UiA University of Agder

Master

Blockchain-based Smart Contracts for Consent Management in eHealth

HUY TRAN AND THANUKAN JEGATHEESWARAN

SUPERVISOR
Associate Professor Martin W. Gerdes

CO-SUPERVISOR
Professor Vladimir Oleshchuk

University of Agder, 2021
Faculty of Engineering and Science
Department of Engineering and Sciences

Obligatorisk gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

1.	Vi erklærer herved at vår besvarelse er vårt eget arbeid, og at vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.	Ja
2.	**Vi erklærer videre at denne besvarelsen:** • Ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • Ikke refererer til andres arbeid uten at det er oppgitt. • Ikke refererer til eget tidligere arbeid uten at det er oppgitt. • Har alle referansene oppgitt i litteraturlisten. • Ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	Ja
3.	Vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	Ja
4.	Vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert.	Ja
5.	Vi er kjent med at Universitetet i Agder vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk.	Ja
6.	Vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider.	Ja
7.	Vi har i flertall blitt enige om at innsatsen innad i gruppen er merkbart forskjellig og ønsker dermed å vurderes individuelt. Ordinært vurderes alle deltakere i prosjektet samlet.	Nei

Publiseringsavtale

Fullmakt til elektronisk publisering av oppgaven Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).
Oppgaver som er unntatt offentlighet eller taushetsbelagt/konfidensiell vil ikke bli publisert.

Vi gir herved Universitetet i Agder en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	Ja
Er oppgaven båndlagt (konfidensiell)?	Nei
Er oppgaven unntatt offentlighet?	Nei

Acknowledgement

We would first like to express our utmost gratitude to our supervisors from the University of Agder, Associate Professor Martin W. Gerdes and Professor Vladimir Oleschunk, for guidance and feedback during the writing of thesis. Their academical experience through valuable supervision has genuinely helped us improve the quality of this work. An extended thanks goes to Associate Professor Gerdes for allowing us to pursue a highly interesting and equally challenging thesis subject.

Furthermore, we would like to thank Dr. Kugathas Subramaniam and Mr. Prashanth Sivaganesh for using their academical experience in writing to aid us in proof-reading and feedback on thesis composition.

Last but not least, we want to thank our family and friends, as well as our partners for their endless support and unconditional love, and we hope they forgive us for the amount of time spent at school and not at home.

Abstract

Since the introduction of Bitcoin by Satoshi Nakamoto in a white paper in 2008, Blockchain has gathered considerable attention because of its ability to be decentralized and immutable. Blockchain is still considered as a new and experimental technology, and the state-of-the-art literature review was conducted to identify various use cases for the Blockchain technology for the healthcare industry. Consent management is one of the most critical components in healthcare because of the constantly evolving eHealth services requiring access to personal data, and of corresponding privacy laws, such as the European General Data Protection Regulation (GDPR), created to provide patients with more control over their healthcare data. This thesis focuses on how Blockchain can be used to facilitate consent management, where the patients are in control of who can access their personal data. This thesis creates a consent management solution for healthcare data using Blockchain-based smart contracts built on an Ethereum platform. These smart contracts are developed in Solidity programming language, and deployed in a test network for verification.

Our study shows that consent management can be implemented through Blockchain-based smart contracts. In order to be compliant with the applicable GDPR requirements, our conceptual design enables patients to have control over their own data through the use of various smart contracts. Through our conceptual design we describe scenarios of how patients can give and revoke consent utilizing smart contracts. We set up a test network and deployed a simple smart contract that demonstrates how a patient can sign consent. Furthermore, limitations are briefly discussed and future work should address these in order to successfully implement our design in healthcare.

Contents

List of Figures

List of Tables

1 Introduction

In the healthcare sector a massive amount of sensitive and critical data is stored and shared to help the patients, the healthcare professionals, and other stakeholders in the best possible way. Data security and data ownership are always in the focus when talking about personal health data. The rapid development in healthcare has resulted in the digitalization of health records, and the patients' smart devices that gather large amounts of "Person Generated Health Data (PGHD)" create a concern against secure data storage, processing and provisioning. Storing and sharing personal data done properly will be beneficial for all stakeholders in the healthcare system. Unfortunately, data storage and sharing between different healthcare service providers has fallen behind on the adoption of electronic health records (EHR). The lack of a secure infrastructure for storing sensitive data can cause major data breaches. Furthermore, patients still do not have full ownership of their data, which causes a worrying trend correlating to the declining trustworthiness towards healthcare institutions.

Personal data is the patient's personal asset and must by law be owned and controlled by the patient. Storing the patient's personal data in different healthcare systems or third party "cloud" services prevents lawful data sharing, and can jeopardize the patient's privacy [1]. A difficult challenge that vendors of eHealth solutions for healthcare providers are facing, is the process of gathering, storing and analyzing personal health data without raising privacy concerns. In this digital age, patients are more willing to manage and share their data, but in a privacy-protecting way. Hence, there is a need for the shift towards a more patient-centered model in the healthcare industry and will result in new challenges related to security and privacy. In recent years, a new technology called Blockchain has been introduced. Blockchain can elevate data processing in healthcare to a new level without the patient relying on a trusted third party, and most importantly, making the patient the owner of their data.

1.1 Background on Blockchain Technology

With the growing interest in Blockchain and its adoption in different industries, Blockchain could change how data is handled in healthcare. In 2008, a white paper about a decentralized peer-to-peer network for an electronic cash system was introduced by Satoshi Nakamoto [2]. He presented an idea of Bitcoin transactions on an online platform, where payments could be transferred directly from one party to another, without the need of a trusted third party. This platform is known as Blockchain, and can be defined as a:

> "Distributed, decentralized and tamper-proof ledger without any centralized control and could increase data privacy and empower individuals with control and access over their data, including health data." [3]

Blockchain, also referred to as a Distributed Ledger Technology (DLT), is a chain of blocks. It can be seen as a decentralized information system, and preserves information about every past transactions [4, 5]. Each block plays a vital role in connecting the previous block to the following block, making it a chain of blocks [6]. This implies that a block in a chain cannot be deleted or changed, since this would change every following block [7]. The primary function of each block is to record, confirm and distribute the transactions between other blocks in the Blockchain network. It runs on a pre-selected protocol that determines the performance and validation of the transactions. One of the key security features of Blockchain is that the information stored inside is immutable [8]. Every node gets a copy of the transaction, and if an attacker tries to change any data or information on the block, it will be detected and the chain will break. As a result, data transactions cannot be altered or erased [9].

Since the release of the white paper, Blockchain has improved vastly and moved to the second generation of Blockchain-based technology. The second generation has features like smart contracts and was introduced with the open-source Blockchain platform called Ethereum [9]. Smart contracts are programmable contracts that executes functions inside Blockchain that carry out transactions when all the requirements are fulfilled [10]. A smart contract can be seen as a regulator inside the Blockchain network [3], e.g. a smart contract can authorize a doctor to access patient's medical data, but only if the patient has given consent.

1.2 Potential use of Blockchain Technology in eHealth

Electronic health, usually called eHealth, uses electronic and digital technology such as computers, mobile devices, the Internet and other related technologies to help improve healthcare services. eHealth is a field that has emerged because of the need to improve documentation and tracking of a patient's health data. Health data and medical records are traditionally kept on paper, but the use of EHR has been greatly increased as the technology has advanced [11, 12].

Patients have lost their trust in providers that maintain their health data, due to weak security systems and policy enforcement, resulting in data theft through the years [9]. However, Blockchain embodies impressive features that can be beneficial to healthcare. Blockchain can upgrade the existing way of storing and sharing data, and most importantly improve data management that struggles with balancing data privacy. Blockchain can be used as a technology for storing health data securely, while sharing the data between a patient and a third party with a focus on privacy in each step [13]. Blockchains ability of decentralized management creates an environment with no central trusted third party. This results in an immutable and tamper-proof system that eliminates the need for a central storage and authority, which removes single-point of failure and reduces the risk of tampered data [14]. Blockchain can also help anonymize a patient's identity by using cryptographic keys that represents the patient's identity [4]. Most importantly, Blockchain could transform the healthcare industry into a patient-driven interoperability, where patients can manage and share their own health data. It is time for patients to control their own data, and Blockchain can potentially be the solution to support this [15].

In order to design a Blockchain-based system relevant for real-world healthcare use cases, the solution has to comply with General Data Protection Regulation (GDPR) laws. GDPR is a set of rules put into effect by the EU in 2018 to give individuals more control over their personal data [16]. In order to fully utilize Blockchain, it has to comply with the laws of GDPR. How Blockchain in healthcare can interact and comply with GDPR remains to be addressed and therefore be a turning point for the adaptation of this new technology in the healthcare sector[15].

1.3 Project Scope and Problem Statement

1.3.1 Project Scope

In Scope

This thesis will focus on how Blockchain can be used to facilitate consent management for data sharing, where the patients are in control of who can access their personal data. This thesis aims to create a consent management for healthcare data using Blockchain-based smart contracts built on an Ethereum platform. These smart contracts are developed in the programming language Solidity, and deployed in a test network. Additionally, only selected key requirements of GDPR described in table 2 will be fulfilled in this thesis.

Out of Scope

- The detailed functioning of Blockchain inside our conceptual design will not be evaluated.
- The performance of the chosen consensus algorithm will not be evaluated.
- The thesis does not cover the evaluation of performance characteristics related to the deployment of smart contracts.
- No specific cryptography algorithm for encryption and decryption of patient health data will be chosen.

1.3.2 Problem Statement

Blockchain is still considered as a new and experimental technology, and there are many potential use cases of Blockchain technology in the healthcare sector. The overall research theme in this thesis is about how Blockchain can be used in healthcare to create a consent management for sharing of eHealth data. There have been incidents where personal health data have been misused for illegal purposes, because it has been accessed without the knowledge of the patient [17]. Consent management is the process or action to manage a patient's consent for handling health data, and therefore, it is an essential topic in healthcare. It ensures that patients can give and revoke consent as desired. The patients will by result be in control of who can access their health data. The main objective of the thesis is to collect consent from patients and store it in a secure manner, which then allows patients to revoke their consent. All use cases for Blockchain technology in healthcare sector has to comply with the strict regulations of GDPR.

1.4 Research Questions

The research question, sub-objectives and sub-question in correlation with project scope and problem statement for this thesis are as following:

Research Question: How can consent management for sharing of eHealth data be realized with the use of Blockchain-based smart contracts?

The sub-objectives for this thesis are:

- **Sub-Objective 1:** Propose a conceptual design for the use of Blockchain-based smart contracts for consent management of eHealth data
- **Sub-Objective 2:** Proof-of-concept implementation for verification of the proposed conceptual design with smart contracts for consent management.

The sub-question for this thesis is:

- **Sub-Question:** How can the selected key requirements of GDPR be fulfilled with our proposed solution?

1.5 Thesis Outline

The rest of the thesis is outlined as followed:

- Chapter 2 - State of the Art: We will present a state-of-the-art literature review for how Blockchain can be utilized and the possible usage of smart contracts in healthcare.
- Chapter 3 - Theory: We present an overview of Blockchain technology and cryptography, followed by GDPR and an outline of the selected key requirements of GDPR.
- Chapter 4 - Methodology and Tools: We describe the tools used to find a solution to the problem statement.
- Chapter 5 - Solution: We present our proposed conceptual design and the implementation of smart contracts for consent management. An explanation of scenarios regarding the access of health data is given by a step-by-step model.
- Chapter 6 - Discussion: We answer and discuss the research question, and reflect on the fulfillment of the sub-question.
- Chapter 7 - Conclusion: This section will address the solution to problem statement in regards to discussion. Additionally, it will include future work.

2 State-of-the Art

2.1 How can Blockchain be utilized in Healthcare?

The biggest question is how Blockchain can solve digital healthcare challenges and still be compliant with the GDPR. A problematic topic with the use of Blockchain in healthcare is that the data stored inside a Blockchain cannot be tampered with or be erased. One of the key aspects in the GDPR is the "right to be forgotten", meaning the right a patient has to request his or her data to be deleted. There are different solutions for these challenges, and numerous papers agree that the best way to handle this is to divide the data and store the sensitive health data off-chain [7, 9, 15, 18, 19].

The accumulated health data from wearables and mobile devices cannot be stored directly in a Blockchain due to their high frequency and large size. This means storing large amounts of health data in the Blockchain is not viable, as the data needs to be replicated across a great number of nodes. The solution will be to only store and share metadata of the original dataset and the bare minimum data required for the transaction on the Blockchain, while using secure cloud storage as a solution for storing the massive amount of health data [18, 19].

Agbo et al. [9] have solved this challenge by storing the health data encrypted and off-chain, while the instructions for accessing the data are stored in the Blockchain. The data that is stored on the Blockchain will be used as a pointer to show where the health data is stored off-chain.

Bayle et al. [7] proposed a model that uses the same approach by storing sensitive data off-chain and smart contracts to meet the requirements of the GDPR. The Blockchain will keep track of all interactions between data provider and data consumer, while maintaining records of the amassed interaction data between them. As opposed to make the Blockchain GDPR compliant, they move the sensitive data off-chain to the cloud. The provider of the cloud service is thereby responsible to be compliant with the GDPR.

Gordon et al. [15] presented a model that stores metadata like time and location in the Blockchain, and primary health data is kept off-chain. Hasselgren et al. [20] concludes that Blockchain technology in healthcare is not GDPR compliant, however, it may be accomplished in the future by the implementation of encryption.

2.2 How can Smart Contracts be used?

New opportunities emerge for administration and management of health data as the use of digital health data and cloud storage becomes a common procedure in the healthcare industry. It is therefore imperative to ensure that the patients are in control of their data by providing favorable means for sharing the data to trusted recipients in a secure manner [4]. Blockchain-based smart contracts can be the solution to address these challenges. There has been a growing interest in smart contracts ever since Ethereum first integrated the use of it [21]. To sum it up:

> "Smart contracts can allow patients to manage access to their health records, secure data exchange, and ensure the privacy of those exchanges" [21].

There are several papers that look into how smart contracts can be used in healthcare. They address various implementation methods based on different focal perspectives to create conceptual designs. Few of these designs have been tried in real-world scenarioes, acquiring valuable data for further work and research. Hoai Luan Pham et al. [22] proposed a design to manage patients' (PGHD) through Internet of things (IoT) devices using Blockchain-based smart contracts. The number of IoT devices in healthcare industry has increased exponentially, hence the main objectives of identifying and addressing patients' privacy and security issues. The produced PGHD from these IoT devices cannot continuously be written into the Blockchain due to its size and must therefore be reduced. A processing mechanism that filters data from the IoT sensors before they are sent to the Blockchain was therefore proposed. Their processing mechanism will be able to analyze the data and alert healthcare providers of abnormal data collected from sensors. They use an Ethereum test network and controlled data rates by using gas prices. A gas price in this instance is the mining cost of the Ethereum block. The flow of normal data from the sensors will be written into Blockchain with normal gas prices, however, if the processing mechanism detects abnormal data, the gas prices will be set to higher rates for the purpose of expediting the Blockchain transactions and warn the healthcare provider or doctor. Their design consists of three main participants, i.e. a hospital, a doctor, and a patient. A total of four smart contracts are deployed and contain contracts for monitoring the patient and authorization, as well as registration of patient and doctor. Note that the data is encrypted and then stored into the Blockchain, and only trusted parties such as authorized doctors and hospitals will have an access key to decrypt the data. This is to ensure security and privacy of identifiable details regarding the patient, in line with patient and third-party confidentiality and disclosure agreements in healthcare.

Asaph Azaria et al. [23] developed a prototype for decentralized record management called MedRec to manage electronic medical records using Blockchain. Their system consists of three contracts with metadata concerning record ownership, permissions and data integrity, i.e. a registration contract, a patient-provider relationship contract, and a summary contract. Patient-provider relationship contracts contain pointers that are used between different providers, and the summary contract contains the history of signed contracts between the participants. Only the patient has the right to accept or reject a relationship using the summary contract. Data exchanges are handled off-chain between databases under the assumption that the patients trust the data host providers.

