyckelord: Blockkedjor, Digitala certifikat, Hyperledger Sawtooth, Certifiering

Acknowledgments

I would like to thank everyone at the Innovation Engineering division who has helped me create this thesis. There has been lots of insightful discussions and good feedback during our meetings.

I would also like to thank all external parties that have helped me gather the information I have needed. A special thanks goes out to those who have taken their time for an interview with me.

To Satoshi Nakamoto, wherever and whoever you are: thank you for creating blockchain!

Lund, January 2019

Fredrik Wegelid

Table of contents

List of acronyms and abbreviations

API	application programming interface
CPU	central processing unit
IoT	internet of things
IP	internet protocol
PoW	proof-of-work
PoET	proof of elapsed time
REST	representational state transfer
RQ	research question
SHA	secure hash algorithms
SRS	software requirement specification
SWEDAC	Swedish board for accreditation and conformity assessment
SWETIC	Swedish association for testing, inspection and certification
TCP	transmission control protocol

1 Introduction

This chapter gives the reader an introduction to this thesis. The first section gives a background on certificates and the blockchain technology. After that the problem statement, purpose, research questions and delimitations of the thesis are presented.

1.1 Background

Standards and certifications have existed for a very long time. Standards is a way to ensure quality and to make different systems work together. A standard is a set of requirements that can be fulfilled. The structure of a standard is very different depending on the area for which the standard is applicable. Certification is the process of determining whether a standard has been achieved or not. Today they are a lot more structured and legitimate than they have been throughout history.

Imagine a farmer in the 15th century. This farmer had no need for formal certifications because they did not exist. He might have some set of requirements that he or his father had made up. These requirements were however not part of a standard that all farmers followed and there was no one verifying that the farmer followed the requirements. He could sell his grains to a villager, who in turn might recommend his neighbour to buy from that farmer if the grains were of good quality. In one way, this is a kind certification process. The villager who bought the grains evaluates them from some other set of requirements that she has. If they fulfil these requirements she recommends them, and in a very informal way she has certified that farmer and his products.

Today the process for becoming certified is more complex. The recommendation from your neighbour that the nearby farmer has good quality grains might not be enough when there are multiple steps between that farmer and the product being in your hand. Standards such as KRAV, a Swedish label for organic products, exist as a set of requirements for food that is organically and socially sustainable [1]. This allows customers to verify that the food they are buying is grown in a sustainable way.

Certification bodies exist to inspect companies, e.g. a farm, and determine whether the requirements set by organisations such as KRAV are fulfilled. If they are, the

farm receive a certificate that shows consumers that the food they are about to eat fulfils the requirements set by KRAV.

These certificates can be shown in many different ways. In the case of a consumer buying a product the KRAV-label on the package might be enough. A bakery that produces bread baked on only organically grown grains might want a better proof that the seller of the grains actually is KRAV-certified. The process to verify a certificate is very different depending on the underlying standard and the certification body that issued the certificate. There is no single system today to verify the authenticity of a certificate.

The term blockchain has gained much popularity the last few year. By looking at Google Trends [2] it can be seen that the interest in the term "blockchain" stayed fairly low until 2015 where it started growing. It reached its all-time high, measured in number of searches, in December 2017. The blockchain technology was first presented by Satoshi Nakamoto in 2008 as the basis for the electronic cash system Bitcoin [3] and is a way of structuring data storage.

Bitcoin and other digital assets based on the blockchain technology are called cryptocurrencies. At the time of writing this thesis the combined value of these cryptocurrencies is just under 1 200 billions SEK [4]. This is equal to a quarter of the gross domestic product (GDP) of Sweden in 2016 [5].

It is perhaps this economic growth that has made people so interested in the blockchain technology. While some say that blockchain is the new internet [6] others say that blockchain should be treated with caution and that it is not a silver bullet that can be used to solve any problem [7]. This thesis will try to answer if blockchain is a solution that can be used to store digital certificates.

1.2 Problem statement

Today there exists many different standards and certification bodies. It is easy for a company to flash their certificate to a customer and require a higher price. Due to the many standards and certification bodies that exist today it may be difficult for the buyer to verify the certificate. Both digital and physical certificates can be forged. Online records may not be up to date or have missing information.

The blockchain technology might be able to solve these problems by providing a new way to store digital certificates.

1.3 Purpose

The purpose of this thesis is to investigate what the blockchain technology is and how it can be used in the certification industry. The author will investigate how certificates are used today, specifically how they are stored and accessed.

An application based on the blockchain technology is designed and developed. The purpose of the development is to put the strengths and weaknesses of blockchain in a practical context. This application is evaluated against the findings of the current situation of storing and using certificates.

The thesis is done in an exploratory way.

1.4 Research questions

This thesis aims to answer the following research questions (RQ) in order to fulfil the purpose.

(RQ1) How are certificates stored and accessed today?

(RQ2) How can the blockchain technology be used to design and develop a system for storing digital certificates?

1.5 Delimitations

In this thesis there are some delimitations. Many of these delimitations have evolved over time due to the exploratory approach of the thesis. At first, it was decided that only actors and certificates in Sweden would be of interest. In addition to this there have been some delimitations on what types of blockchains are looked at. Since the term blockchain is very broad not all aspects could be included. For instance only permissionless blockchains have been included, and Hyperledger Sawtooth is the only platform for building the application that has been used.

2 Methodology

This chapter presents and motivates the research strategy and research method chosen for this thesis. The chosen strategy is an exploratory approach and the method is a case study and prototype development.

2.1 Research strategy

According to Höst et al. [8] there are four main ways to describe the nature of the research being conducted. These are descriptive, exploratory, explanatory and problem solving studies. Not all research is done solely in one way, and may mix multiple of the strategies mentioned.

In this thesis an exploratory approach has been chosen, which aims to understand and explain something on a deep level. The reasoning for this approach is the juvenile nature of the blockchain technology. There is not much research that has been done in this area, meaning that an exploratory approach can be meaningful and discover new things.

The exploratory research strategy also means that purpose, scope and methodology of the thesis has changed throughout the process.

2.2 Research method

Two methods have been chosen for this thesis. The first method is a case study of different actors using certificates today. The second method is the development of a prototype application using blockchain to store digital certificates.

The case study includes a farmer who produces and sells KRAV-certified rapeseed, two third party certification bodies and a website that collects and displays digital certificates.

The main method of collecting data for the case study has been interviews with employees at the subjects of the case study mentioned above. The data collected is qualitative. The motivation for this is the exploratory nature of the work done for this thesis, which according to Höst et al. [8] makes interviews a good choice of data

collection. The interviews have been conducted in a semi-structured way. This allows the interviewee to freely talk about the subject at hand, while the interviewer still leads the conversation when needed.

Since the goal of the interviews were to collect qualitative data the focus when selecting interviewees were not to represent the entire population of this industry. The interviewees were instead selected with the stratification of the industry in mind. Since the interviewees were not selected randomly from the population, no general conclusions have been made regarding the industry. [8]

The purpose of the prototype development is to explore if this solution is possible and if it is a good choice. Characteristic for prototype development in a thesis is that the specifications are not determined when the work starts [8]. These are instead developed alongside the application itself throughout the process. This iterative development process suits the exploratory nature of the thesis.

In addition to this a literature study has been performed to construct the theory section of the thesis. This section is necessary to fully understand the blockchain technology. The literature study was done using LUBsearch, Lund University's search engine, and Google Scholar. The reference section of other master theses were also used.

3 Theory

This chapter gives the reader a theoretical ground for the following parts of the thesis. The chapter contains four sections. The first section presents information on certificates and related terms. In the second section the reader is given knowledge on cryptography needed to understand the third section, which presents the blockchain technology. The last section handles the Hyperledger Sawtooth project which was used for development.

3.1 Standards, certification and accreditation

3.1.1 **Standards**

A standard is a set of requirements on a product, system or organisation. A simple example of a standard is the one of electrical outlets. Without this our everyday life would be much more complicated. The standard allows us to use our electrical devices at our friend's house without any trouble. The standard also lets most Europeans use outlets in other countries without extra thought. Of course, if the same European travels to America they will however have trouble using their device without a converter since there is another standard there.

Another standard is the ISO 14000 family of standards. These are standards in the area of environmental management created by The International Organization for Standardization (ISO) [9]. These standards include many requirements for how a company or organisation should work to ensure that they manage their environmental responsibilities.

3.1.2 **Certification**

ISO defines certification as:

> *"the provision by an independent body of written assurance (a certificate) that the product, service or system in question meets specific requirements."[10]*

The process of certifying something is the same as saying that the product or system under inspection follows some requirements, most often a standard. How the process is structured depends on many factors. Mainly these factors are the standard that is being certified and the certification body.