Gaby G. Dagher et al. [24] proposed a Blockchain-based framework called Ancile. This framework enables efficient and secure access to medical records between patients, providers, and third parties while maintaining the privacy of the patient using Ethereum smart contracts. The patient data is stored at the provider's database as off-chain storage with the Blockchain functioning as an access control layer. Access permissions, hashes of data, and pointers to the data providers are stored in the

Blockchain. Note that the medical records are stored at the provider under the impression that the patients trust them, however, these patients are in control over their data as they are required to approve any transfer or changes by the provider. Ancile uses in total six smart contracts. Briefly, the first smart contract is a Consensus Contract and responsible for user registration and maintaining Blockchain mining. Ancile is designed to function in a permissioned Blockchain and therefore uses a consensus algorithm called QuorumChain. The second contract is the Classification Contract and used for the classification of levels for each node. Third contract, the Service History Contract, maintains the relationship history of the nodes. Ownership Contract is the forth contract and contains pointers to find requested data at the provider. The fifth contract, the Permission Contract, has an overview of different levels of access as read and transfer rights. The final contract is Re-encryption Contract that handles encryption when sharing between Blockchain and off-chain, and storing data off-chain. The involvement of these six contracts in Ancile will have a performance cost, however, the framework provides more security when compared to MedRec [23].

Kristen N.Griggs et al. [25] developed a system for secure transfer and management of medical sensors between the patient, the hospital, and the Blockchain network. The development was encouraged by the general acknowledgment among the writers that healthcare is moving towards a more patient-controlled access. They use a consortium-based Blockchain and uses Practical Byzantine Fault Tolerance (PBFT) as a consensus algorithm. The system allows different IoT devices that may or may not have different manufacturers to be formatted and processed together, and thereafter linked to one patient for better and integrated health management. The formatted data will be sent to the main contract, and then distributed to other contracts. All data will be forwarded to a trusted provider as off-chain storage. If the data is processed successfully, a new transaction will be added to the Blockchain.

Rantos et al. [26] proposes a consent management solution called ADVOCATE. This is a patient-centric solution that allows a patient to manage their consents regarding their health data. ADVOCATE is a cloud-based service that acts as an intermediary between the doctors and hospitals, and the patients. The platform is managing the access requests from a third party such as a doctor, and gather the given consents from the patients. ADVOCATE is also responsible for maintaining and protecting the integrity and versioning of the consent, based on a Blockchain. This ensures that no unauthorized alteration will be made to the consents.

3 Theory

Although the detailed functioning of Blockchain technology inside our conceptual design is out of scope for this thesis, it is important for our result and discussion to understand the technology. The chapter concludes by presenting the selected key requirements of GDPR that are important to fulfill in order to implement blockchain in healthcare.

3.1 Blockchain Technology

Blockchain is first and foremost a distributed ledger technology that was first implemented in 2008 by Satoshi Nakamoto [2]. He released a white paper for the cryptocurrency Bitcoin with Blockchain as an underlying technology to remove the need for a centralized trusted party and exchange Bitcoins among participants in a distributed network [9]. However, Haber and Stornetta were the first to come up with the idea of Blockchain in 1991 when they described their work as a cryptographically secured chain of blocks [27]. Satoshi Nakamoto never used the term Blockchain; he only used the term "a chain of blocks" in his white paper [2].

One of the key characteristics of Blockchain is decentralization by the removal of the need for a centralized trusted party. A central authority would suffer from single point of failure, but Blockchain overcomes this problem by being a distributed ledger. In a centralized ledger, all records are placed in a central place, and the central authority will consult any disagreements or any faulty participants. In a distributed ledger, however, all the participants will have a copy of the record as there are no centralized trusted third party. Blockchain uses consensus algorithms in order to maintain data consistency in a distributed network [9, 28]. There are different ways of achieving consensus in Blockchain, for example, Bitcoin uses Proof of Work(PoW) to achieve consensus in the network. Another key attribute to Blockchain is the reciprocity between a node and its digital signature. The latter is considered the backbone of Blockchain and used by a node when joining a network. It should be noted that the entering node is neither trusted or not trusted by the nodes already present in the network [29]. Said node can subsequently interact with the network to validate or send and receive transactions.

The fundamentals of Blockchain is built around cryptography and nodes. Each node acts a participant and uses public key infrastructure to create and submit transactions [9]. Transactions are simple messages that contain information, in essence, records stored within a chain of blocks and can therefore be seen as a flow of information [30]. A participant has a pair of public and private keys used for digital transactions. A digital signature is created from both keys and used for signing and verifying a transaction. The private key is confidential and should never be disclosed, while the public key is freely shared. A transaction includes the public key of both the user and the receiver, and a transaction message. Before the transaction is broadcasted throughout the Blockchain, the user finalizes a transaction by digitally signing it with his private key. For the receiver to verify the transaction, he uses the sender's public key in order to check if the transaction has been tampered with. There may be concerns regarding privacy if these transactions are public, however, Blockchain offers anonymity because the users interact with generated addresses that do not contain distinct details that can be used to identify and verify a user's real identity [2, 9, 28].

3.1.1 Types of Blockchain

Blockchain is divided into two subcategories, permissionless and permissioned Blockchain. A permissionless Blockchain is an open network and therefore accessible to everyone to join and interact with each other without needing permission to enter the network. Bitcoin was the first permissionless Blockchain and its primary application is transfers of digital currencies between users. Ethereum, the focal platform of this thesis, is another permissionless Blockchain [31]. These are some of the advantages of permissionless Blockchain [31]:

- It is not required to have complete trust in any single entity in a permissionless Blockchain. Everyone can read and trace the transactions in the network.

- Participants do not need to disclose their real identities, as their digital signature acts as identification of each participant in the network.
- Permissionless Blockchain is decentralized and immutable as it is open to all and therefore a transparent network.

The main drawback of permissionless Blockchain is the low performance of transaction speed. This is clearly evident in Bitcoin during its applications. It is also harder to scale, and the data is publicly accessible as it is fully replicated among all nodes. This implies that this type of Blockchain can suffer from confidentiality issues [29]. Permissionless Blockchain will therefore not suit every need as the drawbacks are of major concern. Permissioned Blockchain was introduced as an alternative to address these concerns.

Permissioned Blockchain was introduced to run Blockchain technology among a set of known and identifiable participants. This type of Blockchain subcategory requires the participants to have permission to join and participate in the Blockchain network [32], addressing the drawbacks of the aforementioned permissionless Blockchain. The subsequent exclusion of nonviable participants serves as a purpose of making the Blockchain network private. Permissioned Blockchains are therefore more practical when compared to permissionless Blockchain and thus favorable for business applications with organizations as the governing entity of the network. A prime example from the business perspective is the requisite of identifiable participants in a typical use case [29]. The trust level of participants in a permissioned Blockchain is by that reasoning higher in comparison to a permissionless Blockchain, which implies that the permissioned system is more convenient when trusted participants is a desired prerequisite [33]. Some of the advantages following the use of permissioned Blockchain are:

- Permissioned Blockchain is far more scalable than permissionless and offers an organization the ability for customization [31].

- Transaction speed is more effective and faster as the organization chooses a suitable consensus method, and transaction validation is done by a predefined set of nodes [29].
- Generally better in terms of confidentiality. The network is private, and all sensitive data within the network is therefore isolated from public access [29].

The drawbacks of permissioned Blockchain correlates with these advantages to some extent. This subcategory of Blockchain is not truly decentralized considering a central administrative entity or entities decide user roles and participation. Additionally, the use of such Blockchain network requires more trust in participants including their consent to be identified, making each participant less anonymous [29, 31].

The Blockchain systems are categorized roughly into public Blockchain, private Blockchain, and consortium Blockchain. A public Blockchain is for all intents and purposes a permissionless Blockchain. Anyone can join the network without permission and take part in the consensus process and transactions, with records visible to the public. As a consequence, these records cannot be tampered with as many participants store the records. The system will in addition experience efficiency challenges due to low performance during transactions. A private Blockchain is a permissioned Blockchain and is fully controlled by one organization. The Blockchain system is therefore in all aspects regarded as a centralized network, evidently during the consensus process in which the organization regulates the network by accepting only those approved and/or affiliated with the organization. Consortium Blockchain is similarly to a permissioned Blockchain, but governed by multiple organizations in the network. It is for this reason viewed as partially decentralized, as a group of pre-selected nodes are responsible for validating blocks in the network. Both private and consortium Blockchain is more effective for transaction throughput than public Blockchain as a result of fewer validators [28].

3.1.2 Blockchain Architecture

As mentioned earlier, transactions are records that are stored in a chain of blocks. Each block consists of a collection of valid transactions and are chained together to form a Blockchain. The amount of transactions that are stored inside a block is dependent on block size and transaction size. As an exemplification, Bitcoin limits its block size to 1MB. Validation of transaction is the process of ensuring the legitimacy of the proposed transaction, e.g. verification of sender by reviewing if the transaction was sent from an authorized user. Once the transaction is validated, it will be placed into a block, and the order in which the validated blocks are chained together is determined by the consensus algorithm of the Blockchain. There are special nodes in Blockchain called miners that are responsible for running the consensus algorithms. A miner's job is therefore to validate transactions, and determine the order of implementation regarding said transaction blocks into the Blockchain [9]. The phrase "mining" refers to this process and is to be considered a household term in 2021 due to the popularity of Bitcoin and cryptocurrency in general.

In order to chain blocks, Blockchain uses another cryptography function called hashing. In short, hashing is a mathematic function that can convert an input of arbitrary length into a fixed length of hash output. No matter how large a message or a file is, it will always produce the same unique hash with the same size. Hashes are often called "collision-free" because two different messages will never produce the same hash output. And the same message will always produce the same exact hash output. In addition, hashes are "one-way", meaning that a hash can never be used to find the original data or message, making it tamper-proof. Even one small change to the data or the message will change the entire hash output. This feature makes hashing the backbone of Blockchain. There are different hashing algorithms, and Blockchain uses the SHA-256 algorithm that generates a hash output length of 256-bit every time [34].

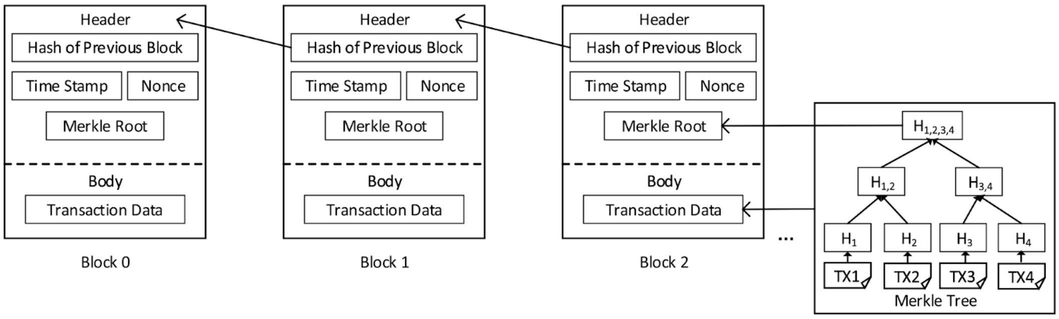


Figure 1: Blockchain architecture [35]

Figure 1 shows how blocks are chained in order to form a Blockchain. A block is made up of the block header and the block body. The block body consists of transactions, and as mentioned before, the amount of transactions depends on the block size and the size of each transaction. The block body consists of constants as Time Stamp, Hash of the previous block, Merkle Root, and variable of a nonce which is relevant for the consensus algorithms like Proof of Work. One of the most important parts included in the block header is the hash of the previous block. For each block, a hash of the previous block's header is included. Hash of the previous block links every valid block to the ones before it. Therefore, by linking to the previous block, a chain of blocks is created and a Blockchain is established. The first block in a Blockchain is called the genesis block and has no previous block hash [9, 28]. Merkle tree is an important feature that makes sure transactions are not altered. Every transaction on the Blockchain has a hash related to it, and these hashes are structured like a tree that is linked together, forming a parent-child relation [36]. This results in the most important feature of Blockchain, being immutable. If any alterations in the block occur, the hash output will change, resulting in breaking the chain. This is one of the reasons Blockchain is immutable, and immutability ensures that records cannot be retrieved or modified once created. The only way of changing anything after being stored inside a Blockchain is by creating a new record to update the old record [9].

Bitcoin is a public Blockchain where everyone can participate in the mining process. Different miners can produce different valid blocks, but only one miner is allowed to add their valid block. This is why there is a need to reach a consensus in a distributed network, which can be done in different ways [9].

3.1.3 Consensus Algorithm

Since Blockchain is a trustless decentralized network, it becomes difficult for the participants in a decentralized network to come to a consensus, and this is known as the Byzantine Generals Problem. It is used in computing to refer to situations where certain participants in the system fail, are corrupt or disagree. The problem of disagreement and the required task is not tackled is known as Byzantine Fault [37]. There needs to be reached a common conclusion among the participants for which blocks containing transactions are accepted into the Blockchain network. Within the Blockchain architecture, consensus is an essential part. For the network to function properly, consensus needs to be reached consistently [29, 37]. However, there are several algorithms to choose from in order to reach consensus in the Blockchain and these are some of them:

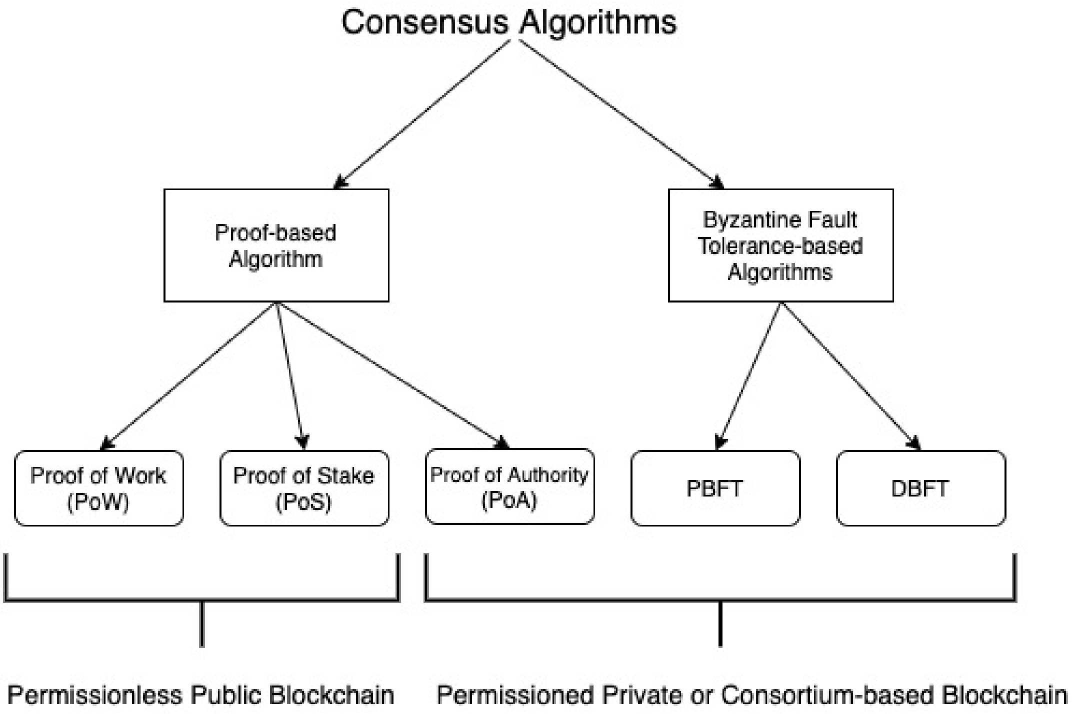

Consensus Algorithms
Proof-based Algorithm
Byzantine Fault Tolerance-based Algorithms
Proof of Work (PoW)
Proof of Stake (PoS)
Proof of Authority (PoA)
PBFT
DBFT
Permissionless Public Blockchain
Permissioned Private or Consortium-based Blockchain

Figure 2: Consensus Algorithms

Proof-based algorithms consist of miners that have to prove that they can mine blocks. The most popular proof-based algorithm is Proof of Work (PoW), introduced to be implemented in Bitcoin by Satoshi Nakamoto [2]. A node has to be selected to record transactions in a decentralized network, create a block, and distribute it to other nodes in the network. This process is referred to as block creation, and the node that does all this work is rewarded. However, multiple nodes in a decentralized network want to be rewarded, and only one node is selected for this process. The best way to do this is by randomly selecting nodes for block creation. This will, however, be vulnerable to attacks as random selection can invite malicious nodes. PoW selects miners by giving the node a lot of work, as in computing calculation, to become the miner and start the block creation. This results in different nodes competing against each other in order to solve the computing calculation and add their block of transactions into the Blockchain. Miners have to do a lot of computing calculations, and the reason for this is that when a node does a lot of work, they are more likely not to attack the network. However, all these computer calculations lead to a lot of energy and resources wasted and this is a major drawback with PoW [28, 29].

Proof of Stake (PoS) is considered to be an alternative to PoW. It is an improvement to PoW when it comes to energy consumption. In PoS, a user has to stake their cryptocurrency in order to participate in the mining or validation in the network. A miner gets their mining power based on how many coins they hold [38]. This means that a miner is selected based on a probability proportionate to the amount of its stake in order to forge the next block [29]. By selecting which miner gets to forge the next block based on their account balance can be seen as an unfair selection, as the richest person can dominate in the network. However, it is believed that a user with more stake is less likely to be fraudulent [28]. This is because if the miner happens to be a fraud, they will be punished by losing some of their staked coins. This solution makes the PoS more energy-friendly and efficient than PoW.

Proof of Authority (PoA) is a reputation-based algorithm and is especially effective and practical for permissioned Blockchains. It can be seen as a hybrid consensus algorithm based on PoS and Byzantine Fault-tolerant (BFT). Instead of miners and validators staking coins as they do in PoS, they stake their reputation in PoA. To participate in the network, they have to use their real identity, and therefore their reputation is at stake [39]. PoA uses trusted nodes called authorities and are considered to be trustworthy entities. These authorities are the ones that need to reach a consensus to process transactions. PoA uses mining rotation, which is used to distribute the responsibility of block creation amount authorities fairly [29, 39].

BFT is about the network working correctly and consistently reaching consensus even with bad actors who post false transactions or fail to send information. In short, BFT is a system that reaches consensus even if the system exists of some malicious nodes. PBFT tolerates byzantine faults and is a replication algorithm. PBFT can handle up to one third of malicious nodes in the network [28]. The nodes are referred to as replicas and divided with a leader, the primary node, and the rest are the backup nodes. In PBFT, all nodes are known to the network, and they need to constantly communicate with each other in order to reach a consensus [29]. Since the nodes are constantly communicating with each other, PBFT only performs well in a network that consists of a small group of nodes. Therefore it is suitable for a permissioned Blockchain system [29]. Delegated Byzantine Fault Tolerance (DBFT) can be seen as an improved version of PBFT. DBFT is a proxy voting system that allows large-scale participation to reach a consensus. Proxy voting is best described by [29] as:

> "Proxy voting means participants can delegate their votes to representatives every round, and a selected group of representatives reach consensus between themselves in the PBFT manner" [29].

DBFT is more efficient than PBFT since there is delegation, which means a smaller group of participants reaches consensus. This is very good for scaling and is preferred in a permissioned Blockchain that consists of many nodes. However, both PBFT and DBFT have restrictions on the amount of validation nodes [29].

3.2 Bitcoin

Bitcoin was first introduced in 2008 by an individual or a group of individuals under the name Satoshi Nakamoto [2]. It was introduced as a peer-to-peer electronic cash system that allowed an online payment to be sent directly between two parties without the need of a third party, such as a bank. Bitcoin was the first to realize this cryptocurrency concept, where you can buy and sell without a trusted third party [2, 40].

Bitcoin uses cryptographic proof instead of trusting a third party for two parties that are willing carry out an online transaction directly with each other. Every transaction that happens in the Bitcoin platform is recorded and made public to everyone, making it harder to reverse or fake a transaction. Bitcoin also solves the double-spending problem. Double-spending is when someone tries to spend the same money twice. Bitcoin prevents this problem by implementing a confirmation mechanism that only confirms and verifies the first transaction. The second transaction will be identified as invalid by the confirmation mechanism and will not be verified. If both transactions are taken simultaneously from the confirmation pool, the Blockchain will include the transaction with the most confirmations [2, 41, 42].

In Bitcoin, a block is generated and added to the end of the chain approximately every 10 minutes. Every block contains a detailed record of the transactions that are confirmed and verified in each period [2, 42].

3.3 Ethereum

The most particular thing that makes Ethereum different from Bitcoin is that Ethereum does not act only as a payment system but also as a computing platform [43]. Ethereum is referred to as the ”next-generation smart contract and decentralized application platform” in the Ethereum white paper by Vitalik Buterin [44]. The main components and features of Ethereum are introduced in this white paper, including its overall design, the state of Ethereum, mining process, application on token systems, financial derivatives, decentralized file storage and challenges [44].

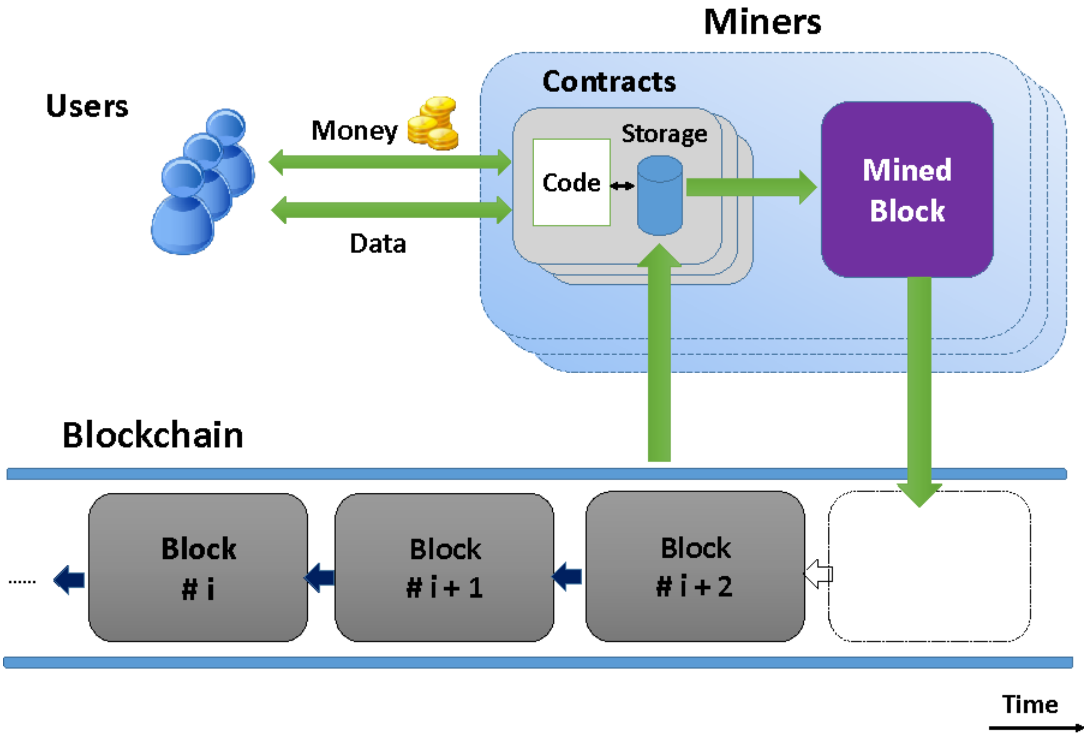


Figure 3: Schematic of a Blockchain system with smart contracts [45]

Figure 3 demonstrated by Delmolino et al. [45], is a schematic of a Blockchain system with smart contracts, and in this case, it is for the Ethereum platform. There is a smart contract program that is executed by a network of miners. When the miners reach a consensus on the result of the execution, the contract's state on the Blockchain is updated accordingly. As well as sending money, in this case Ether, the users can also send data or messages to a contract. Therefore, smart contracts play a big part in Ethereum because they are one of the main components in a platform like this [45].

In order to prevent over-consumption of resources, for example a smart contract program that causes the miners to loop forever, Ethereum introduces the concept of "gas" fee. Whenever a user creates a transaction, the user must spend "gas" in order to execute the transaction. "Gas" can be obtained by Ethereum's cryptocurrency

Ether, which can be purchased with real currency. The "gas" fee is not only to prevent over-consumption of resources but also to motivate and drive miners to execute these transactions, and also to prevent DoS attacks [45, 46].

3.3.1 Ethereum Protocol Changes

Protocol change, also known as a hard fork, is:

> "A radical change to a network's protocol that makes previously invalid blocks and transactions valid, or vice-versa. A hard fork requires all nodes or users to upgrade to the latest version of the protocol software" [47].

When a hard fork happens, the nodes of the newest Blockchain will not accept the older version of Blockchain anymore, and this results in a permanent divergence from the prior version of Blockchain [47]. There are many reasons why a hard fork may be implemented, such as adapting the system to handle new needs, adding new functionalities, or fixing security risks found in the older versions [47, 48]. Currently, Ethereum uses PoW as its consensus algorithm, however, the Ethereum community is changing the consensus algorithm to PoS in the near future [49].

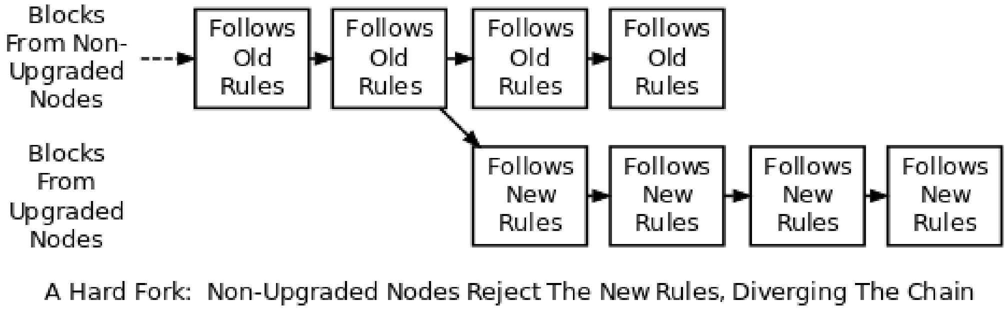


Figure 4: A hard fork in Blockchain [50].

3.4 Smart Contracts

Smart contracts are one of the main components in several platforms and applications that are developed using Blockchain. A smart contract is a program that automatically runs when some preset conditions are met [51, 52]. A smart contract can be developed to send funds between two parties, get an agreement between two parties, or record and transfer data. When a smart contract is deployed and executed in the Blockchain, it is added to a block in the Blockchain, and cannot be changed or deleted because of the security and immutability that Blockchain offers. Most smart contracts are written in a programming language that is suited for it, such as Solidity, which is the most popular programming language for smart contracts because it is supported by the Ethereum developers [51, 53].

Before a smart contract can be deployed and executed on a Blockchain, a payment fee for the transactions is required in order for the smart contract to be added to the Blockchain. The smart contract will then, in the case of Ethereum Blockchain, be executed on the Ethereum platform [45, 51]. When smart contracts become complex, the gas fee for the deploying and executing smart contracts will be higher. Therefore, the gas fee also prevents smart contracts from over-consuming resources on the Ethereum platform [51, 54].

3.5 Cryptography

Cryptography is the science of secure communication techniques that only allows the sender and the recipient to see the contents of a message. Cryptography is closely related to encryption, which is the process of taking a readable text referred to as plaintext and turning it into something unreadable, called ciphertext, by using an algorithm or series of mathematical operations. In order for the intended recipient to read this scrambled text, he or she needs to turn the ciphertext back into plaintext, which is called decryption. Currently, there are different types of cryptography, but the most common ones are symmetric and asymmetric cryptography [55, 56].

3.5.1 Symmetric Encryption

Symmetric encryption is the most straightforward method, and it uses mathematical permutations to encrypt and decrypt a message. In symmetric cryptography, the same key is used to encrypt and decrypt. The sender and the recipient have exact identical copies of the key, and the key is kept secret and not shared with anyone. The secret key can be a number, a word or a string of random letters to scramble and change the content of a message in a certain way [57, 58].

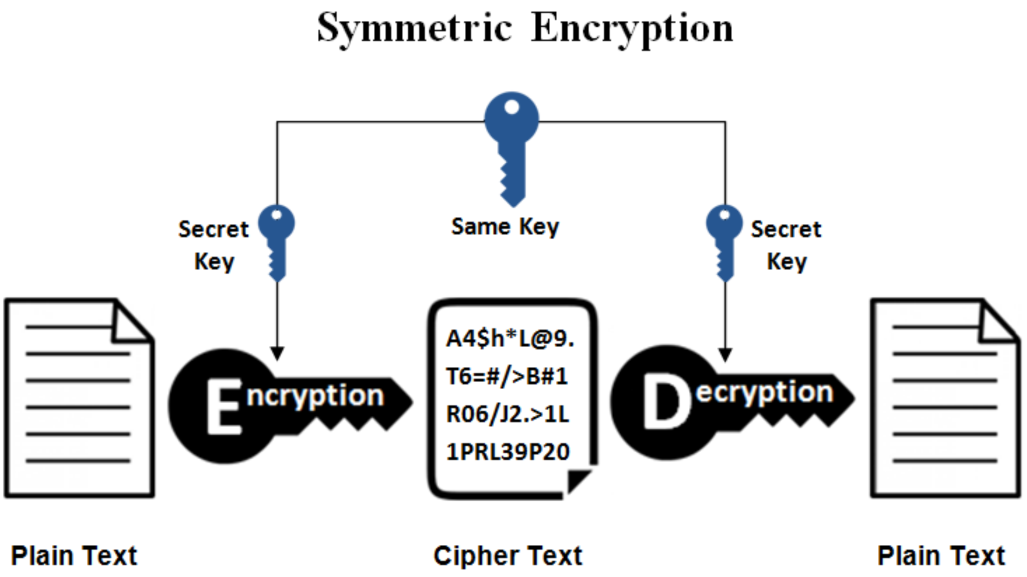


Figure 5: How symmetric encryption works [58]

3.5.2 Asymmetric Encryption

Asymmetric encryption, also known as public-key cryptography, is newer and more advanced compared to symmetric encryption. Asymmetric encryption uses a key pair of two different keys to encrypt and decrypt data [58]. A public key that is freely shared and made available for anyone who intends to communicate with you, and a private key that is kept secret and not shared with anyone [59].

A plaintext that is encrypted using a public key can only be decrypted with the corresponding private key, and a plaintext that is encrypted using a private key can only be decrypted with the public key. The security regarding the public key is unnecessary because it is freely shared and publicly available for anyone. This removes the risk of having the key compromised because the message can only be decrypted using the private key [58, 59].