A certification results in a certificate, which also can be very different depending on the standards and the issuing certification body.

Third party certification means that there is another, preferably independent, party that performs the certification [10]. A first party certification is when the owner of the product that assurers that they follow a set of requirements or a standard. In this thesis the phrase certification will always mean a third party certification, unless specified otherwise.

3.1.3 **Accreditation**

Accreditation is the act of giving a company or organisation credibility when it comes to control and certification. The accredited company or organisation has to have a certain level of competence and systems in place to perform their work. This gives them an international approval as an unbiased actor. [11]

SWEDAC is the Swedish national accreditation body. They have been given this task by the Swedish government [12]. This means that they have the right to determine which companies and organisations in Sweden that are accredited.

Depending on the field it may be a requirement for certification bodies to be accredited. In these cases only accredited certification bodies may issue a certificate [11]. In fields where it is not a necessity there still are accredited certification bodies. The accreditation gives them extra credibility compared to companies that are not accredited.

3.1.4 **The relationship between the terms**

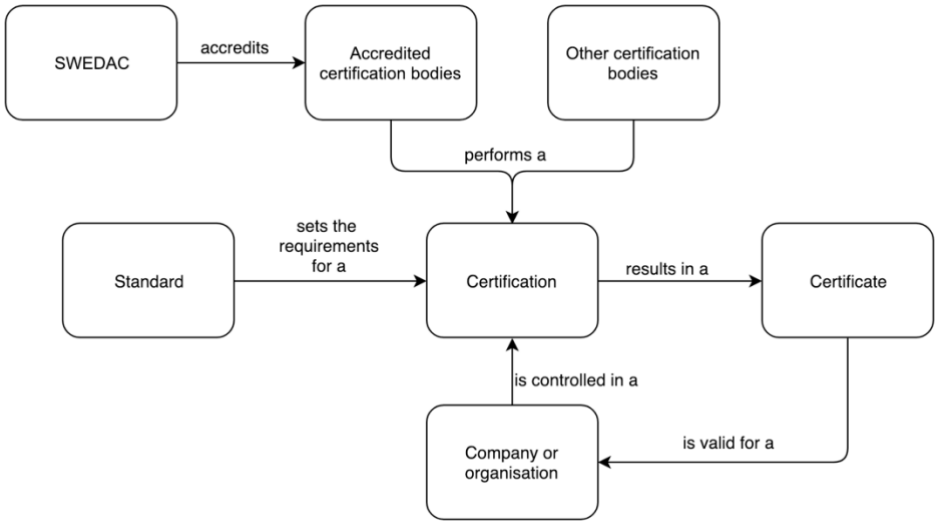

Figure 1. The relationship between the terms standard, certification and accreditation. This figure was made by the author of this thesis.

Figure 1 shows how the different terms mentioned above are related. A certification is a process that takes three inputs.

- A company or organisation that wishes to be certified
- A standard to be certified against
- A certification body that performs the certification. These may be accredited.

The certification body inspects the company or organisation and determines if they fulfil the requirements of the standard or not. A certificate is the result of a successful certification.

Note that a specific product can be certified for a standard instead of the company or organisation being certified. For this thesis the focus will lie on certifications for companies and organisations.

3.2 Cryptography

Knowledge within cryptography is needed to fully understand why the blockchain technology is secure. Readers that are not interested in this may wish to briefly read this section and refer to it when needed. Note that the following section does not

fully go into all the mathematical details and proofs of the concepts, since it is not the focus of this thesis.

3.2.1 **Hash functions**

A hash function is a function that takes a string of any length as input and produces a string of a fixed length. If the hash function *H* is applied to the message *m*, the resulting hash value *h* is produced. A hash function is deterministic and requires little computational power. [13]

$$H(m) = h \tag{3.1}$$

Figure 2. Two messages and their corresponding SHA-256 hash value. This figure was made by the author of this thesis.

Figure 2 shows how two messages of different lengths give a hash value of the same length when used as input for the SHA-256 hash function.

A subset of hash functions are cryptographic hash functions. According to Smart [13] cryptographic hash functions should be:

- Preimage resistant
- Collision resistant
- Second preimage resistant

Being preimage resistant can be translated to being one-way functions. This means that for the hash function *H* it is hard to produce the original message *m* given only the resulting hash *h*. [13]

Collision resistance means that it is hard to find two different messages that have the same resulting hash value [13]. In other words it should be infeasible to find *m* and *m'* such that

$$H(m) = H(m') \text{ where } m \neq m' \tag{3.2}$$

The third property, second preimage resistance, relates to both other properties. Cryptographic hash functions should be constructed in such a way that given a message *m* it should be hard to find another message *m'* that has the same resulting hash value as the first message [13].

The difference between the second and the third property is mostly the situation they protect against. Collision resistance protects against two arbitrary messages getting the same hash value by accident. Second preimage resistance is protection a malevolent user that tries to take on the identity of another user.

The properties of cryptographic hash functions are said to make a certain action hard or infeasible. In mathematical terms this means that it cannot be done in polynomial time [14].

In most cases cryptographic hash functions are meant when talking about hash functions. This is also the case in this thesis and the reader should remember the three properties of cryptographic hash functions when reading about hash functions further on.

3.2.2 **Public key cryptography**

In the following two sections the persons Alice, Bob and Eve are used. The examples are based on those presented by Smart [13]. Alice and Bob wish to communicate over a public channel without anyone else being able to read their messages. Eve represents another user who has access to the encrypted messages and wishes to read them.

One of the simplest ways for Alice and Bob to encrypt their messages is through symmetrical encryption. Alice wishes to send a message to Bob and encrypts the message which she sends over the public channel. Alice and Bob share a secret key that Bob can use to decrypt the message. Eve can only see the encrypted message and are therefore unable to read the message. This approach has two main problems. The first one is that Alice and Bob has to share the secret key in a secure channel that makes it impossible for Eve to also receive this secret key. The second problem is the addition of new people in the communication. If Alice wants to send a message to a third person, Charlie, she either has to share a new secret key with Charlie or use the same one as the one she and Bob are using which would reduce the security of that key.

The solution to the problems with symmetrical encryption are solved by assigning two keys to each user, one public key and one secret key. This is called asymmetrical encryption. The public key is as suggested by its name public and available to everyone, while the secret key is only available to the owner. Note that the keys come in pairs, i.e. each public key has a specific private key and vice versa.

If Alice now wishes to send a message to Bob she encrypts her message using Bob's public key. The message is sent over the public channel, but it can only be decrypted using Bob's private key. This means that Eve is still unable to read the message. If Bob wishes to respond to Alice's message he encrypts his answer using her public key.

3.2.3 **Digital signature**

Using asymmetrical encryption it is possible to share encrypted messages without having to share a common secret key. While Eve cannot read the messages she is still able to cause damage. She can simply send a message to Bob, since his public key is available to her, and pretend that the sender is Alice. Bob has no way of knowing if Alice truly wrote the message. The solution to this problem is the introduction of digital signatures.

The digital signature protocol is applied with the addition of two functions. The first function takes a message and a private key, and returns a signature. The second function takes a message, a signature and a public key, and returns a true or false.

Alice wants to send a message to Bob and wants to make sure Bob knows she sent it. She takes her message *m* and her private key *sk* to produce a signature *s*. The message and the signature are sent over a public channel to Bob. Bob then takes the message *m*, the signature *s* and Alice's public key *pk* to verify that Alice is the sender of the message.

Note that in this example the message itself is not confidential and Alice's only interest is to prove that she is the sender of the message. Digital signatures can however be used together with the public key encryption to both encrypt the message and to prove who the sender of the message is.

3.2.4 **Nonce**

A nonce is a number that is only used once. These numbers are often random. Nonce number are used to alter the hash value of a message without changing the message itself. They are often appended to the message when computing the hash value but discarded by the receiver of the message. [13]

3.2.5 **Secure hash algorithms**

Secure hash algorithms (SHA) is a family of hash functions. They are all iterative in their nature and fulfil the requirements of cryptographic hash functions [13]. Two of these will be mentioned and used further on in the thesis.

- SHA-256: Uses 64 single step iterations to produce a hash value with a length of 256 bits.
- SHA-512: Uses 80 single step iterations to produce a hash value with a length of 512 bits.

3.3 The blockchain technology

There are multiple definitions available for what a blockchain is. In the following section the origin of the technology and some definitions of it will be presented. Note that blockchains are also called distributed ledgers.

3.3.1 **Origin**

In 2008 the white paper for Bitcoin was released under the alias Satoshi Nakamoto [3]. Nakamoto presents Bitcoin, a system for electronic payments in a peer-to-peer system that does not require users to have trust in any other party. At the time, other systems for making secure electronic payments in a peer-to-peer system existed, but none of them had solved the problem of double spending without a trusted third party. The double spending problem means that one user can make multiple payments with the same asset. With a trusted third party involved the solution is simple, the payee simply asks them if the asset has been spent before. [3]

Nakamoto solved this problem by constructing what he called a "block chain". Today the term blockchain is more common. The term comes from the data being stored in blocks, where each block contains the hash value of the previous block, and therefore forming a chain.

3.3.1.1 Bitcoin

A distributed timestamp server using proof-of-work is proposed in the white paper to be used to reach a consensus on which blocks are valid. The idea behind proof-of-work is to let the majority of nodes, determined by contributed CPU power get to decide which the next block in the chain is. This system can be seen as a lottery where each nodes receives a larger chance to win based on their contributed CPU power. The winner of the lottery gets to decide what transactions should be included in the next block. The proof-of-work used in Bitcoin is explained in section 3.3.3.2. [3]

Nodes show that they accept a certain chain of blocks by using the latest block in that chain as their previous block when computing the hash of a new block. Honest nodes will see the longest chain as the correct one, but it is possible for malevolent nodes to try to build from some earlier node. [3]

Nakamoto states that the first transaction of each block is a transaction that creates new coins that are awarded to the miner of the block. Together with transaction fees this gives miners an incentive to spend the CPU power needed for the proof-of-work. The transaction fee is determined when a user broadcasts their new transaction. They specify an input value (from an address where they own enough assets) and an output value to some other address. If the output value is lower than the input value, the difference is received by the miner as an extra incentive to include that specific transaction in their next block. [3]

3.3.2 **Definitions**

In this section some definitions of blockchain will be presented. The author finishes with his own definition of the term.

Narayanan et al. [15] describe a block chain as a data structure which consists of a linked list with hash pointers. Each element in the list is a block containing some data and the hash of the previous block. This makes it a tamper-evident log, meaning that it only is possible to append data to the list and impossible to alter previous data without detection.

Iansiti and Lakhani [6] define a blockchain with the following sentence:

> *"... blockchain is an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way."*

They [6] also provide five basic properties of the blockchain technology:

1. Distributed database. Each participant in the network can access the entire database and its history. They are able to verify all entries themselves.
2. Peer-to-peer transmission. Each transmission is done between two single nodes. There are no central, intermediary nodes.
3. Transparency with pseudonymity. All transactions are transparent and visible to all participants in the network. Users are identified by addresses and they decide if they want to disclose their real life identity linked to this address.
4. Irreversibility of records. Once a transaction is added to the database, it is impossible to alter that transaction. This is due to the chain structure of the blockchain.
5. Computational logic. Transactions can be programmed to automatically occur according to algorithms and rules.

Hyperledger Sawtooth [16], the open source project used to build the application developed in this thesis, defines a blockchain as

> *"... a database (a ledger) of transactions to all participants in a network (also called "peers" or "nodes"). There is no central administrator or centralised data storage."*

They [16] give the blockchain three properties:

1. Distributed. The database is shared between all participants, that may or may not be trusted. All participants in the network have the same information.
2. Immutable. It is not possible to alter the history of transactions. The immutability is ensured by the block hashes.
3. Secure. All transactions (changes) are signed by known identities.

The author of this thesis summarizes the definitions above with his own definition.

> *"A blockchain is a way to store data. The data is stored amongst multiple nodes and there is no owner of the system. All entries in a blockchain is dependent on older entries through hash values."*

In addition to this a blockchain should have the following three characteristics.

1. Transparency. All data in the blockchain, and the structure of the blockchain, is available to all users.
2. Security. The blockchain is secure against tampering and the data is immutable.
3. Decentralization. There is no owner of the blockchain. A blockchain is simply a set of users that agree to run the same system.

3.3.3 **Consensus**

In distributed ledgers, such as blockchains, there has to be some way that the different nodes reach a consensus on what data should be valid. There are multiple ways to design this protocol.

3.3.3.1 The Byzantine generals problem

The Byzantine generals problem is relevant to blockchains. The problem is an analogy for computer nodes trying to reach a consensus, where there may be nodes that are either malevolent or faulty.

The analogy consists of multiple Byzantine generals surrounding an enemy city. They each control a small portion of the army and want to decide on a strategy. In its simplest form the strategy can be to attack the city or to retreat. The generals wish to reach a consensus on which strategy to use. This can be done by letting all generals vote and use the majority vote as the chosen strategy. However, some of

the generals are traitors and wish to affect some generals in such a way that they take another action than the majority decision. [17]

Lamport et al. [17] present algorithms that let the generals reach a consensus under different circumstances.

They show that with oral messages, more than two thirds of the general have to be honest generals for them to be able to reach the true consensus. An oral message can be seen as a messenger being sent, where a traitor can potentially intercept and replace this messenger with one of their own. [17]

In a system where the generals are able to send written, unforgeable messages the authors show that the problem is solvable for any number of generals and traitors. This solution will however be very expensive, due to the large amount of messages necessary. [17]

The Byzantine generals problem can be applied to blockchains by seeing each node in the network as a general. The nodes want to reach consensus on which blocks are valid. Just as in the original problem, some nodes may want to confuse the other nodes for their personal winning. The written, unforgeable messages can be achieved with the public key cryptography system and digital signatures.

3.3.3.2 Proof-of-work

A proof-of-work (PoW) is a mechanism for a user to show that they have invested some kind of computational resource, often CPU power. In blockchains PoW is used to give nodes voting power according to the amount of resources they contribute with. The reason that this is needed is because if it was not included, a single user could simply connect a large number of nodes and get more voting power than others. With the PoW system this single user will have to invest more resources to get more power.

By using cryptographic hash functions it is possible to construct a PoW that has some desirable attributes. The task is to compute a hash value that starts with a specific number of zeroes. The hash value is computed from a specific message and an added nonce number using a specific hash function [14]. The following examples shows how this would look in practice.

The message is “I want to show that I have invested some computational power” with a nonce appended at the end. The message should be hashed using SHA-256. In a simple implementation of this the program starts with a nonce of one and increments this by one until the correct hash value is found. Depending on how many leading zeros that are required, it will require more CPU power and time to find the correct nonce. Table 1 shows the results for up to five leading zeroes.

Table 1. The nonce number required depending on how many leading zeroes the hash value should contain.

Number of leading zeroes required	*Nonce number*	*First ten characters of resulting hash*
1	23	0ABE297B79
2	72	00AC234A6A
3	2347	00010C9BAA
4	138326	00009688E5E
5	345147	000003EAEA

For instance in the last example the message “I want to show that I have invested some computational power345147” encoded with SHA-256 will give a hash value with five leading zeroes. This means that over 300 000 values had to be tried before the correct nonce could be found. Note that there may be other nonce numbers that give a hash value with five leading zeroes.

In other implementations a random number may be used instead of one that counts up from one. This is because every nonce number has the same probability to give a satisfactory hash value. The “counting” nonce was used in this example to show that by increasing the number of leading zeroes required it takes a longer time to find the correct nonce.

According to Judmayer et al. [14] a PoW used for blockchains should have the following characteristics:

1. Any given PoW is easy to verify.
2. The PoW is difficult to generate.
3. The difficulty of the PoW is parametrizable (sic).
4. It should not be possible to reuse previously generated PoWs.
5. It should not be possible to generate PoWs ahead of time and use them later.

The following paragraphs describe why the system with hash value and leading zeroes, described by the example above, has these five characteristics.

The first characteristic is fulfilled due to hash functions being easy to compute. To verify that the nonce 345147 is the correct value in the case above, one single hash value has to be calculated. This hash value has to be checked for a certain number of leading zeroes. This is very computationally easy.

The second characteristic comes from the preimage resistance property of cryptographic hash functions. Recall that this means that the function is a one-way function only. There is no way to construct the nonce from the desired hash value. The only way to find it is to try different values until a match is found.

By changing the number of leading zeroes desired in the hash value the third characteristic is fulfilled. Increasing the number increases the difficulty of the PoW. [14]

The fourth characteristic requires an addition to the system mentioned in the example. By adding a time-stamp to the message it is not possible to reuse an old nonce number. This also enables the fifth characteristic. Judmayer et al. [14] mention that it actually would be possible to generate PoWs ahead of time if one would know the exact message and time this will be sent.