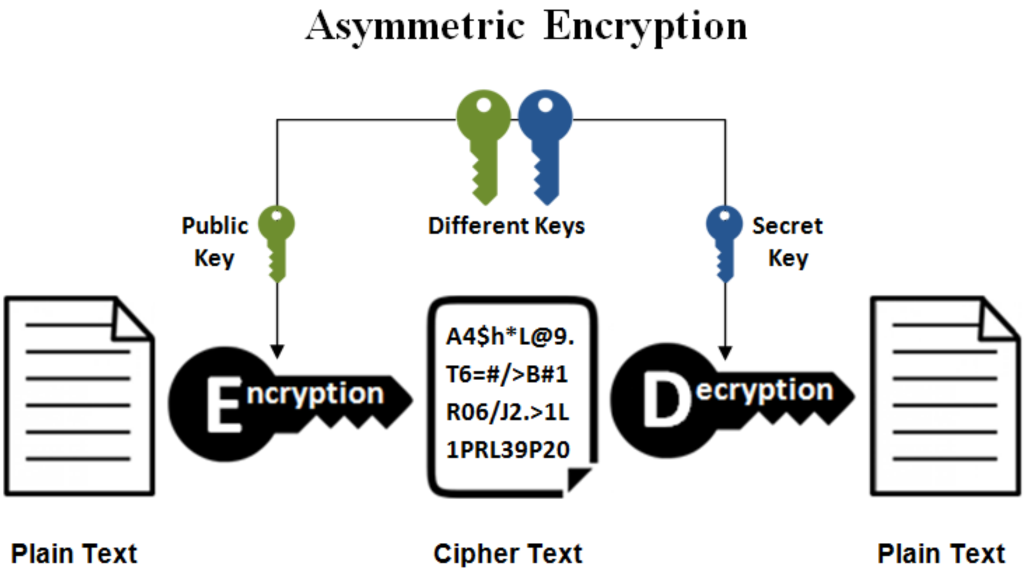


Figure 6: How asymmetric encryption works [58]

3.6 GDPR

General Data Protection Regulation is a set of rules created to give individuals more control over their personal data. It exists to solve the public concern over privacy and came into force in 2018. Organizations from both within and outside of the EU that provide services to businesses or individuals within the EU have to comply with GDPR, and as a consequence, noncompliant organizations will face penalties [60, 61]. Personal data must be processed the right way, and there must be a purpose for collecting personal data. GDPR defines two types of data handlers, the data controller and the data processor. In the context of healthcare, a data controller can be a patient that determines the purpose of why and how their personal data will be processed [62]. The data processor can be a hospital that processes personal data on behalf of the controller. Data processing is about storing, collecting, erasing, or organizing data [63].

GDPR has had a major effect in the healthcare industry on collecting, processing, and securing personal health data. It has pushed healthcare institutions to require permission from patients for using their data to ensure that collected data is applied for a specific purpose and for that purpose only. GDPR identifies that health data ownership and thereby control of the data should be with the patients [3]. GDPR requires data processors, such as the organizations that stores and collects personal data, have to report a data breach within a time period and notify the affected data controllers [64, 65].

Table 2 shows an overview of the selected key requirements of GDPR that are relevant to the created scenarios and needs to be fulfilled when applying Blockchain in healthcare. The requirements represent which GDPR laws organizations within healthcare have to consider during implementation and application of Blockchain. In layman terms, the table summarizes some essential articles of GDPR that are meant to protect sensitive data.

Art. in GDPR	**Description**	**Effect in healthcare**
Art.6 [66]	"Lawfulness of processing" [66]	The data processor has to have a purpose for processing health data. It is required that a health institution should justify one of the six conditions [66].
Art. 7 [67]	"Conditions for consent" [67]	Consent has to be given before retrieval of data. A patient has to be given the opportunity to revoke consent.
Art.17 [68]	"Right to erasure/Right to be forgotten" [68]	A patient has the rights to request and erasure of all their health data from the data processor.
Art.32 [69]	"Security of processing" [69]	Peronal data has to be encrypted, pseudonymized or anonymized when data is processed.

Table 2: Overview of GDPR laws relevant for Healthcare

4 Methododology and Tools

4.1 Design Science Research Methodology

This thesis follows an adopted approach of the design science research methodology (DSRM) for information systems [70]. It is an established research methodology that provides guidelines for evaluation and iteration for research projects. DSRM focuses on the development and performance aspects of a designed system and aims to improve the functional performance of the system [71]. The adopted research approach includes the following steps: (1) problem identification, (2) define solution objectives, (3) design and development, (4) demonstration and (5) communication [70].

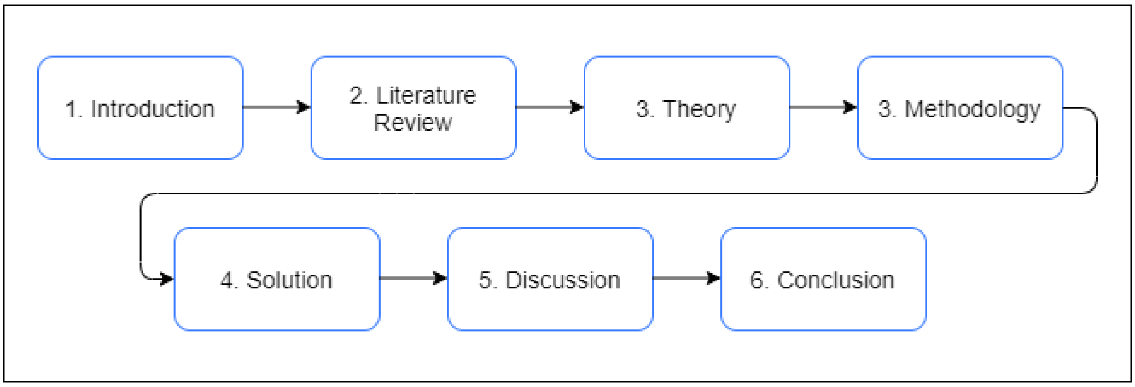


Figure 7: This is our design science research publication schema [71]

4.2 Prototype Development and Test Network

This section includes a description of the tools that have been used to develop and verify our conceptual design.

4.2.1 Remix IDE

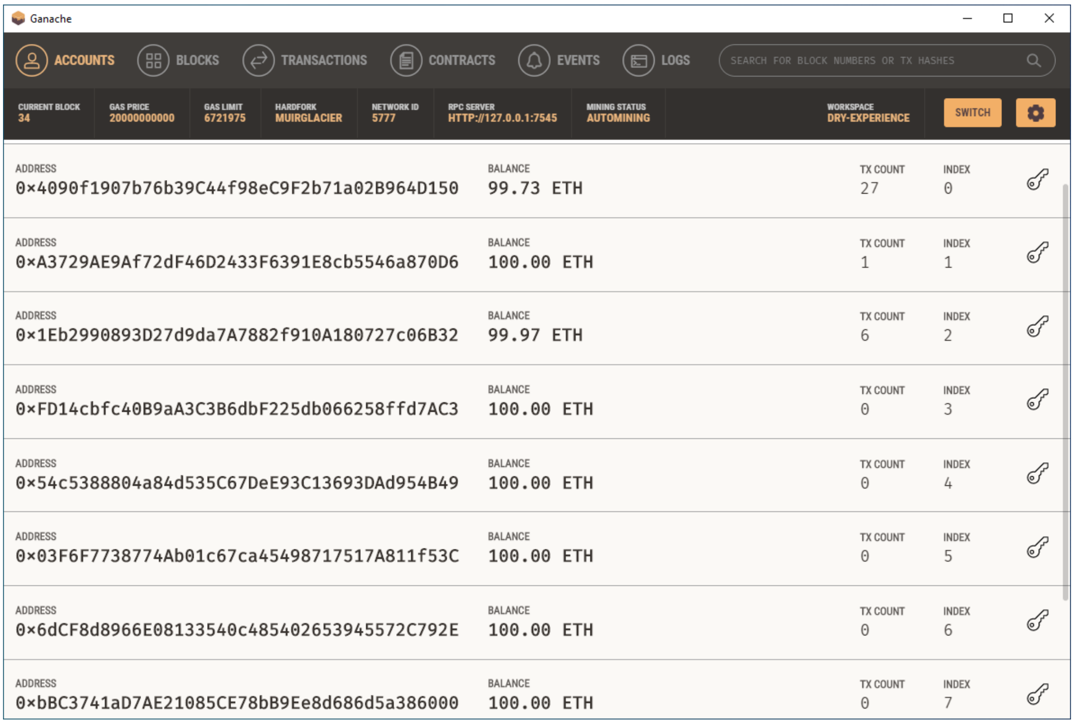
Remix is an open-source and web-based integrated development environment (IDE) for writing smart contracts. It is a tool that enables the developer to write and compile Solidity code straight from the browser [72]. Solidity is a high-level programming language for developing smart contracts, and it is influenced by other programming languages such as C++, JavaScript and Python [73].

Remix IDE is easy to access and has no downloading requirements. Remix IDE can be accessed directly via the web browser by visiting https://remix.ethereum.org/. The user will be presented with a code editor and different tools for compiling, running and debugging smart contracts [72].

4.2.2 Ganache

Ganache is a test network for blockchain, and it is a tool that lets users create a virtual blockchain locally and test smart contracts [74]. Ganache simulates features of a real Ethereum network, but there is no mining in Ganache. Thus, the transactions can be confirmed and smart contracts can be deployed instantaneously. Ganache comes with ten default Ethereum addresses, each with its private key and pre-loaded with 100 simulated Ether [75].

Ganache comes in two versions, a command-line interface (CLI) and an application with a user interface (UI). In our research, we chose to use the UI version of Ganache because it is easier to set up and is more user-friendly [74].



CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
34	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	DRY-EXPERIENCE

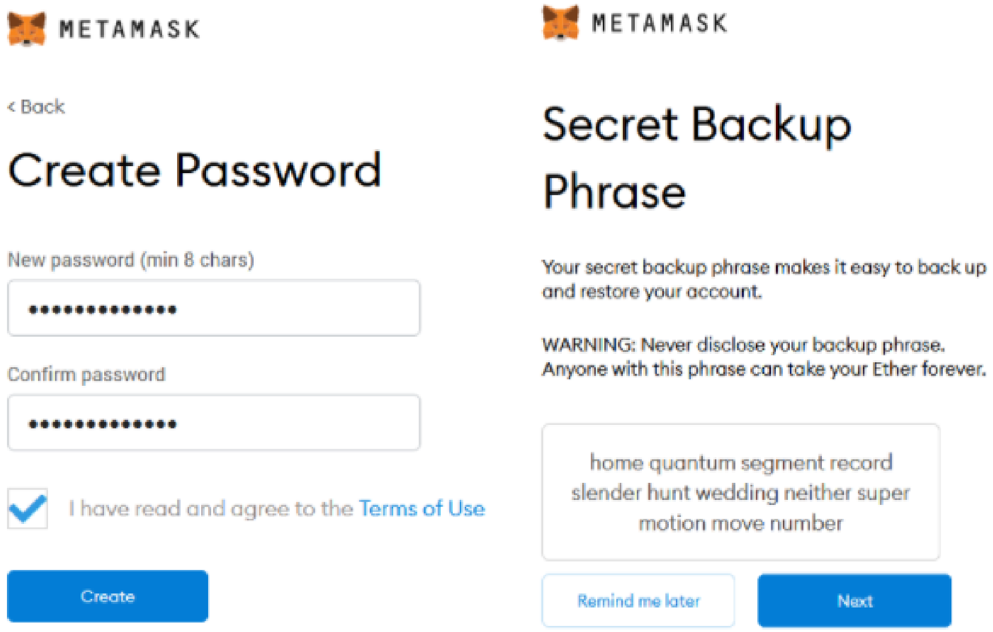
ADDRESS	BALANCE	TX COUNT	INDEX
0×4090f1907b76b39C44f98eC9F2b71a02B964D150	99.73 ETH	27	0
0×A3729AE9Af72dF46D2433F6391E8cb5546a870D6	100.00 ETH	1	1
0×1Eb2990893D27d9da7A7882f910A180727c06B32	99.97 ETH	6	2
0×FD14cbfc40B9aA3C3B6dbF225db066258ffd7AC3	100.00 ETH	0	3
0×54c5388804a84d535C67DeE93C13693DAd954B49	100.00 ETH	0	4
0×03F6F7738774Ab01c67ca45498717517A811f53C	100.00 ETH	0	5
0×6dCF8d8966E08133540c4854026539455 72C792E	100.00 ETH	0	6
0×bBC3741aD7AE21085CE78bB9Ee8d686d5a386000	100.00 ETH	0	7

Figure 8: Overview of Ganache

When running the UI version of Ganache, there should be a screen like this to see the different addresses with the simulated Ether. Ganache allows you to have your own virtual Ethereum node running, and you can connect it to a web browser wallet like MetaMask.

4.2.3 Connecting Metamask to Ganache

MetaMask is a browser extension that allows you to connect to an Ethereum node remotely and enables you to manage your Ethereum wallet through the browser [76]. When installing MetaMask, initially, there are terms and conditions that need to be accepted, and then you have to create your password. This password will be used to encrypt the wallets and is required every time MetaMask is accessed.



New password (min 8 chars)

••••••••••••••

Confirm password

••••••••••••••

☑ I have read and agree to the Terms of Use

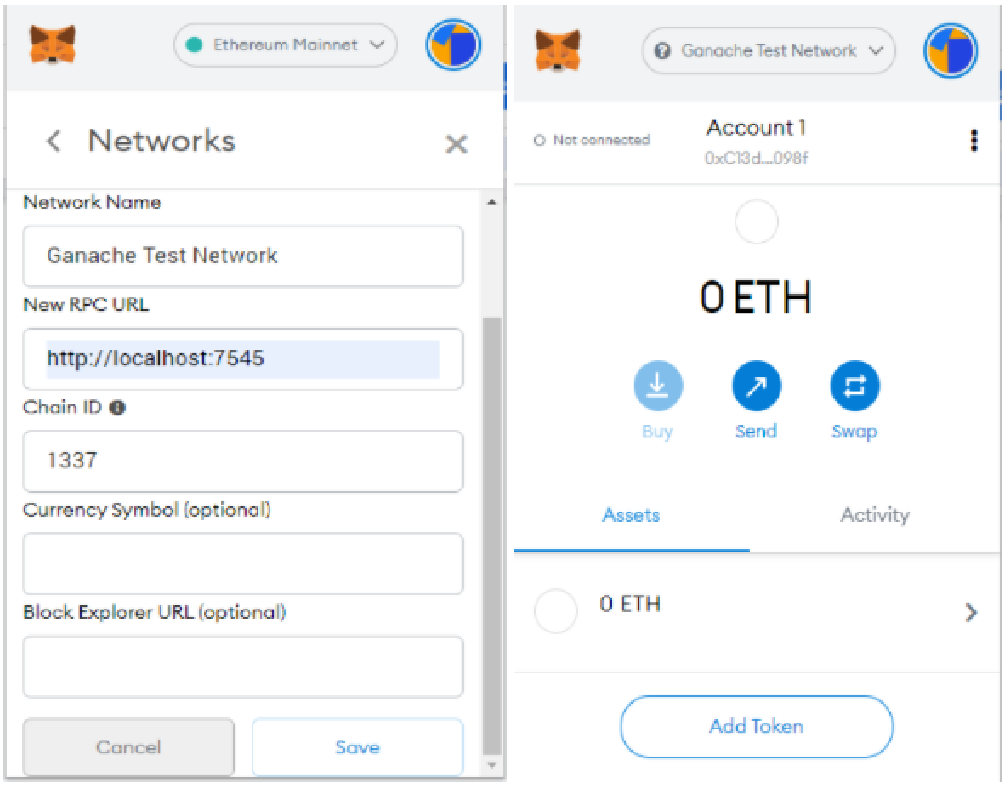
Your secret backup phrase makes it easy to back up and restore your account.

WARNING: Never disclose your backup phrase. Anyone with this phrase can take your Ether forever.

home quantum segment record slender hunt wedding neither super motion move number

Figure 9: Creating an account in MetaMask

After creating a password, you will be provided a seed phrase used to recover your account. This seed phrase is unique and consists of 12 words. It is essential to write this seed phrase down and store it in a safe place because it is the only way of accessing the wallet without a password. Once this process is done, it is time to connect to Ganache. By default, Ganache uses a certain IP address and a port, 127.0.0.1:7545 or localhost:7545. This IP address and port is used to connect MetaMask to Ganache.