3.3.3.3 Proof of elapsed time

Proof of elapsed time (PoET) is a consensus method developed to be more efficient than PoW. PoET can be seen as a function that makes a node wait a random amount of time. The function for generating the amount of time a node should wait is run in a "trusted execution environment". This allows the mechanism to detect any users that are trying to make an action before their random time has elapsed. [18]

In practice this mean that the PoET has the same characteristics as PoW but without the need for nodes to actually invest the computing power to find the nonce number. The amount of voting power of each user is limited by the number of processors a user can contribute with. [18]

3.3.4 **Adoption of the blockchain technology**

Iansiti and Lakhani [6] make many comparisons between the blockchain technology and the Transmission Control Protocol/Internet Protocol (TCP/IP) technology. The latter is the base on which the internet is built upon. TCP/IP started out small when it was presented in 1972 and has since then slowly grown to what it is today, with many different applications built on top. The authors state that both TCP/IP and blockchain are "foundation technologies" and not "disruptive technology". They therefore predict that the blockchain technology will go through phases similar to those of TCP/IP before becoming a foundation in different areas. They state that "While the journey will take years, it's not too early for businesses to start planning". The authors present a framework to evaluate different applications in regards to blockchain adoption.

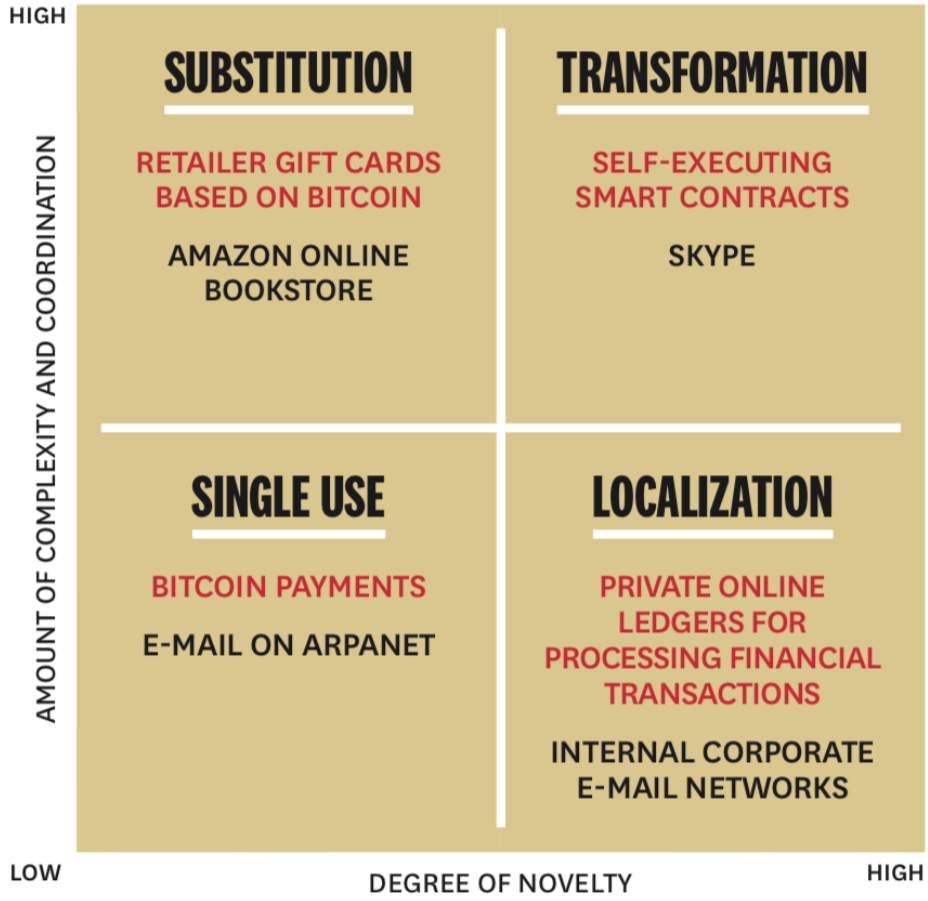

Figure 3. The different types of blockchain applications. [6]

According to Iansiti and Lakhani [6] there are two dimensions that can be used to describe the use cases of foundation technologies. Novelty describes to which extent the use case is new to the users. The other dimension says how much coordination and how complex the use case is. In general this relates to how many users that are needed within the new application to make it worthwhile for the users.

3.3.4.1 Single use

Applications in this section do not solve a new problem and are fairly simple to implement and start using. This is where the first applications using the new technology are placed, which for blockchain is Bitcoin payments. They offer immediate value to the users and do not require a large user base. [6]

3.3.4.2 Localization

The use cases that are high in novelty but still low in complexity and coordination are placed within the localization area. They solve a new problem but do not require a large user base. According to Iansiti and Lakhani [6] these kind of use cases are applications very much like those in the single use area but are instead used in a private network between organisations that trust each other and are willing to share a distributed ledger. The authors give examples of how this is being used in the financial sector. [6]

3.3.4.3 Substitution

Applications that aim to substitute existing systems are low in terms of novelty but require a high amount of coordination. These use cases are applicable in areas where the blockchain technology can outperform current technologies. In most cases it does however require a complete reconstruction of the existing infrastructure which leads to high barriers of entry. The reconstruction is required since the new solution will only work if everyone uses it. [6]

3.3.4.4 Transformation

The final sector in the framework contains application that have the potential to make big changes to existing systems. They both require coordination from many different actors while at the same time requiring users to completely change their behaviour. Smart contracts in the blockchain are able to live up to these promises. For example this could lead to automation by introducing self-executing contracts in banking. One of the main challenges identified by the authors is how the contracts should be designed, verified, implemented and enforced. [6]

3.4 Hyperledger Sawtooth

Hyperledger is an open source project launched in 2016 aiming to advance the blockchain technology in multiple industries. It is hosted by The Linux Foundation who collaborate with world leaders in finance, banking, Internet of Things, supply chains and manufacturing. [19]

The Hyperledger project compares blockchain technology to the Web itself, and believe that it shows much potential. The main aspects of the blockchain that are believed to have this potential is the peer-to-peer distributed ledger forged by consensus and smart contracts. The goal is to build transactional applications that have trust, accountability and transparency. [19]

Hyperledger Sawtooth is one of the projects hosted under the Hyperledger umbrella. It is a platform for building blockchain applications on an enterprise level. The two

main design goals are to keep the ledger distributed and to make the smart contracts safe. [16]

Hyperledger Sawtooth uses a modular approach that separates different parts of the system. This means that the blockchain level is decoupled from the application level. The modular design also means that different components of the blockchain can be changed, depending on the need of the project. Examples of the components that can be configured are transaction rules, permissioning and consensus algorithm. [16]

3.4.1 Global State

A distributed ledger, or a database, has a global state. The global state is all the information and current status of everything stored in the blockchain. The information included in the global state is very different depending on the usage of blockchain. [20]

3.4.2 Transactions and Batches

The transactions are the actions of the users, also known as nodes, in the blockchain. The purpose of the transactions is to modify the global state of the ledger. Just as the global state, these differ depending on the application.

A transaction consists of the transaction header, and its corresponding digital signature, and the payload of the transaction. The header contains metadata on the transaction; such as who performed the transaction, which type of transaction it is and any other transactions in the same batch that must be handled before this transaction. [21]

In Hyperledger Sawtooth, and many other implementations of blockchains, the transactions are put into batches. Batches are used when the order of transactions is important. By putting these transactions into the same batch, the transactions will be handled in the correct order. If a transaction does not depend on any other transaction, except those already validated and stored in the blockchain, the sender can create a single batch with only that transaction. [21]

3.4.3 Transaction families

Hyperledger Sawtooth uses transaction families to define what transactions are possible to perform. The transaction header contains information on which transaction family a transaction belongs to. The transaction family is the system which handles the payload of the transaction. It is here that the business logic can be defined, ensuring that the transactions affect the state in the correct way. A

transaction family has a version number, ensuring that transactions towards an older version of the code still can be handled. [22]