Networks

Network Name

Ganache Test Network

New RPC URL

http://localhost:7545

Chain ID

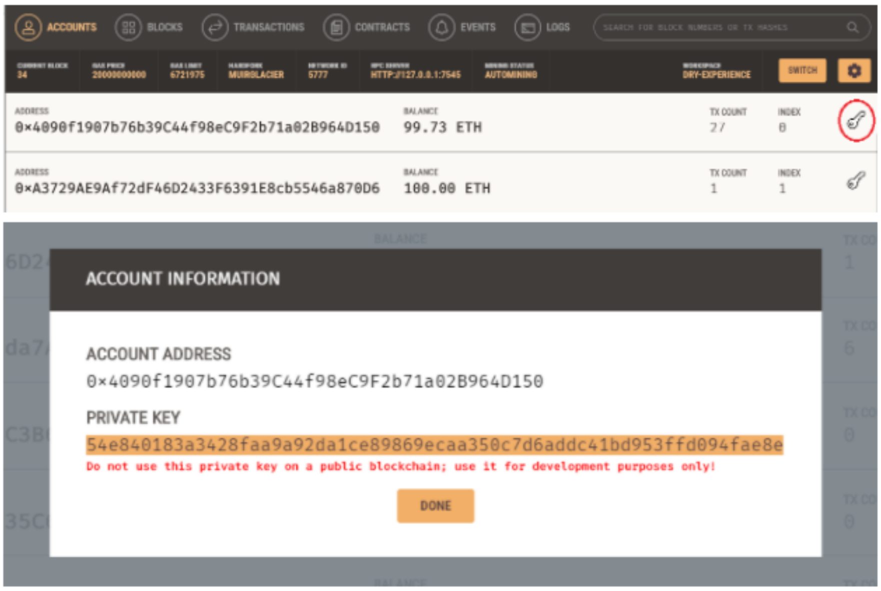
1337

Currency Symbol (optional)

Block Explorer URL (optional)

Figure 10: Connecting MetaMask to Ganache

After you are connected to the Ganache test network, you can import the simulated addresses in Ganache to MetaMask by using their private keys, and then copy this private key and paste it into MetaMask under “Import Account”, and your test network is ready.



CURRENT BLOCK 34 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE DRY-EXPERIENCE

ADDRESS	BALANCE	TX COUNT	INDEX
0×4090f1907b76b39C44f98eC9F2b71a02B964D150	99.73 ETH	27	0
0×A3729AE9Af72dF46D2433F6391E8cb5546a870D6	100.00 ETH	1	1

ACCOUNT INFORMATION

ACCOUNT ADDRESS
0×4090f1907b76b39C44f98eC9F2b71a02B964D150

PRIVATE KEY
54e840183a3428faa9a92da1ce89869ecaa350c7d6addc41bd953ffd094fae8e
Do not use this private key on a public blockchain; use it for development purposes only!

Figure 11: The private key of the simulated address

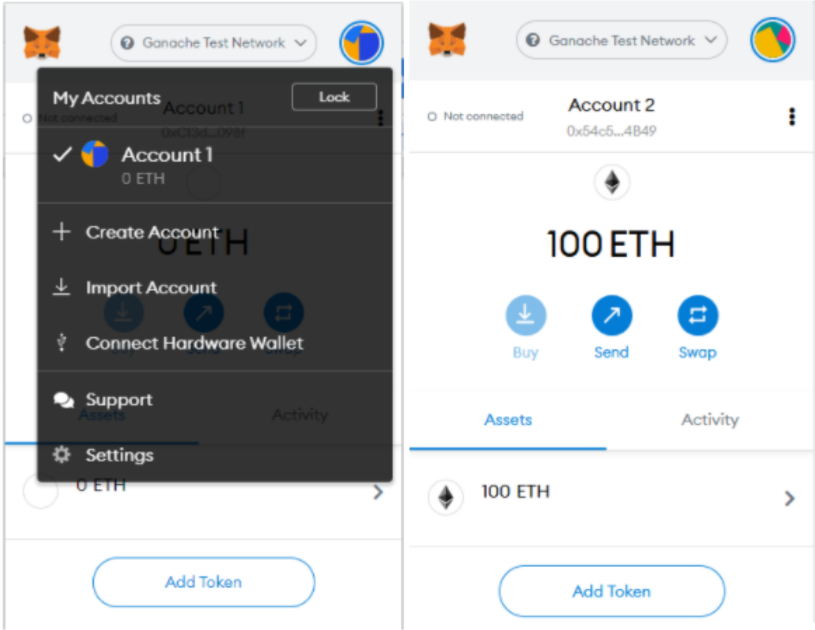


Figure 12: Importing an account to MetaMask from Ganache

5 Solution

5.1 Conceptual Design

Our conceptual design consists of three participants, i.e. a doctor, a patient, and a data host that stores patient health data. Communication is done through Blockchain and directly between participants. Inside our Blockchain network, we have several smart contracts that have specific roles. The data host will never be trusted; it is only an off-chain storage solution since storing extensive data in the Blockchain is not feasible. We assume that the health data is already stored at the off-chain storage by the patient. The Figure 13 illustrates the communication between participants with smart contracts through Blockchain of an Etherium network.

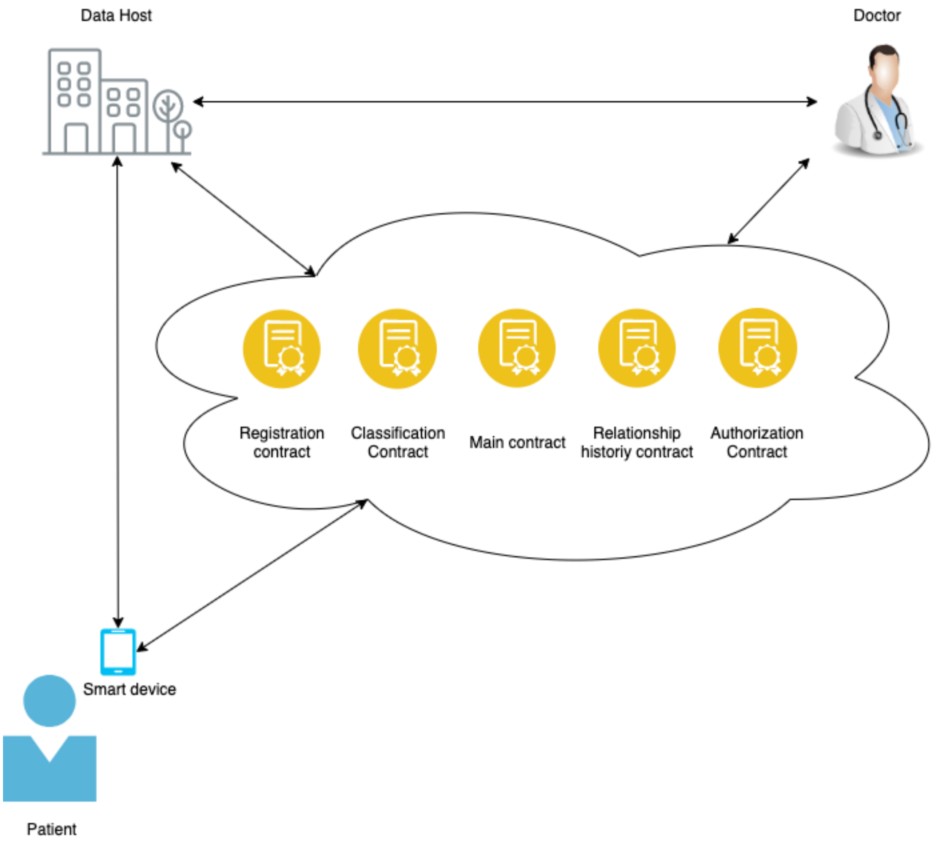


Figure 13: Overview of our conceptual design

Registration Contract: This contract is responsible for maintaining user registration. As an example, a relationship is created between the patient and the doctor when registering a doctor, while the Main Contract and the Relationship History Contract are created for their relation. Additionally, the Registration Contract validates nodes and registers the new node into the permissioned consortium Blockchain.

Classification Contract: This contract classifies the different levels of nodes in the system, such as patients, doctors, pharmacies, or third parties. This contract can validate if a node is already registered in the system to prevent double registration.

Main Contract: The Main Contract is responsible for tracking the data that the data host stores for patients and is created when a new relationship is formed between the two nodes. It also contains information needed to find specific data, as it can be seen as a pointer to a specific dataset at the off-chain storage. A Main Contract includes a query link and a hash of the encrypted data. The address of this Main Contract is stored inside the Relationship History Contract.

Relationship History Contract: This contract maintains the relationship histories of nodes and is created during the registration process. It provides a comprehensive list of previous and current healthcare relationships for the patient. Most importantly, the consent signed by the patient is stored inside the Relationship History Contract.

Authorization Contract: This contract specifies the different permissions given to each participant. At the same time, it determines the authorization a specific doctor has for a patient. The Authorization Contract stores the address of the Relationship History Contract.

Eth_call is used for easy interactions between contracts. The reason for using eth_calls is to execute and receive messages instantly without creating a standard Blockchain transaction [77].

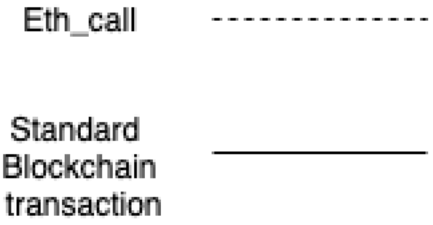


Figure 14: Eth_call and standard Blockchain transaction

We use a combination of symmetric and asymmetric key encryption for storing and sending data in our design.

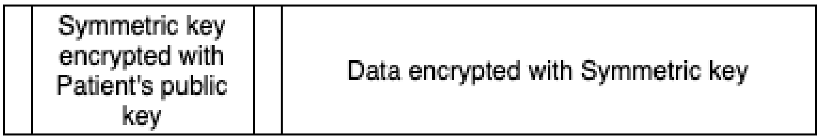

Symmetric key encrypted with Patient's public key
Data encrypted with Symmetric key

Figure 15: Combination of symmetric and asymmetric key encryption

Patient health data will be stored as different dataset, with each dataset encrypted with different symmetric keys. The symmetric keys will further be encrypted with the patient's public key. This ensures that only the patient can decrypt data, and the untrusted data host is only storing the encrypted data.

5.1.1 Implementation of Blockchain in our Design

The chosen type of Blockchain for our conceptual design is a permissioned consortium Blockchain. An option was to use a permissionless public Blockchain to fully benefit from a decentralized network, however, there are drawbacks to permissionless Blockchain that would not fulfill the needs of our design.

First of all, making the design a permissionless public Blockchain makes the network open for everyone to join. Everyone after joining can download a record of the entire Blockchain, and data inside Blockchain would be viewable for the public. There are no sensitive data stored inside Blockchain, even though there would be no access control over who has the information about transactions between participants. Additionally, there is no governance in public Blockchain to set standards that suits specific needs in the system. Two of the most popular consensus algorithm for permissionless public Blockchain are PoW and PoS, and these are not suitable for our design.

Permissioned Blockchain is more suitable for "business" applications and has the characteristic that suits our design. A permissioned Blockchain is not open for everyone to join and can restrict access into the Blockchain network. This will result in better confidentiality because transactions are isolated from public access. Additionally, there is more management and control in a permissioned Blockchain, and the participants will have defined roles. There is an access control of who gets into the network, with administrative nodes deciding user roles and participation, and transaction validation is done between a set of known participants. The reason for not choosing a private Blockchain is because of its characteristics of being too centralized. Consortium Blockchain is partially decentralized as a small group of nodes from different organizations can participate in the consensus. Overall, a permissioned consortium Blockchain is best suited for our design, and the consensus algorithm that suits our design is Proof of Authority (PoA).

The participants' reputation is at stake as the participants are required to give their real identities. This results in more trust between participants in the network. As the network is not open to everyone, the use of PoW would not be favorable. The energy requirements and computational power would make it inefficient. Having participants stake their coins using PoS can result in miners having more to say in the network than the authorities that control the network, as they have more probability of getting selected to mine the next block. PBFT does not scale well because the nodes are constantly communicating with each other. While DBFT is a better solution for scalability, however, the number of validation nodes is restricted. Taking these factors into consideration, PoA is the consensus algorithm that would suit our conceptual design.

5.1.2 Trusted Parties

Party	**Trust**	**Comment**
Patient	Yes	N/A
Data host	No	Data host will always be untrusted and only used as data storage. Data stored here will always be encrypted.
Doctor/third party	Yes	Consent is only given if the patient trusts a doctor or a third party. If there is no trust, consent will not be signed.

Table 3: Trusted parties

5.2 Consent Management

Two scenarios are described to show how consent is managed in our conceptual design. The first scenario illustrates when a doctor requests access to a patient's health data. Second scenario references a situation where a patient revokes the doctor's access after giving consent to the requested dataset.

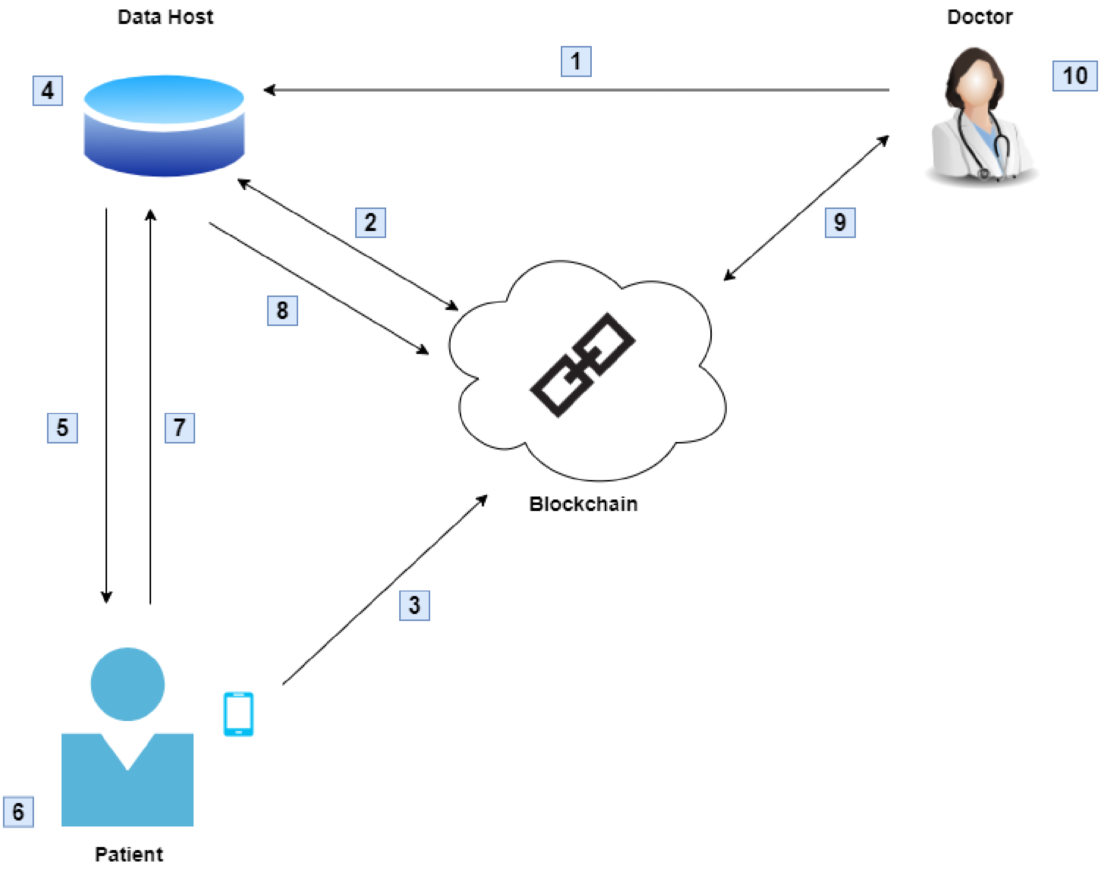


Figure 16: Consent Management

Requesting Access

1. A doctor sends a request to the data host regarding access to the health data of a patient. The doctor signs the request with a digital signature so that the data host can confirm the requester's identity.

2. The data host checks if the doctor is registered in the system.
   - If no, the doctor has to be registered. To register a new node as a doctor, several smart contracts will be used and a relationship between the doctor and the patient is created, thereby creating Relationship History Contract and Main Contract.
   - If yes, the request is added as a transaction as shown in Figure 17.
3. The patient is notified on his or her smart device that someone is requesting access to their data, and we assume that the patient signs the consent through a smart contract in Blockchain.
4. The data host gets a notification that the consent has been signed, and the data host takes a copy of the requested health data.
5. The data host then sends the requested encrypted dataset directly to the patient.
6. The patient will then decrypt the secret key with their private key and then use the secret key to decrypt the actual data. The data will then be encrypted with a new secret key, and this secret key will be encrypted with the doctor's public key. This ensures that only the doctor can access the requested data.
7. The patient returns the newly encrypted data to the data host.
8. The data host creates a query link with a copy of the requested health data. The data host sends a Blockchain transaction to the Main Contract with the query link of the new encrypted data and a hash of said encrypted data.
9. The doctor can now retrieve the query link through the Main Contract inside the Blockchain.
10. The doctor can now access the data through the query link using their private key.