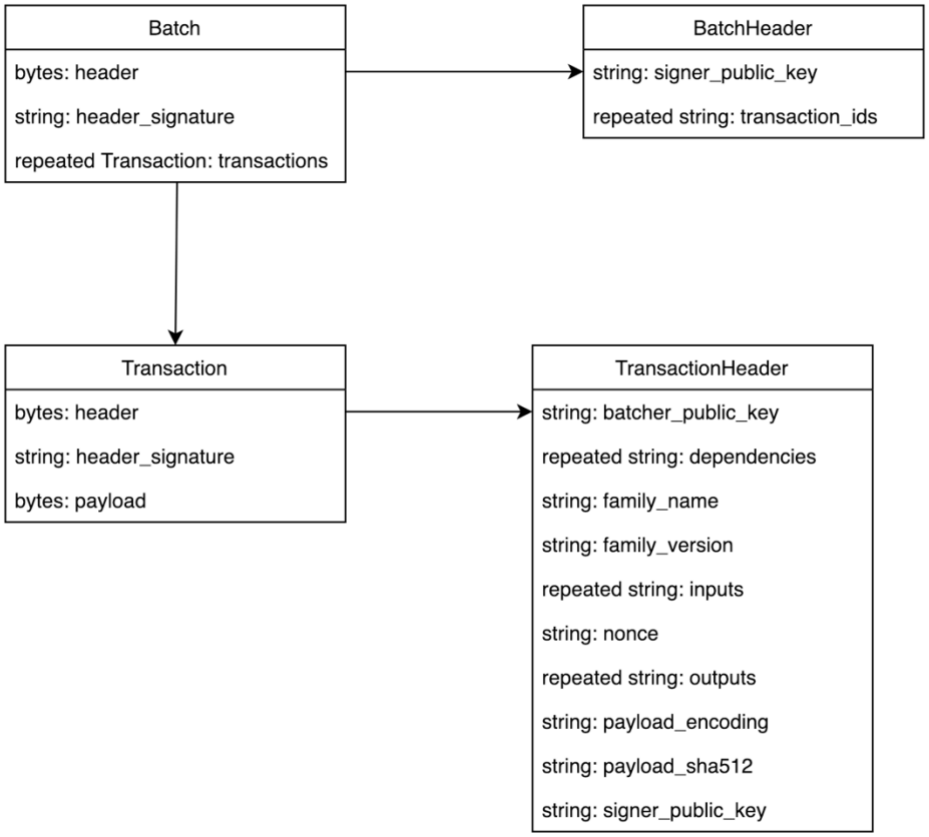

Figure 4. The relationship between the Batch, BatchHeader, Transaction and TransactionHeader classes. [21]

Figure 4 shows how transactions and batches are related. The word before each attribute states the type of the attribute, for instance string or bytes. Repeated means that the class may contain multiple instances of this attribute, e.g. a batch can contain multiple transactions.

Both the Transaction class and the Batch class has a header. The headers contain meta information about the transaction or batch.

The Batch class contains a serialized version of the BatchHeader, a signature of the header and the references to the transactions included in the batch. The BatchHeader class contains information on who signed the batch and the IDs and the transactions included in the batch. [21]

The Transaction class also contains a serialized version of its header, a signature of the header and the payload of the transaction. The payload is perhaps the most interesting part of the transaction, as this is the actual information on what the sender wants the transaction to do. [21]

The TransactionHeader class contains lots of information. The public key of the batcher must match the key that was used to sign the batch which this transaction is included. The transaction header also has information on what transaction family, and version, that should handle the payload of the transaction. The dependencies, inputs and outputs attributes are used when different transaction has to be handled in a specific order. The payload_sha512 is a SHA-512 hash of the payload. This can be used to verify that the payload was transferred correctly. The nonce attribute is used to separate two otherwise identical transactions. [21]

3.4.4 **An example with cryptocurrencies**

These terms can be explained with an example of cryptocurrencies. In a cryptocurrency built on Hyperledger Sawtooth the global state would be the amount of money each individual owns. The individuals have a unique address, and each of these addresses has a certain amount of assets.

A transaction is the addition or removal of assets from one of the addresses. The most common transaction would probably be the transfer of an amount of assets from one address to another. Another example of a transaction is the creation of new assets.

The transaction family handles the transaction. It is here that it is made sure that the transaction actually is valid and corresponds to the rule set of the cryptocurrency. The sender of the transaction has to be the user with ownership of the address from where the assets are taken. The transaction family might also ensure that there are sufficient funds on the address.

4 Case study

This chapter presents the case study done in this thesis. It presents four different actors that all are working in fields related to certificates.

4.1 The case

The case is a study of four different actors that in some way work with certifications. The first actor is a farmer that is KRAV-certified and wants to show this to the buyers of his products. The second actor is a company that performs certifications. The third actor is a research institute with a section which is very similar to the second actor, but they are not their own company. The fourth actor is a website that shows digital certificates from different certification bodies. The website does not perform any certifications themselves.

The purpose of the case study is to answer RQ1, "*How are certificates stored and accessed today?".* The actors have been chosen with the hopes of getting different perspectives on how the certificates are used today.

4.2 The farmer

The farmer is the owner of a farm that only produces ecological products. Amongst products such as ecological fertiliser they grow rapeseed. The rapeseed is grown on approximately one fifth of the total area used for farming. The farm is KRAV-certified for all their products and has been since 2005. [23]

The farm has a physical copy of their active KRAV-certificate. This is given to them by the certification body and is valid for one year. The certificate states which products the farm is certified for, which in their case is all of their products. The physical certificate plays an important role when the farm sells their crops. A physical copy has to be present throughout the transportation of the crops to show that the product is KRAV-certified. [23]

In the case of rapeseed the farmer has a buyer that buys most of what the farm produces. This buyer sells the rapeseed to Germany where it is used for lotions.

When a truck arrives at the farm to pick up a delivery the farmer has made a physical copy of the KRAV-certificate. The driver of the truck verifies that the certificate contains the correct information, for instance that it has not expired and that rapeseed is present under the list of certified products. The certificate is then kept with the load and must be shown when the truck arrives at its destination. The farm is not aware of anyone verifying the validity of the certificate and says that people trust the physical copy. [23]

The farm does not buy any products that are used in the production of their KRAV-certified products. However, it happens that they store crops for other nearby farms. The reason for this is that there are farms that are certified for production but not for storage. Due to the rules of certification the farmer must ensure that the products he stores with his own crops also are KRAV-certified. If they are not KRAV-certified he has to clean his storage area before he can store his own products there. If he fails to do this he risks losing his own certificate. The process is similar to that when he is selling rapeseed. The other farmer, who wants to store their crops at the interviewee's farm, brings a physical copy of their certificate together with their crops. The farmer checks that crops being stored are present in the certificate and that all other information is correct. He only checks the physical certificate. [23]

The farmer feels that he can trust the physical certificates. He says that even though he can check if a certain farm or company is certified online, he often needs more information than that which is available. This information can be things such as which products are included in the certificate and can only be found on the physical copy. He has a positive attitude towards a more modern system with online storage and verification of certificates. He says it would be possible for someone with the right knowledge to create a false physical certificate, although he has never experienced it. [23]

The farm has been certified by the same certification body since they first were KRAV certified in 2005. At the time they were the only option due to legislation. They have been satisfied with their service and has not seen any reason to switch. The person performing the inspection has changed during the years and the farmer says that some are more thorough than others. He wishes that all the inspections were thorough since he has put a lot of time and effort to fulfil the requirements. [23]

4.3 The certification body

The certification body is one of the largest Swedish certification bodies and have grown during the last years. They certify for standards ISO 9001, ISO 14001 and many more. They currently have more than 1000 active certificates that have been issued by them. They are accredited by SWEDAC. [24]

On the company's website it is possible to see the companies that currently have active certificates. Under each company the name of the standard they are certified under are listed. There is a separate list for recalled certificates. They also refer the viewer to an external website to see their active certificates. [24]

The company store their certificates in their own database. This database has a connection to the external website that once a day keeps the records synchronised. The reason that they are connected to the external website is that many other companies accredited by SWEDAC also have this connection. The company also issue a certificate as either a physical copy or a digital file to the customer that has been certified. [24]

Even though all their certificates are kept in an updated version on both their own website and the external website there are still companies or organisations that contact the certification body directly to ask about the validity of a certificate. The reason in these cases are according to the employee that the person wants a verification that the information is correct. [24]

All the information contained in the certificates issued by this company is public. This means that as soon the certificate has been issued anyone can access the information. The interviewee mentions that this is one of the basic ideas of the certificates they issue. The company being certified want to share that they have been certified. [24]

The information in each certificate may differ from different companies and certificates. In some cases the certificate is only valid for one part of the company, for instance the production. Common for all certificates is that they include the name of the standard, a start date and a valid duration. [24]

4.4 The research institute

The research institute has over 2000 employees. The institute is owned by the Swedish state. Amongst other areas they perform certifications and issue certificates. The certification section of the institute has approximately 80 employees. They are accredited by SWEDAC. [25]

On their website it is possible to search for certified products, persons and management systems. It is only certified products and persons that can be found on their own website. Information about the management system that they have certified can be found on an external website that the user is redirected to. [25]