Adding a Request

We assume that the process of registering a node as a doctor is done mentioned in step 2 in Figure 16. The Authorization Contract has the address of the Relationship History Contract. Additionally, we use the Authorization Contract to verify if the doctor has permission to request data.

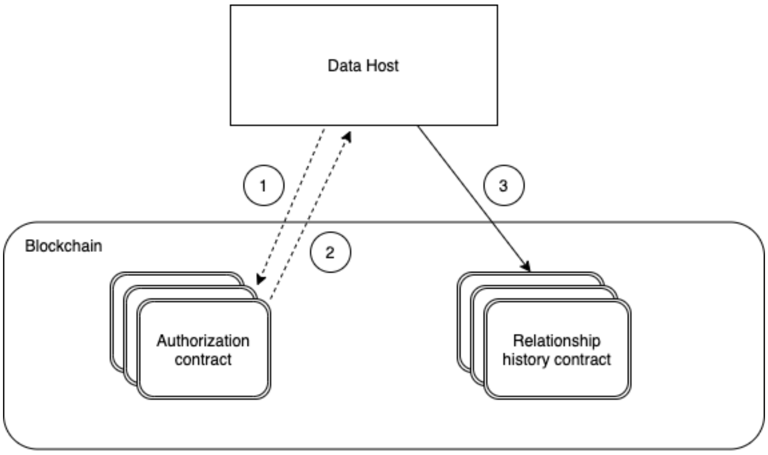


Figure 17: Adding a request

1. For adding a request, we send an eth_call to the Authorization Contract to verify if the doctor has permission to access the patient's data.
2. The Authorization Contract sends a return to the data host with permission and the address of the Relationship History Contract. We assume the doctor has permission.
3. To register a request, the data host adds the request as a transaction to the Blockchain.

Storing Consent and Adding Record

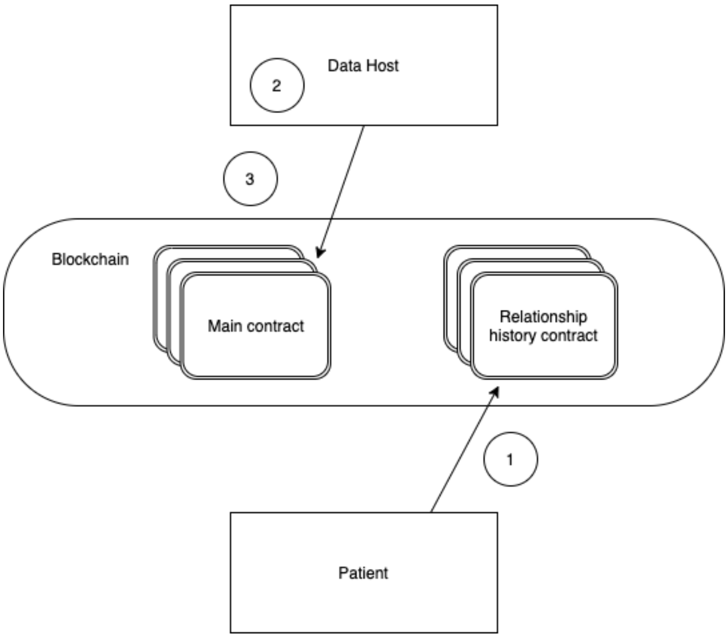


Figure 18: Store consent and add record to Blockchain

1. After the provider has added the request as a transaction to the block, the patient gets a notification on his smart device about the request. The Patient can either accept or reject the request through a smart contract. We assume the patient accepts the request and adds another transaction to the Blockchain, as shown in step 3 in Figure 16.

2. Once a transaction is registered in the Relationship History Contract, a notification is sent to the data host about the patient's signed consent. When the signed consent from the patient is received, the data host creates a query link with the requested and newly encrypted data.

3. The data host executes a Blockchain transaction to the Main Contract with the query link and a hash of the encrypted data. This is shown in step 8 in Figure 16.

5.3 Revoking Given Permission

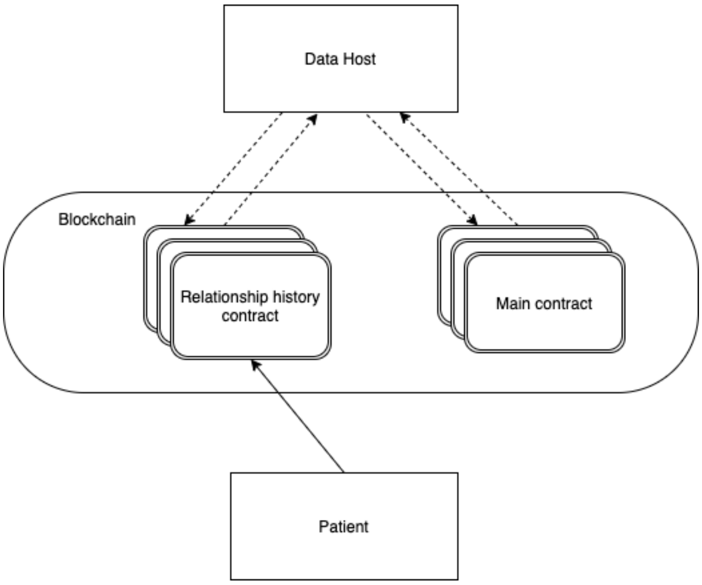


Figure 19: Revoking Given Permission

1. The patient wants to revoke the access given to the doctor. The patient can do this through Blockchain, and a notification will be sent to the data host.

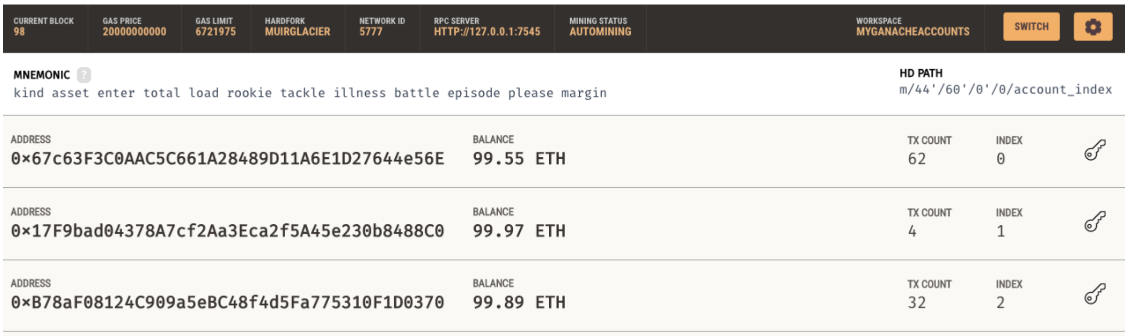
2. The data host receives a notification about the revoked access. The data host then sends an eth_call to the Relationship History Contract and asks for the address of the Main Contract.

3. The data host receives the Main Contract address, thereafter sends an eth_call to the Main Contract, and afterwards asks about the pointer to where that data is stored and which query link needs to be removed. The Main Contract returns with the pointer, and the data host removes the data by deleting that query link.

5.4 Verification of Consent Management with Smart Contracts

In this section, we demonstrate how a patient can accept a consent request through a transaction in Blockchain using a smart contract.

First and foremost, we have to open Ganache to start our local test network. This facilitates the connection between Metamask and our test network, and it can get the accounts ready for transactions. Figure 20 and Figure 21 show how the accounts from Ganache already added to Metamask and readily available to send and receive tokens.



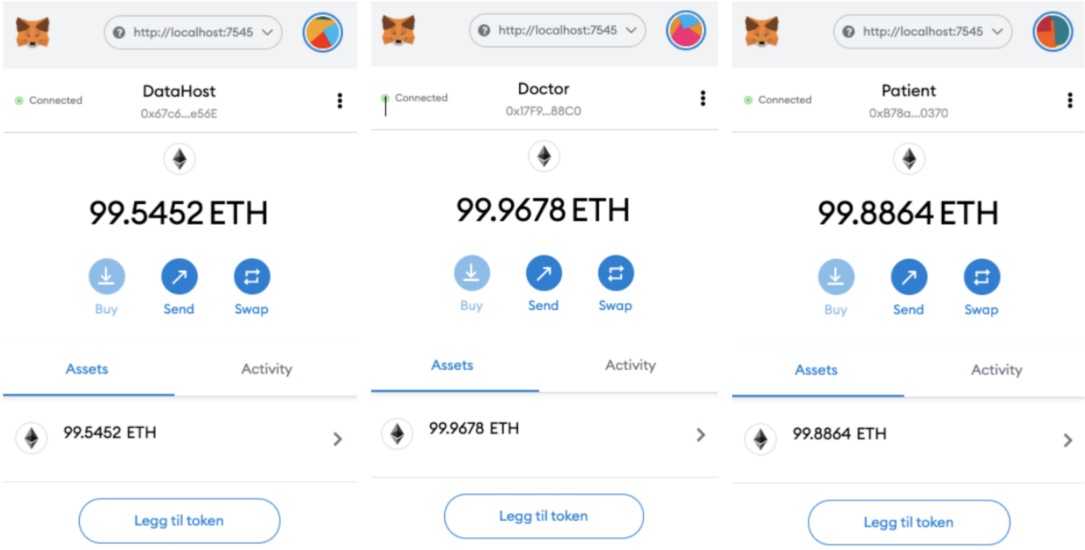
CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
98	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	MYGANACHEACCOUNTS

MNEMONIC
kind asset enter total load rookie tackle illness battle episode please margin

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0×67c63F3C0AAC5C661A28489D11A6E1D27644e56E	99.55 ETH	62	0
0×17F9bad04378A7cf2Aa3Eca2f5A45e230b8488C0	99.97 ETH	4	1
0×B78aF08124C909a5eBC48f4d5Fa775310F1D0370	99.89 ETH	32	2

Figure 20: Our Ganache test network



DataHost 0x67c6...e56E 99.5452 ETH

Doctor 0x17F9...88C0 99.9678 ETH

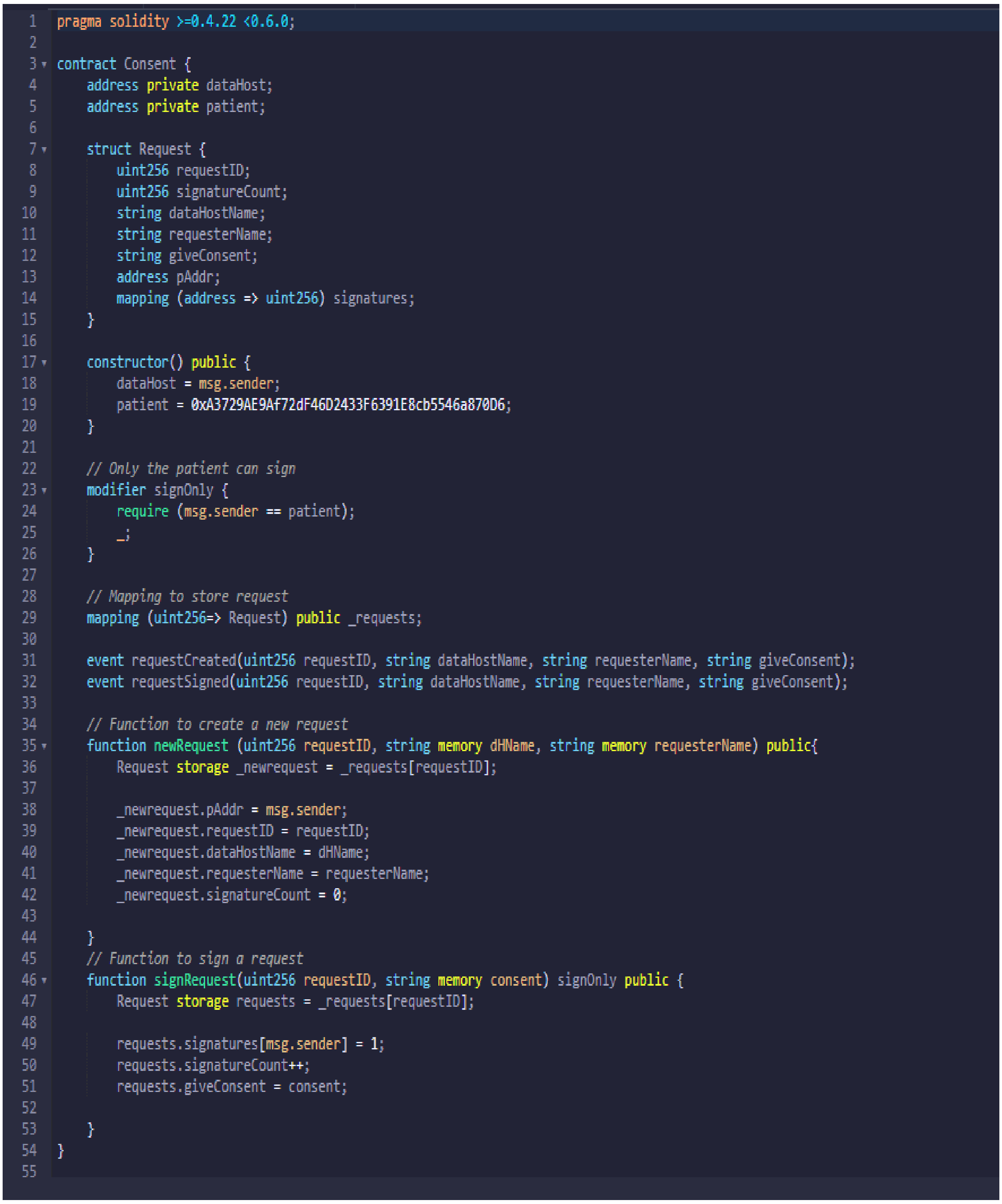
Patient 0xB78a...0370 99.8864 ETH

Figure 21: Participants in our test network

The compiled code in Remix IDE enables deployment of the contract. We choose Injected Web3 environment for deployment and select which account the smart contract will be deployed by. In our case, the data host will make a transaction and deploy the contract. It is important to test the contracts in a test network first to see if everything is working properly before deployment in real world network.