After the research institute has issued a certificate they both send a certificate to the certified company and store it in their own system. The certificate that is sent to the company can either be physical copy or a digital file. The digital file contains a digital signature to show that the certificate is real. [25]

Every now and then the research institute is contacted by someone who wants to verify the authenticity of a certificate. The most common situation is that they are unable to verify the certificate by themselves online. The interviewee thinks this might be due to them not finding the information or that they do not try to find it. However, it also happens that someone has found the certificate online and wants to ensure that it actually is valid by contacting the research institute. [25]

4.5 The certificate website

The certificate website is the external website where both the certification body and the research institute refer their viewers to see their active certificates. The website has connections to other certification bodies as well. They only show certificates from accredited certification bodies. [26]

The website has been active since 2004 and was an effort by the Swedish association for testing, inspection and certification (SWETIC) to remove unserious websites for finding online information about certificates. SWETIC is an association for Swedish certification bodies. [26]

They have an automatic connection to all their connected partners that once a day retrieves the latest information on their certificates. The certificates are shown on the website and the user can search for a specific company to see if they are certified. If the company is certified by a company that is not connected to the website they will not appear in the search. [26]

The reason that some certification bodies have chosen to not be connected to the website is according to the website themselves competition. They say that some companies are afraid that other certification bodies will take their customers. In some cases the companies are active in countries where legislation make it hard or impossible for them to connect their internal databases to the website. [26]

5 Building the application

In this chapter the author tries to answer RQ2. The chapter presents information on how the application was built for this thesis and the development process. The code for the application can be found in Appendix A.

5.1 Development process

The Hyperledger Sawtooth platform was selected as the basis for the application developed in this thesis. Two of the reasons for this is the modular approach which enables configuration and the existing open source projects to build upon.

The developer, and author, started with thoroughly reading the documentation provided by Hyperledger Sawtooth to gain knowledge regarding the platform. In addition to this some examples and tutorials were followed to setup the development environment.

The development process was performed in an iterative fashion. Iterative software development can contain many different steps. Figure 5 shows each iteration of the development in this thesis.

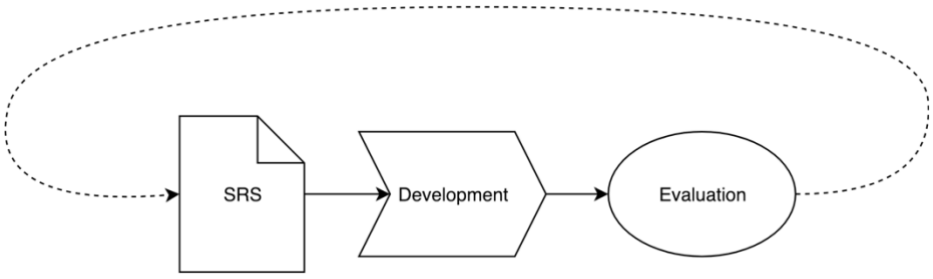

Figure 5. The development process in this thesis.

The first step in each iteration is to specify the requirements of the application. These requirements are written as a Software Requirement Specification (SRS). The development of the application is then focused to create an application that fulfil these requirements. At the end of each iteration the application is evaluated. The

result of this evaluation is an updated version of the SRS. Each revision of the SRS can both add and remove requirements.

The development was done in parallel to the case study described in the previous chapter. In the first iterations the requirements were more loosely specified and served as guidance for setting up the groundwork of the application. In the later iterations the requirements were more tailored to the current situation and the developer used his findings from the case study when he specified the requirements.

5.1.1 Software Requirement Specification

This is the SRS of the final version of the application.

Certificates

Requirement 1.1 A certificate should include the name of the company that has been certified.

Requirement 1.2 A certificate should include the name of the standard for which it is issued.

Requirement 1.3 A certificate should include a date for when it was issued.

Requirement 1.4 A certificate should include a duration for the amount of days the certificate is valid.

Requirement 1.5 A certificate should include a field for a string of an arbitrary length, intended for extra information about the certificate.

Requirement 1.6 A certificate should include information on the identity of the user that added the certificate to the system.

Requirement 1.7 The system should be constructed in such a way that it is possible to change what attributes a certificate should include.

Data storage

Requirement 2.1 It should be possible to add a certificate to the system.

Requirement 2.2 It should be possible to verify the identity of the user that added a specific certificate to the system.

Requirement 2.3 A user should be able to prove their identity by providing a public key corresponding to a private key that only they have access to.

Requirement 2.4 It should not be possible for a user to remove a certificate added by another user.

Requirement 2.5 It should be possible to see all certificates, and their history, added to the system.

Transparency

Requirement 3.1 All parts of the system should be transparent and visible to all users.

Requirement 3.2 It should be possible for a new user to join the system and add certificates under their identity without the approval of another user.

5.1.2 Development environment

The author's personal computer was used for the development of the application. The same computer was used to run the application for a development and testing environment. Docker was used to run the different components of Hyperledger Sawtooth. Figure 6 shows the components.

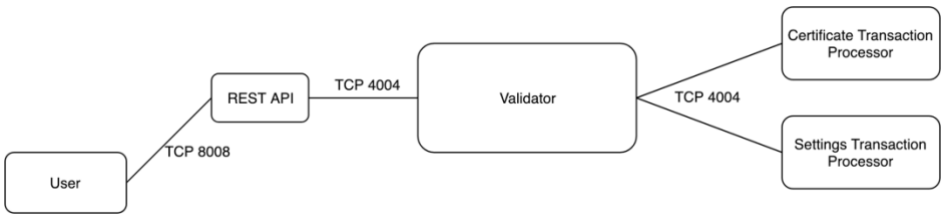

Figure 6. The different components of the development environment and the ports they use.

The validator is a single validator node that uses developer mode consensus. Developer mode consensus chooses a leader randomly instead of using another type of consensus, e.g. PoET. This is of course only meant to be used for development purposes where there are no malevolent nodes.

There are two transaction processors connected to the validator. The transaction processors are related to the transaction families. Depending on the transaction family specified in the transaction, the validator will let the corresponding transaction processor handle the payload of the transaction.

The certificate transaction processor is the one that handles transaction related to adding a new certificate. The setting transaction processor is provided, and required, by Hyperledger Sawtooth to allow the nodes in the network to coordinate their settings.

The REST API lets the user communicate with the application in a more flexible way. Independent applications in forms of website, mobile applications or integration with Internet of Things (IoT) can be developed against this API.

5.1.3 Evaluation

The evaluation process at the end of each iteration was done with the goal of seeing if the requirements from the SRS were achieved. In the case of a requirement not

being fulfilled one of two things happened. If the developer thought the reason to be a small bug or not require lots of time to fix the code was fixed right away. If it was estimated to take a longer time the problem was left until the next iteration.

The evaluation was done through manually testing, i.e. the developer trying out the application in a manual way without running any automated scripts or programs. The reason for this is the explorative nature of the thesis. Because of this approach the developer did not want to spend time writing test programs that might be obsolete in the next iteration due to the scope of the application changing.

5.1.4 **Running the application as a real network**

In this thesis the application has only been run with a single validator, hosted on the developer's computer. This is obviously not how the system would look if this was an application to be used for real certificates. The scope of this thesis does not include setting up a real network but it will be discussed in this section. The reason for this is to show that this is an application that actually could be used and that it is scalable.

The major change is that a network requires multiple users. A user is whoever is interested in contributing to the network, i.e. a certification body or creator of a standard.

Each user would have to run the following components [27]:

- At least one validator
- A set of transaction processors that matches that of the other users
- Optionally a REST API

These components can be run on a physical computer or in the cloud.

To start the network the first validator node will create a genesis node, the first block of the chain. All other blocks will be appended to this block. Note that the validator node that created this block has no extra authority, and is seen as similar to all other validators. [27]

In the network the PoET consensus should be used instead of developer mode consensus [27].

By setting up this kind of network the application can be used in the same way as it has been used for the development in this thesis. There is no owner of the network, there is only a set of nodes that all agree to run the same code.

6 Analysis

In this chapter the author tries to answer RQ1. This is done by analysing the current situation based on the case study from chapter 4. The author also discusses how the application presented in chapter 5 is suitable as a solution for the problems found in the current situation.

6.1 How certificates are stored and used today

The following section analyses how certificates are stored today based on the case study.

6.1.1 **The farmer**

The certificates used by the farmer are extremely tied to the physical world. His KRAV-certificate is issued to his farm and only makes sense in that context. He sells physical products and that is when he has to prove that he is certified. He keeps his certificate in a physical copy that he makes copies of. The certificate has to be kept with his products throughout the transport, even when he no longer has control over them. The reason for this is so the driver of the truck that transports the crops can prove that the crops are KRAV certified at a later stage in the supply chain.

When the farmer stores crops for other farms he is on the other side of the process. He has to verify that the crops he stores actually are KRAV certified. In a similar manner to when he is selling his products he checks a physical copy of the certificate.

The main reasons that the farmer uses physical certificates seems to be due to the rules. It is currently not possible to verify the certificates in an easy way online and they therefore use the physical certificates to prove the validity of the certificates. He does seem to be positive towards a more modern system, if it would exist.

6.1.2 **The third party certifiers**

The third party certifiers and the certificate website contains information on active certificates. This information can be accessed by anyone but is limited.

To get the full information on certificates someone has to have a copy of the certificate. The validity of these can be trusted in some way but not fully.

Both third party certifiers receive requests to verify certificates issued by them. This indicates that people do not trust the certificates all the way.

6.2 The problems with the current situation

In the case of the rapeseed farmer there is certainly room for improvement when it comes to how certificates are stored and used today. The current system with physical copies of the certificate does make sense in one way. The farmer is the owner of the certificate and since he only deals with physical products the physical aspect is not that big of a problem.

A potential scenario is that where a false certificate is created by a dishonest farmer. The farmer uses this to sell his crops at a higher price. When the product, or a product including crops from the dishonest farmer, reaches a consumer there is a problem. The consumer has paid a higher price for a KRAV-certified product which in reality might not fulfil the requirements that KRAV have created.

A similar scenario could happen when the farmer, or someone else along the supply chain, simply is ignorant instead of dishonest. They might look at a physical copy of certificate and think that is valid when it for instance might have expired or be a certificate for another product or company.

These problems could be solved without changing the current system. It is possible to check with the third party certifier if a certain certificate is valid. This is however time consuming and requires the buyer to be familiar with many different systems. In most cases it is also not possible to see the specifics of the certificates online which means that the buyer would have to call the certifiers directly to be absolutely sure that the certificate is valid.

6.3 Blockchain as a solution

The application developed for this thesis provides a solution built with blockchain to store certificates. The application is built with Hyperledger Sawtooth and is able to store multiple types of certificates. The application fulfils the three characteristics that, according to the author, defines a blockchain.

Transparency

The application provides a transparent system where anyone is able to see the certificates stored and their history. By providing an identity when a certificate is added it is possible to see who the certifier is. The REST API provided allows many different applications to connect to the application. A third party certifier can connect their internal databases to the application to allow automatic addition and updates of certificates. All code needed for the application can also be made public allowing anyone to inspect the system.

Security

The blockchain technology implemented in Hyperledger Sawtooth ensures that the application is secure. The chain dependency from new blocks to older blocks makes it impossible to alter old certificates without detection. The issuer of a certificate is still able to make updates by providing their identification key.

Decentralization

The blockchain technology allows the entire system to be decentralized. Although the system requires multiple nodes in the network there is no owner of the network.

By letting certifiers provide their identification key the issue of who to trust is moved from the application to the users. The application itself does not determine if a certificate is legitimate or not, it simply keeps a record of it and which entity that added it. A user is then able to check a certificate and match the identity of who added it to a third party certifier. If the user trusts the certifier they can also trust the certificate.

6.3.1 Trusting third party certifiers

Although the application that has been presented improves the current situation, it is not fool proof. The application relies on there being honest third party certifiers. If there exist certification bodies that issue certificates without doing a proper inspection the application would fail. This is not seen as a big problem as there has been no such indicators during the research of this thesis. If there would be, they would probably be detected by other actors in the market. These actors could then choose to not trust this certifier and therefore disregard certificates issued by them.

6.3.2 Could digitalization be enough?

One aspect of the application developed, and the blockchain technology in general, is that it digitalizes the current processes. This gives the user more information that is easily accessible. It would be possible to achieve many of the beneficial aspects

of this application without the use of application. The main benefit of the blockchain technology, according to the author, is the decentralized ownership.

With a blockchain application there is no need to trust one single organisation. Trusting a single organisation might be possible within one standard, e.g. everyone that trusts the KRAV-label should trust the KRAV organisation. By using a blockchain application it is possible for actors from many different industries to come together in one single system. This hopefully reduces costs and makes it easier to find different certificates.

6.4 The development process

The development process in this thesis has two main purposes. The first purpose is to create an application that can be used to stored digital certificates. The second purpose is for the developer, author and even the reader to learn things from following the process. From an academic perspective the second purpose is very interesting. This section tries to answer some of the questions the developer asked himself during the process.

6.4.1 **How should different types of certificates be handled?**

One of the most important aspects of the blockchain technology is that the data is immutable. This means that if a new property is necessary for a certificate, it is not possible to simple add this attribute to all existing certificates. This would change the hash of each block, which would lead to all blocks built on top of the changed block would detect the change and reject it. This could potentially lead to a problem if the attributes belonging to a certificate are changed.

Hyperledger Sawtooth provides the solution for this problem. Since the transaction families have a version number, they can handle different versions of certificates being stored in the blockchain. This means that all old certificates will remain in the blockchain, keeping the data immutable, while the new certificates simply are added at the end. A new transaction can make sure that any new properties can be added to the state of the existing certificates.

6.4.2 **Can the developer of the system be trusted?**

The application created for this thesis is stated to be secure and decentralized. This statement is made by the developer of the application. It would be possible for the developer to include their own loophole which allows them trick the system and potentially create fake certificates that look real. Another, and more likely scenario,

is that there exists unseen loopholes that another, malevolent user can find and exploit at a later time.

Whoever intends to use this system should be critical towards the application and make their own verification of these statements. The good news is that due to the lack of centrality of the blockchain technology, and the application, it is possible for someone else to go through the code for the application. If they are content with the code, they can also trust the system. This is one of the key properties of the blockchain technology, it moves trust from a third party to the technology itself.

6.4.3 **What incentive is there to run a validator?**

For the application that has been developed for this thesis to be used in a real environment, multiple validators are needed. In a cryptocurrency there is an incentive to run a validator since each time a block is mined, there is a reward. In addition to this there are transaction fees. The difference with this application is that there is no such asset to give out to validators.

The developer believes that actors that are interested in this kind of a system would be willing to invest the resources needed to run a validator. This would strengthen the trust of the application and make certificates stored on the blockchain more secure. If there are no actors interested in making this, there is perhaps no interest for the application itself.

7 Conclusion & Discussion

In this chapter the author summarizes the conclusions to be drawn from the rest of the thesis. Discussion on the results and thoughts about future research are also presented.

7.1 Conclusion

The purpose of the thesis can be summarized in the following points:

1. Investigate the blockchain technology
2. Investigate how certificates are stored and accessed today
3. Design and develop a blockchain application for storing digital certificates
4. Apply the strengths and weaknesses of the blockchain technology to a practical context

The first purpose is fulfilled by the theory section on cryptography and blockchain. The second purpose is achieved through the case study and the analysis of this. The third purpose is achieved through the application that has been developed in this thesis. The fourth purpose is fulfilled in the analysis of the application in chapter 6.

(RQ1) How are certificates stored and accessed today?

There are many different ways to store and access certificates today. There are physical and digital certificates. The digital certificates are stored either at the owner of the certificate, the certifier that issued the certificate or at an external website. There is no single way to check the validity of a certificate.

(RQ2) How can the blockchain technology be used to design and develop a system for storing digital certificates?

The application developed in this thesis is built using Hyperledger Sawtooth. This application is transparent, secure and decentralized. It is shown that it can be developed by a single developer in a relatively short amount of time. It has scaling capabilities to be used in a real environment.

7.2 Discussion

7.2.1 **Validity of the result**

The answer on (RQ1) has been based on a smaller case study. This has been in a more qualitative nature to achieve a deeper understanding of the situation. To be able to draw a conclusion on the whole certification industry a more quantitative study should be done with a larger sample size.

The application developed has been evaluated by the developer himself. As it has been discussed in section 6.4.2 the code should be evaluated by anyone who wishes to use it in a real environment.

In addition to this the concept of blockchain as a solution for storing digital certificates has been discussed in more general terms with the interviewees of the case studies. Many have been hopeful and positive towards blockchain as a concept. The application itself has not been tested in a real environment.

7.2.2 **Future research**

Future research in this area should be based on the shortcomings found in this thesis. Future research could for instance take the application developed in this thesis and develop it further. By setting up a real network and developing a mobile application or a web interface it could be tested in a real situation. This would achieve two things. The first is that the technical solution can be more thoroughly tested and evaluated. The second thing is that it could be investigated how the industry would accept this system. It does not matter how good a system is if there are no users.