5.4.1 Giving Consent

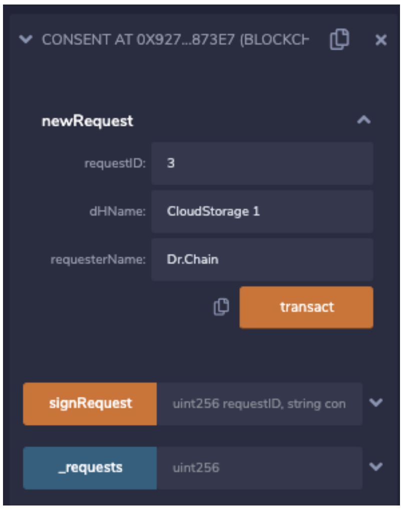
This smart contract enables the data host to send a request through the Blockchain, and allows a patient to accept and sign consent request. The data host can also check to see if a patient has given consent or not. We chose to include a request ID, the data host's name and the doctor's name in this smart contract. In line 19 of the code, we have inserted the patient's Ethereum address from Metamask, allowing only the intended patient to sign the request.



```
pragma solidity >=0.4.22 <0.6.0;

contract Consent {
    address private dataHost;
    address private patient;

    struct Request {
        uint256 requestID;
        uint256 signatureCount;
        string dataHostName;
        string requesterName;
        string giveConsent;
        address pAddr;
        mapping (address => uint256) signatures;
    }

    constructor() public {
        dataHost = msg.sender;
        patient = 0xA3729AE9Af72dF46D2433F6391E8cb5546a870D6;
    }

    // Only the patient can sign
    modifier signOnly {
        require (msg.sender == patient);
        _;
    }

    // Mapping to store request
    mapping (uint256=> Request) public _requests;

    event requestCreated(uint256 requestID, string dataHostName, string requesterName, string giveConsent);
    event requestSigned(uint256 requestID, string dataHostName, string requesterName, string giveConsent);

    // Function to create a new request
    function newRequest (uint256 requestID, string memory dHName, string memory requesterName) public{
        Request storage _newrequest = _requests[requestID];

        _newrequest.pAddr = msg.sender;
        _newrequest.requestID = requestID;
        _newrequest.dataHostName = dHName;
        _newrequest.requesterName = requesterName;
        _newrequest.signatureCount = 0;

    }
    // Function to sign a request
    function signRequest(uint256 requestID, string memory consent) signOnly public {
        Request storage requests = _requests[requestID];

        requests.signatures[msg.sender] = 1;
        requests.signatureCount++;
        requests.giveConsent = consent;

    }
}

```

Figure 22: Smart contract for signing consent

In our conceptual design, we described that the patient has to accept consent first in order for a doctor to gain access to patient health data from the data host. This ensures that the patient is in control of their own personal data and is given the choice of accepting or rejecting the doctor's request. In step 3 shown in Figure 17, the data host adds the request to the contract through a Blockchain transaction.



newRequest

requestID: 3

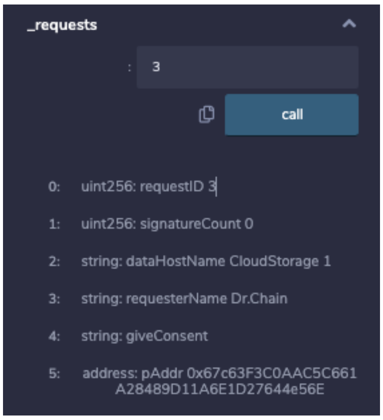
dHName: CloudStorage 1

requesterName: Dr.Chain

signRequest uint256 requestID, string con

_requests uint256

Figure 23: Register a request

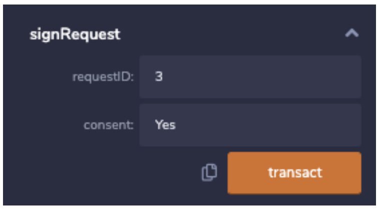


```
: 3

0:  uint256: requestID 3
1:  uint256: signatureCount 0
2:  string: dataHostName CloudStorage 1
3:  string: requesterName Dr.Chain
4:  string: giveConsent
5:  address: pAddr 0x67c63F3C0AAC5C661
        A28489D11A6E1D27644e56E
```

Figure 24: Call the contract

To give the patient more details about what data is requested, we can add more fields in the newRequest function that the data host can fill out, such as type of health data and date of request. This allows the patient to know what type of health data and the date of request they are giving consent to. To see the request, we can call the request using its ID, shown in Figure 24.

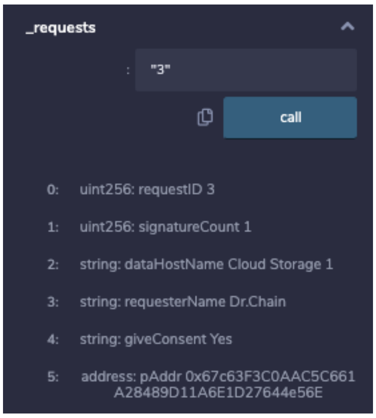
The patient then gets a notification about the request and can choose to either accept or reject it. We assume the patient accepts the request. In order to give consent, the patient needs to perform a Blockchain transaction to sign the request as shown in Figure 25.



requestID: 3

consent: Yes

Figure 25: Signing request



: "3"

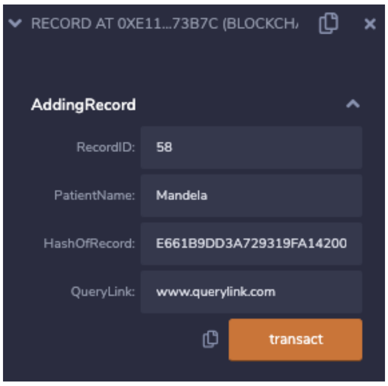
```
0:  uint256: requestID 3
1:  uint256: signatureCount 1
2:  string: dataHostName Cloud Storage 1
3:  string: requesterName Dr.Chain
4:  string: giveConsent Yes
5:  address: pAddr 0x67c63F3C0AAC5C661
         A28489D11A6E1D27644e56E
```

Figure 26: Signed contract

After the patient has accepted the request and given consent by signing the corresponding smart contract, the data host can check the request status by calling the signed contract. In comparison to Figure 24, we can see in Figure 26 that the signature count has increased due to the patient signing the contract. Once the data host has registered that the request is signed, they start the process of copying the requested data from their database and send it for the decryption and encryption process done by the patient (step 6 in Figure 16).

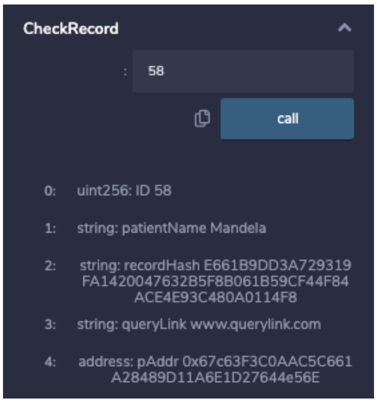
5.4.2 Check for Added Records

Once the data host receives the newly encrypted data, they will add a hash of the encrypted data and the query link into the Main Contract. The doctor can thereafter access the link through Blockchain, as shown in step 9 in Figure 16.



RecordID: 58
PatientName: Mandela
HashOfRecord: E661B9DD3A729319FA14200
QueryLink: www.querylink.com

Figure 27: Adding record



: 58

```
0:   uint256: ID 58
1:   string: patientName Mandela
2:   string: recordHash E661B9DD3A729319
     FA1420047632B5F8B061B59CF44F84
     ACE4E93C480A0114F8
3:   string: queryLink www.querylink.com
4:   address: pAddr 0x67c63F3C0AAC5C661
     A28489D11A6E1D27644e56E
```

Figure 28: View Record

6 Discussion

6.1 How can consent management for sharing of eHealth data be realized with the use of Blockchain-based smart contracts?

The research question in our thesis is answered through our sub-objectives. The research work for this thesis has allowed us to propose a solution on how Blockchain-based smart contracts can be used to realize consent management for data sharing. The conceptual design consists of two different scenarios which show the process of signing and revoking consent by the patient. Our conceptual design shows that it is possible to realize a consent management for data sharing by using Blockchain, where a patient can give and revoke access to their health data. Regardless, an important question we have to ask ourselves is "*why use Blockchain?*". One could argue that most people find Blockchain complex and hard to understand. Additionally, it is challenging to develop a system without involving a trusted party. Another inconvenience with Blockchain is that it is irreversible, and by human errors, incorrect and false information is likely to be entered into the Blockchain. However, the ability to store transactions and information, and the ability to trace transactions to ensure transparency while being immutable, makes it an attractive platform not only for the healthcare sector, but also for other sectors.

A key component in our conceptual design is the encryption, and as proposed, we chose to use both symmetric and asymmetric encryption. The health data stored at the data host is encrypted with symmetric encryption, while the symmetric encryption key is itself encrypted using asymmetric encryption. This is because encrypting extensive data with asymmetric encryption takes far longer and is inefficient. We also assumed that every patient uses a smart device, and will therefore have limited processing power to use asymmetric encryption on more extensive data. Additionally, we chose to use asymmetric encryption to add another layer of security because it uses two different keys, i.e. a public key accessible to anyone is used to encrypt the symmetric encryption key, and a related private key that is never shared is vital to decrypt the symmetric encryption key. Asymmetric encryption helps in ensuring that only the intended recipient, i.e. the owner of the private asymmetric key, can decrypt the requested health data. By using a combination of both symmetric and asymmetric encryption, we improve the efficiency and security of our consent management at the same time.

6.2 How can the selected key requirements of GDPR be fulfilled with our proposed solution?

It is essential that the laws of GDPR are followed when healthcare embraces new technologies like Blockchain. Issues like misuse of patient health data and breaches leading to exposure of sensitive patient data happen because of the embracement of new technologies. This can result in reduced trust in healthcare institutions, and discourage or delay the adoption of necessary or beneficial new technologies. GDPR has strict rules regarding data processing when using Blockchain technology. Table 2 in chapter 3 summarizes the selected key requirements in focus in order to utilize Blockchain in healthcare. Our primarily objective is to realize consent management for data sharing through smart contracts. However, we realized that there are several other GDPR rules that could also be addressed in our design, with some still needed to be solved in order to properly facilitate the use of Blockchain in healthcare.

Art.6 is about the purpose of processing data, and this is achieved in our design. When a doctor or another third party requests patient health data, the request will be placed in the Blockchain. The patient is the one that has to give consent, and if consent is given for processing data, the first of six conditions within Art.6 will be fulfilled [66].

The ”Rights to be forgotten” is defined in Art.17, and this is one of the most challenging GDPR laws that may hold back the adoption of Blockchain technology in healthcare. The ”Rights to be forgotten” has raised the question of whether

Blockchain can be GDPR compliant at all, considering that the Blockchain immutability does not allow this by design. As most of the state-of-the art papers stated, we agree that the fundementals of Blockchain makes it challenging to be GDPR compliant. However, there are ways to use Blockchain and its features to further improve data integrity, ease consent management, and give patients control over their data. Our design solves the issues of the "Rights to be forgotten" by moving sensitive data required to be deleted at any time on behalf of the patient outside of the Blockchain. We store patient data in an off-chain storage, while consent, query links, and hash of encrypted data is recorded inside Blockchain by using smart contracts. This solution provides a transparent and immutable system using Blockchain, and at the same time separately storing patient data in off-chain storage that can be deleted at the patient's request. The process to fulfill the "Rights to be forgotten" is simplified for the patient, as they only need to send a request to the data host for any deletion. The query link created for the requested patient health data and shared with third parties can easily be deleted at a patient's request.

Art.32 of GDPR states that personal data must be encrypted, pseudonymized, or anonymized when data is processed. We accomplish this by always having data encrypted at the off-chain storage. The patient is the only one with the key to decrypt these data as we do not trust the data host in our conceptual design. If there is any need to decrypt the data, the patient is making the decision since they are the ones with the decryption keys.

6.2.1 Consent

Art.7 regarding that the patient has to give consent is the core of our conceptual design. This GDPR law is the one that drives our design. Our main objective was to use Blockchain technology to realize consent management for data sharing in eHealth, meant to manage consents regarding access to personal health data. Figure 16 illustrates step by step how consent management is achieved when a doctor request access to patient health data. The design is responsible for accepting a request, obtaining the patient's signed consent request, and forwarding the requested health data to the requester. Additionally, the integrity and versioning of a patient's signed consent request is maintained by the Blockchain. The right to withdraw consent at anytime is an important part of Art.7, and we have addressed this under section 5.3 in chapter 5. It should be as easy to withdraw consent as it is to give consent, and this is achieved in our solution. The patient sends a request into the Blockchain about revoking a given permission, and it is the data host's responsibility to fulfill this request through updates of the corresponding smart contracts.

6.3 Limitations in our Conceptual Design

As our design only uses the data host as an untrusted off-chain storage, the stored health data is always encrypted. When a third party, such as a doctor, is requesting access to specific data, the data host must search for the requested data through encrypted datasets. This implies that searchable encryption should be in place in order for the data host to find the requested dataset. Searchable encryption allows searching for specific data in an encrypted database without having to decrypt the data first. As such, searchable encryption is used to protect personal and sensitive data from an untrusted data host, and at the same time allows the data host to search through encrypted data. Note that it allows the data host to search encrypted data without leaking any information in plain text. It has been in development for a long time, and there are different methods and solutions to achieve searchable encryption [78, 79, 80].

Another limitation to our conceptual design is encrypting and decrypting locally on a patient's smart device. The problem here is that not everyone has a smartphone or similar smart device that can be used to do this process. Furthermore, another problem is the usually limited processing power of the portable smart devices. If the requested data is too large, the patient will not have enough processing power to encrypt and decrypt this data. Even though the data host has more processing power, we choose to execute this task locally on a patient's smart device because of privacy and trust issues. A possible solution in the future can be to move the encryption and decryption processes to a trusted third party.

7 Conclusion

The aim of our thesis was to see if it was possible to use Blockchain-based smart contracts to achieve a consent management for sharing eHealth data, which is realized through completing our sub-objectives. Additionally, selected key requirements of GDPR are fulfilled through the proposed design. Blockchain has many qualities like traceability and immutability that can benefit the healthcare industry. We touched upon different types of Blockchain and consensus algorithms, and assessed which type was the best fit for our design. As a result, our conceptual design is best suited as a permissioned consortium Blockchain with Proof of Authority (PoA) as the consensus algorithm.

The proposed solution seeks to improve a patient's control over their own health data, and this thesis shows that there is potential value in implementing proposed design. The solution was proposed as a result of the literature review conducted in the thesis. Through the use of various smart contracts, we created a design where the patient is in control of and therefore responsible for their own data by letting them decide who they want to trust and give consent to. The design employ scenarios to illustrate how a patient can give and revoke access through smart contracts. We set up a test network and deployed a simple Ethereum smart contract that demonstrates how consent management can be applied by a patient signing a request. The use of encryption throughout our conceptual design demonstrates that we prioritize the patient by making it a patient-centered model and go to great lengths in order to not place patient's privacy at risk. Our current version of the proposed conceptual design contains limitations and leaves space for future work.

7.1 Future Work

Future research provided addresses several aspects of the emerged limitations of the conceptual design. Searchable encryption has to be implemented for our design to be fully functional. Furthermore, decrypting and encrypting data locally at the patient level has limitations and is therefore recommended to be moved to a trusted party by the patient. As our design achieves data sharing with consent management, future work can look at considerations regarding implementation of functions to allow third parties to update patient health data and further elevate the capabilities of the design.

The limitations in relation to lack of smart devices in patients' possessions should be solvable with time, as smart devices of any kind, including everything from home assistants to wearables, becomes more integrated in our daily lives. To substantiate this claim, we make the assumption that processing power of these smart devices improves with each iteration and generation. In other words, it would be wise to move the encryption and decryption processes to trusted parties as mentioned in discussion until the processing power of smart devices catches up with the necessary requirements to do local encryption and decryption of extensive data.

It should be noted that the performance evaluation of both the chosen consensus algorithm and the deployment of smart contracts are not covered in this thesis. This limitation of the thesis should be addressed in order to implement the design for future work.

References

[1] Xiao Yue - Huiju Wang - Dawei Jin - Mingqiang Li - Wei Jiang. *Healthcare Data Gateways: Found Healthcare Intelligence on Blockchain with Novel Privacy Risk Control.* 2016. URL: https://pubmed.ncbi.nlm.nih.gov/27565509/ [Last Accessed June 1, 2021].

[2] Satoshi Nakamotoo. *Bitcoin: A Peer-to-Peer Electronic Cash System.* 2008. [Last Accessed June 1, 2021].

[3] Anton Hasselgren - Paul Kengfai Wan - Margareth Horn - Katina Kralevska - Danilo Gligoroski - Arild Faxvaag. *GDPR Compliance for Blockchain Applications in Healthcare.* 2020. URL: https://www.researchgate.net/publication/344410471_GDPR_Compliance_for_Blockchain_Applications_in_Healthcare [Last Accessed June 1, 2021].

[4] Asma K. *A Blockchain-Based Smart Contract System for Healthcare Management.* 2020. URL: https://www.researchgate.net/publication/338380336_A_Blockchain-Based_Smart_Contract_System_for_Healthcare_Management [Last Accessed June 1, 2021].

[5] BuiltIn. *What Is Blockchain Technology? How Does It Work?* URL: https://builtin.com/blockchain [Last Accessed June 1, 2021].

[6] Jesse Yli-Huumo et al. *Where Is Current Research on Blockchain Technology?—A Systematic Review.* 2016. URL: https://www.researchgate.net/publication/308877750_Where_Is_Current_Research_on_Blockchain_Technology-A_Systematic_Review [Last Accessed June 1, 2021].

[7] Aurelie Bayle et al. *When Blockchain Meets the Right to Be Forgotten: Technology versus Law in the Healthcare Industry.* 2018. URL: https://ieeexplore.ieee.org/document/8609693 [Last Accessed June 1, 2021].

[8] Shermin Voshmgir. *What is Blockchain?* URL: https://blockchainhub.net/blockchain-intro/ [Last Accessed June 1, 2021].

[9] Qusay H. Mahmoud Cornelius C. Agbo and J. Mikael Eklund. *Blockchain Technology in Healthcare: A Systematic Review.* 2019. URL: https://www.mdpi.com/2227-9032/7/2/56 [Last Accessed June 1, 2021].

[10] Marek Laskowshi Henry M.Kim. *A Perspective on Blockchain Smart Contracts: Reducing Uncertainty and Complexity in Value Exchange.* May 2017.

[11] G Eysenbach. "What is e-health?" In: *Journal of medical Internet research* 3.2 (2001). DOI: 10.2196/jmir.3.2.e20.

[12] Michael L. Glasser and Karen E. Peters. *E-health.* URL: https://www.britannica.com/science/e-health [Last Accessed June 1, 2021].

[13] Tsung Ting Kuo, Hugo Zavaleta Rojas, and Lucila Ohno Machado. *Comparison of blockchain platforms: a systematic review and healthcare examples.* 2019. URL: https://academic.oup.com/jamia/article/26/5/462/5419321 [Last Accessed June 1, 2021].

[14] Prateek Pandey and Ratnesh Litoriya. *Implementing healthcare services on a large scale: Challenges and remedies based on blockchain technology.* Mar. 2020.

[15] William J.Gordon and Christian Catalini. *Blockchain technology for healthcare: Facilitating the transition to patient-driven interoperability.* June 2018.

[16] GDPR. *What is GDPR, the EU's new data protection law?* URL: https://gdpr.eu/what-is-gdpr [Last Accessed June 1, 2021].

[17] The World Financial Review. *How can personal data be misused?* URL: https://worldfinancialreview.com/how-can-personal-data-be-misused/ [Last Accessed June 1, 2021].

[18] Koosha Mohammad Hossein et al. *Blockchain-Based Privacy-Preserving Healthcare Architectur.* 2019. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8861857 [Last Accessed June 1, 2021].

[19] Xiaochen Zheng - Raghava Rao Mukkamala - Ravi Vatrapu - Joaqun Ordieres-Mere. *Blockchain-based Personal Health Data Sharing System Using Cloud Storage.* 2018. URL: https://ieeexplore.ieee.org/document/8531125 [Last Accessed June 1, 2021].

[20] Anton Hasselgren - Paul Kengfai Wan - Margareth Horn - Katina Kralevska - Danilo Gligoroski - Aril Faxvaag. *GDPR Compliance for blockchain applications in Healthcare.* 2020. URL: https://www.researchgate.net/

publication/344410471_GDPR_Compliance_for_Blockchain_Applications_in_Healthcare [Last Accessed June 1, 2021].

[21] Alain Giordanengo. *Possible Usages of Smart Contracts (Blockchain) in Healthcare and Why No One Is Using Them.* URL: https://munin.uit.no/bitstream/handle/10037/18149/article.pdf?sequence=4 [Last Accessed June 1, 2021].

[22] Hoai Luan Pham - Thi Hong Tran - Yasuhiko Nakashim. *A Secure Remote Healthcare System for Hospital Using Blockchain Smart Contract.* 2018. URL: https://ieeexplore.ieee.org/document/8644164 [Last Accessed June 1, 2021].

[23] Asaph Azaria - Ariel Ekblaw - Thiago Vieira and Andrew Lippman. *MedRec: Using Blockchain for Medical Data Access and Permission Management.* 2018. URL: https://ieeexplore.ieee.org/document/7573685 [Last Accessed June 1, 2021].

[24] Gaby G.Dagher et al. *Ancile Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology.* 2018. URL: https://www.sciencedirect.com/science/article/pii/S2210670717310685 [Last Accessed June 1, 2021].

[25] Kristen N. Griggs et al. *Healthcare Blockchain System Using Smart Contracts for Secure Automated Remote Patient Monitoring.* 2018. URL: https://pubmed.ncbi.nlm.nih.gov/29876661/ [Last Accessed June 1, 2021].

[26] A. Kritas - C. Ilioudis - A. Papanikolaou K. Rantos - G. Drosato and A. P. Filipidis. *A Blockchain-Based Platform for Consent Management of Personal Data Processing in the IoT Ecosystem.* 2019. URL: https://www.researchgate.net/publication/336649345_A_Blockchain-Based_Platform_for_Consent_Management_of_Personal_Data_Processing_in_the_IoT_Ecosystem [Last Accessed June 1, 2021].

[27] Stuart Haber and W. Scott Stornetta. *How To Time-Stamp a Digital Document.* 1991. URL: https://link.springer.com/content/pdf/10.1007/BF00196791.pdf [Last Accessed June 1, 2021].

[28] Zibin Zheng - Shaoan Xie - Hongning Dai - Xiangping Chen - Huaimin Wang. *An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends.* 2017. URL: https://www.researchgate.net/publication/318131748_An_Overview_of_Blockchain_Technology_Architecture_Consensus_and_Future_Trends [Last Accessed June 1, 2021].

[29] Isitan Gorkey - Chakir El Moussaoui - Vincent Wijdeveld - Erik Sennema. *Comparative Study of Byzantine Fault Tolerant Consensus Algorithms on Permissioned Blockchains.* 2020. URL: https://www.researchgate.net/publication/330880555_Consensus_Algorithms_in_Blockchain_Comparative_Analysis_Challenges_and_Opportunities [Last Accessed June 1, 2021].

[30] Bit2me academy. *Bitcoin transactions, how do they work?* URL: https://academy.bit2me.com/en/bitcoin-transactions/ [Last Accessed June 1, 2021].

[31] 101Blockchains. *Permissioned Vs Permissionless Blockchains.* 2020. URL: https://101blockchains.com/permissioned-vs-permissionless-blockchains/ [Last Accessed June 1, 2021].

[32] Marko Vukolic. *Rethinking Permissioned Blockchains.* 2017. URL: http://vukolic.com/rethinking-permissioned-blockchains-BCC2017.pdf [Last Accessed June 1, 2021].

[33] Amani Altarawneh and Anthony Skjellum. *The Security Ingredients for Correct and Byzantine Fault-tolerant Blockchain Consensus Algorithms.* 2020. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=%5C&arnumber=9297326 [Last Accessed June 1, 2021].

[34] Jake Frankenfield. *Hash.* URL: https://www.investopedia.com/terms/h/hash.asp [Last Accessed June 1, 2021].

[35] Ying-Chang Liang. *Blockchain for Dynamic Spectrum Management.* International series of monographs on physics. Springer, 2019.

[36] Jake Frankenfield. *Merkle Root (Cryptocurrency).* URL: https://www.investopedia.com/terms/m/merkle-root-cryptocurrency.asp [Last Accessed June 1, 2021].

[37] Anirudh VK. *How Bitcoin Solved The Byzantine Generals' Problem.* URL: https://analyticsindiamag.com/how-bitcoin-solved-the-byzantine-generals-problem/ [Last Accessed June 1, 2021].

[38] Jake Frankenfield. *Proof of Stake (PoS).* URL: https://www.investopedia.com/terms/p/proof-stake-pos.asp [Last Accessed June 1, 2021].

[39] Binance Academy. *Proof of Authority Explained.* URL: https://academy.binance.com/en/articles/proof-of-authority-explained [Last Accessed June 1, 2021].

[40] Kate Ashford and Benjamin Curry. *What Is Bitcoin And How Does It Work?* URL: https://www.forbes.com/advisor/investing/what-is-bitcoin/ [Last Accessed June 1, 2021].

[41] Michael Crosby - Nachiappan - Pradan Pattanayak - Sanjeev Verma - Vignesh Kalyanaraman. *BlockChain Technology: Beyond Bitcoin.* 2016. URL: https://j2-capital.com/wp-content/uploads/2017/11/AIR-2016-Blockchain.pdf [Last Accessed June 1, 2021].

[42] Nathan Reiff. *How does a block chain prevent double-spending of Bitcoins?* URL: https://www.investopedia.com/ask/answers/061915/how-does-block-chain-prevent-doublespending-bitcoins.asp [Last Accessed June 1, 2021].

[43] Travis Patron. *What's the Big Idea Behind Ethereum's World Computer?* 2016. URL: https://www.coindesk.com/whats-big-idea-behind-ethereums-world-computer [Last Accessed June 1, 2021].

[44] Vitalik Buterin. *A Next-Generation Smart Contract and Decentralized Application Platform.* 2013. URL: https://ethereum.org/en/whitepaper/ [Last Accessed June 1, 2021].

[45] Kevin Delmolino - Mitchell Arnett - Ahmed Kosba - Andrew Miller - Elaine Shi. *Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab.* 2015. URL: https://eprint.iacr.org/2015/460.pdf [Last Accessed June 1, 2021].

[46] Patrick Collins and Carl Farterson. *Ethereum Smart Contract Best Practices.* URL: `https://consensys.github.io/smart-contract-best-practices/` [Last Accessed June 1, 2021].

[47] Jake Frankenfield. *Hard Fork (Blockchain).* URL: `https://www.investopedia.com/terms/h/hard-fork.asp` [Last Accessed June 1, 2021].

[48] Plus500. *The History of Ethereum.* URL: `https://www.plus500.com/Instruments/ETHUSD/The-History-of-Ethereum~4` [Last Accessed June 1, 2021].

[49] Ozora Ogino. *PROOF-OF-STAKE (POS).* URL: `https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/` [Last Accessed June 1, 2021].

[50] Trezor Wiki. *Hard fork.* URL: `https://wiki.trezor.io/Hard_fork` [Last Accessed June 1, 2021].

[51] Stuart D. Levi - Alex B. Lipton - Skadden - Arps - Slate - Meagher and Flom LLP. *An Introduction to Smart Contracts and Their Potential and Inherent Limitations.* 2018. URL: `https://corpgov.law.harvard.edu/2018/05/26/an-introduction-to-smart-contracts-and-their-potential-and-inherent-limitations/` [Last Accessed June 1, 2021].

[52] IBM. *What are smart contracts on blockchain?* URL: `https://www.ibm.com/topics/smart-contracts` [Last Accessed June 1, 2021].

[53] ConsenSys. *A 101 Noob Intro to Programming Smart Contracts on Ethereum.* URL: `https://medium.com/@ConsenSys/a-101-noob-intro-to-programming-smart-contracts-on-ethereum-695d15c1dab4` [Last Accessed June 1, 2021].

[54] CryptoCompare. *What Is the "gas" in Ethereum?* URL: `https://www.cryptocompare.com/coins/guides/what-is-the-gas-in-ethereum/` [Last Accessed June 1, 2021].

[55] Josh Fruhlinger. *What is cryptography? How algorithms keep information secret and safe.* URL: `https://www.csoonline.com/article/3583976/what-is-cryptography-how-algorithms-keep-information-secret-and-safe.html` [Last Accessed June 1, 2021].

[56] Kathleen Richards. *cryptography.* URL: https://searchsecurity.techtarget.com/definition/cryptographyl [Last Accessed June 1, 2021].

[57] Casey Crane. *Symmetric Encryption 101: Definition, How It Works and When It's Used.* URL: https://www.thesslstore.com/blog/symmetric-encryption-101-definition-how-it-works-when-its-used/l [Last Accessed June 1, 2021].

[58] SSL2BUY. *Symmetric vs. Asymmetric Encryption – What are differences?* URL: https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences [Last Accessed June 1, 2021].

[59] AboutSSL. *Symmetric Encryption vs. Asymmetric Encryption – How It Differs?* URL: https://aboutssl.org/symmetric-encryption-vs-asymmetric-encryption/ [Last Accessed June 1, 2021].

[60] Danny palmer. *What is GDPR? Everything you need to know about the new general data protection regulations.* URL: https://www.zdnet.com/article/gdpr-an-executive-guide-to-what-you-need-to-know/ [Last Accessed June 1, 2021].

[61] Michael Nadeau. *General Data Protection Regulation (GDPR): What you need to know to stay compliant.* URL: https://www.csoonline.com/article/3202771/general-data-protection-regulation-gdpr-requirements-deadlines-and-facts.html [Last Accessed June 1, 2021].

[62] European Commission. *What is a data controller or a data processor?* URL: https://ec.europa.eu/info/law/law-topic/data-protection/reform/rules-business-and-organisations/obligations/controller-processor/what-data-controller-or-data-processor%5C_en%5C#examples [Last Accessed June 1, 2021].

[63] Our Health service. *General Data Protection Regulation (GDPR) Frequently Asked Questions.* URL: https://www.hse.ie/eng/gdpr/gdpr-faq/hse-gdpr-faqs-public.pdf [Last Accessed June 1, 2021].

[64] GDPR. *Communication of a personal data breach to the data subject.* URL: https://gdpr-info.eu/art-34-gdpr/ [Last Accessed June 1, 2021].

[65] GDPR. *Notification of a personal data breach to the supervisory authority.* URL: https://gdpr-info.eu/art-33-gdpr/ [Last Accessed June 1, 2021].

[66] GDPR. *Lawfulness of processing.* URL: https://gdpr-info.eu/art-6-gdpr/ [Last Accessed June 1, 2021].

[67] GDPR. *Conditions for consent.* URL: https://gdpr-info.eu/art-7-gdpr/ [Last Accessed June 1, 2021].

[68] GDPR. *"Right to erasure/Right to be forgotten".* URL: https://gdpr-info.eu/art-17-gdpr/ [Last Accessed June 1, 2021].

[69] GDPR. *"Security of processing".* URL: https://gdpr-info.eu/art-32-gdpr/ [Last Accessed June 1, 2021].

[70] Marcus A. Rothenberger Ken Peffers Tuure Tuunanen and Samir Chatterjee. "A Design Science Research Methodology for Information Systems Research". In: *Journal of Management Information Systems* 24.3 (2007), pp. 45–78. DOI: https://doi.org/10.2753/MIS0742-1222240302.

[71] Shirley Gregor and Alan Hevner. *Positioning and Presenting Design Science Research for Maximum Impact.* 2013. URL: https://www.researchgate.net/publication/262350911_Positioning_and_Presenting_Design_Science_Research_for_Maximum_Impact [Last Accessed June 1, 2021].

[72] Remix - Etheruem IDE. *Welcome to Remix's documentation!* URL: https://remix-ide.readthedocs.io/en/latest/ [Last Accessed June 1, 2021].

[73] Solidity. *Solidity!* URL: https://docs.soliditylang.org/en/v0.8.4/ [Last Accessed June 1, 2021].

[74] Truffle Suite. *Ganache Overview.* URL: https://www.trufflesuite.com/docs/ganache/overview [Last Accessed June 1, 2021].

[75] Bruno Skvorc. *Developing for Ethereum: Getting Started with Ganache.* URL: https://www.codementor.io/@swader/developing-for-ethereum-getting-started-with-ganache-l6abwh62j [Last Accessed June 1, 2021].

[76] MetaMask. *Introduction.* URL: https://docs.metamask.io/guide/ [Last Accessed June 1, 2021].

[77] rdocumentation. *Eth_call: New message call.* URL: https://www.rdocumentation.org/packages/gethr/versions/0.1.0/topics/eth_call [Last Accessed June 1, 2021].

[78] Md. Jan Nordin Khadijah Chamili. *Searchable Encryption : A Review.* 2017. URL: https://www.researchgate.net/publication/323123557_Searchable_Encryption_A_Review [Last Accessed June 1, 2021].

[79] Muhammad Saqib Niaz and Gunter Saake. *Forward secure searchable symmetric encryption.* 2017. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8356345 [Last Accessed June 1, 2021].

[80] Florian Hahn and Florian Kerschbaum. *Searchable Encryption with Secure and Efficient Updates.* 2014. URL: https://www.researchgate.net/publication/289676976_Searchable_Encryption_with_Secure_and_Efficient_Updates [Last Accessed June 1, 2021].