References

[1] About KRAV (2018) Retrieved January 4, 2019 from https://www.krav.se/in-english/about-krav/

[2] Google Trends (2019) Retrieved January 4, 2019 from https://trends.google.com/trends/explore?date=all&q=blockchain

[3] Nakamoto S. (2008) *Bitcoin: A Peer-to-Peer Electronic Cash System.* Retrieved January 4, 2019 from https://bitcoin.org/bitcoin.pdf

[4] Global Charts (2019) Retrieved January 4, 2019 from https://coinmarketcap.com/charts/

[5] Sveriges BNP (2018) Retrieved January 4, 2019 from https://www.scb.se/hitta-statistik/sverige-i-siffror/samhallets-ekonomi/bnp-i-sverige/

[6] Iansiti, M., & Lakhani, K. R. (2017). *The truth about blockchain.* Harvard Business Review, 95(1), 118-127.

[7] Marechaux J. (2018) *Blockchain is Not a Silver Bullet.* Retrieved January 4, 2019 from https://medium.com/pragmatic-technology/blockchain-is-not-a-silver-bullet-754eb49e1e91

[8] Höst, M., Regnell B. & Runeson P. (2006) *Att genomföra examensarbete.* Lund, Sverige: Studentlitteratur

[9] ISO 14000 family – Enviromental Management [ca. 2018]. Retrieved January 14, 2019 from https://www.iso.org/iso-14001-environmental-management.html

[10] The facts about certification [ca. 2018]. Retrieved January 4, 2019 from https://www.iso.org/certification.html

[11] Myrén K. [ca. 2018] *Ackreditering eller certifiering?* Retrieved January 4, 2019 from https://www.swedac.se/swedac_magasin/ackreditering-eller-certifiering/

[12] What does Swedac do? [ca. 2018] Retrieved January 4, 2019 from https://www.swedac.se/about-swedac/what-does-swedac-do/

[13] Smart N. (2003) *Cryptography: An introduction.* London, United Kingdom: McGraw-Hill

[14] Judmayer A., Stifter N., Krombholz K. & Weippl E. (2017). *Blocks and Chains: Introduction to Bitcoin, Cryptocurrencies, and Their Consensus Mechanisms.* San Rafael: Morgan & Claypool

[15] Narayanan A., Bonneau J., Felten E., Miller A. & Goldfeder S. (2016) *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton: Princeton University Press

[16] Introduction to Hyperledger Sawtooth (2018) Retrieved January 4, 2019 from https://sawtooth.hyperledger.org/docs/core/releases/latest/introduction.html)

[17] Lamport, L., Pease, M., & Shostak, R. (1982). The Byzantine generals problem. Menlo Park, CA: SRI International.

[18] PoET 1.0 Specification (2018) Retrieved January 4, 2019 from https://sawtooth.hyperledger.org/docs/core/releases/latest/architecture/poet.html

[19] About Hyperledger (2018) Retrieved January 4, 2019 from https://www.hyperledger.org/about

[20] Global State (2018) Retrieved January 4, 2019 from https://sawtooth.hyperledger.org/docs/core/releases/latest/architecture/global_state.html

[21] Transaction and Bathces (2018) Retrieved January 4, 2019 from https://sawtooth.hyperledger.org/docs/core/releases/latest/architecture/transactions_and_batches.html

[22] Glossary (2018) Retrieved January 4, 2019 from https://sawtooth.hyperledger.org/docs/core/releases/latest/glossary.html

[23] The farmer. Personal conversation (2018).

[24] Employee at the certification body. Personal conversation (2018).

[25] Employee at the research institute. Personal conversation (2018).

[26] Employee at the certificate website. Personal conversation (2018).

[27] Creating a Sawtooth Network (2018) Retrieved January 4, 2019 from https://sawtooth.hyperledger.org/docs/core/releases/latest/app_developers_guide/creating_sawtooth_network.html

Appendix A

Appendix A contains the source code for the application developed for this thesis. Note that this is not all the code that has been written, but is instead focused on the unique parts of this application. Anyone who wishes to run the application should follow the Hyperledger Sawtooth documentation and use the following classes to handle certificates.

A.1 main.py

```
from sawtooth_sdk.processor.core import TransactionProcessor

from handler import CertTransactionHandler

def main():

    processor = TransactionProcessor(url='tcp://127.0.0.1:4004')
    handler = CertTransactionHandler()
    processor.add_handler(handler)
    processor.start()

if __name__ == "__main__":
    main()
```

A.2 handler.py

```
from sawtooth_sdk.processor.handler import TransactionHandler
from sawtooth_sdk.processor.exceptions import InvalidTransaction

from cert_payload import CertPayload
from cert_state import Certificate
from cert_state import CertState
from cert_state import CERT_NAMESPACE

class CertTransactionHandler(TransactionHandler):

    @property
    def family_name(self):
        return 'cert'

    @property
    def family_versions(self):
        return ['1.0']

    @property
    def namespaces(self):
        return [CERT_NAMESPACE]
```

```
def apply(self, transaction, context):
    header = transaction.header
    signer = header.signer_public_key

    cert_payload = CertPayload.from_bytes(transaction.payload)

    cert_state = CertState(context)

    if cert_payload.action == 'create':
        cert = Certificate(company=cert_payload.company,
                           certifier=cert_payload.certifier,
                           date=cert_payload.date,
                           duration=cert_payload.duration,
                           extra=cert_payload.extra)
        cert_state.create_certificate(cert)
    else:
        raise InvalidTransaction('Invalid action: {}'.format(cert_payload.action))
```

A.3 cert_state.py

```
import hashlib

CERT_NAMESPACE = hashlib.sha512('cert'.encode("utf-8")).hexdigest()[0:6]

def _make_cert_address(certificate):
    s = certificate.company + certificate.certifier + certificate.date
    return CERT_NAMESPACE + hashlib.sha512(s.encode("utf-8")).hexdigest()[:64]

class Certificate:
    def __init__(self, company, certifier, date, duration, extra):
        self.company = company
        self.certifier = certifier
        self.date = date
        self.duration = duration
        self.extra = extra

class CertState:

    TIMEOUT = 3

    def __init__(self, context):
        self._context = context

    def create_certificate(self, certificate):
        address = _make_cert_address(certificate)
        certificates = self._load_certificates(address)
        certificates[address] = certificate
        self._store_certificate(address, certificates=certificates)
```

```
def get_certificate(self, certificate):
    address = _make_cert_address(certificate)
    return self._load_certificates(address).get(address)

def _store_certificate(self, address, certificates):
    state_data = self._serialize(certificates)

    self._context.set_state({address: state_data}, timeout=self.TIMEOUT)

def _load_certificates(self, address):
    state_entries = self._context.get_state([address], timeout=self.TIMEOUT)

    if state_entries:
        certificates = self._deserialize(data=state_entries[0].data)
    else:
        certificates = {}

    return certificates
```

```
def _deserialize(self, data):
    certificates = {}
    try:
        for cert in data.decode().split("|"):
            company, certifier, date, duration, extra = cert.split(",")
            certificates[company] = Certificate(company, certifier, date, duration, extra)
    except ValueError:
        raise InternalError("Failed to deserialize certificate data")

    return certificates

def _serialize(self, certificates):
    cert_strs = []
    for _, c in certificates.items():
        cert_str = ",".join([c.company, c.certifier, c.date, c.duration, c.extra])
        cert_strs.append(cert_str)
    return "|".join(sorted(cert_strs)).encode()
```

A.4 cert_payload.py

```
from sawtooth_sdk.processor.exceptions import InvalidTransaction


class CertPayload:
    def __init__(self, payload):
        try:
            action, company, certifier, date, duration, extra = payload.decode().split(',')
        except ValueError:
            raise InvalidTransaction("Invalid payload serialization")

        if not action:
            raise InvalidTransaction('Action is required')
        if not company:
            raise InvalidTransaction('Company is required')
        if not certifier:
            raise InvalidTransaction('Certifier is required')
        if not date:
            raise InvalidTransaction('Date is required')
        if not duration:
            raise InvalidTransaction('Duration is required')
        if not extra:
            raise InvalidTransaction('Extra is required')

        self._action = action
        self._company = company
        self._certifier = certifier
        self._date = date
        self._duration = duration
        self._extra = extra
```

```
@staticmethod
def from_bytes(payload):
    return CertPayload(payload=payload)

@property
def action(self):
    return self._action

@property
def company(self):
    return self._company

@property
def certifier(self):
    return self._certifier

@property
def date(self):
    return self._date

@property
def duration(self):
    return self._duration

@property
def extra(self):
    return self._extra
```