UPPSALA
UNIVERSITET

UPTEC STS 18028
Examensarbete 30 hp
Juli 2018

Blockchain consensus mechanisms - the case of natural disasters

Okan Arabaci

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Blockchain consensus mechanisms - the case of natural disasters

Okan Arabaci

Blockchain is described as a trustworthy distributed service for parties that do not fully trust each other. It enables business transactions to be handled without a third party or central governance. For this distributed and concurrent communication to work, a consensus mechanism needs to be implemented into the blockchain protocol. This mechanism will dictate how and when new blocks can be added and in some cases, by whom.

The medical industry suffers from many informational inefficiencies. Data is scattered across many different databases and the lack of coordination often results in mishandling of the data. This is especially clear when a natural disaster hits and time is of the essence.

The purpose of this thesis is to assess how much a blockchain solution and its consensus mechanism can resist unusual behavior before they behave erratically. This involves analyzing design parameters and translating parameters from a disaster into a simulation to run tests. Overall, this thesis will explore if blockchain is a compatible solution to the difficulties in natural disaster response. This was obtained by conducting a qualitative study and developing a prototype and simulating disaster parameters in the prototype blockchain network. A set of test cases was created.

The results show that the resilience differs significantly depending on consensus mechanism. Key parameters include consensus finality, scalability, byzantine tolerance, performance and blockchain type. Blockchain is well suited to handle typical challenges in natural disaster response: it results in faster allocation of medical care and more accurate information collection, as well as in a system which allows seamlessly for the integration of external organizations in the blockchain network.

Handledare: Peter Forsberg
Ämnesgranskare: Kristiaan Pelckmans
Examinator: Elísabet Andrésdóttir
ISSN: 1650-8319, UPTEC STS 18028

Populärvetenskaplig sammanfattning

Satoshi Nakamoto skrev år 2008 en forskningsartikel ”Bitcoin: A peer-to-peer electronic cash system” som ändrade hela uppfattningen av hur transaktioner kan genomföras. Den nya digitala valutan Bitcoin kom att födas, och krävde längre ingen bank för att medla transaktioner mellan individer. Detta var möjligt på grund av den bakomliggande tekniken, blockchain, eller blockkedjan på svenska. Blockkedjan är en databas som lagrar information om transaktioner.

Kärnan i blockkedjan är att information inte kan läggas till eller uppdateras utan godkännande av alla som delar databasen. Denna process kallas konsensus, vilket betyder att användarna söker samförstånd i beslut om uppdatering av den delade databasen. Det betyder inte alltid att alla användare måste vara överens: olika protokoll finns som tolererar olika grad av oenighet. Förutom graden av oenighet som tolereras skiljer sig de olika protokollen avsevärt. Bitcoins protokoll kallas Proof of Work vilket betyder att beslutsfattarna måste bevisa att de har utfört en grad av beräkningskraft för att visa att deras beslut är giltigt. Andra protokoll som exempelvis Proof of Stake kräver ingen beräkningskraft.

Tekniken har fått mycket uppmärksamhet på grund av alla områden som den väntas kunna användas inom. Sjukvården är en av de största användningsområdena för blockkedjan då det finns många sätt denna industri kan gynnas av tekniken. Information om patienter finns splittrad på många olika fysiska platser och delade databaser är ingen norm inom sjukvården. När information ska förflyttas mellan aktörer krävs det många manuella ingripanden vilket är kostsamt och ineffektivt.

Detta blir speciellt tydligt i en naturkatastrof. Plötsligt behöver många människor snabbt få tillgång till sin information i en situation där kommunikation och koordination är bristande. Något som blir tydligt är att ju värre katastrof, desto mer distribuerad är katastrofhjälpen. Detta är på grund av den förstörda infrastrukturen. I denna uppsats studeras hur blockkedjan kan användas för att överkomma problem som uppstår vid katastrofer genom att bygga en blockkedjeprototyp där parametrar från en katastrof används för simulation. Utöver detta studeras vilka

parametrar som är viktiga för att uppnå motståndskraftig och säker konsensus i nätverket. För att ge en bakgrund till arbetet görs först en litteraturstudie där bakgrund till blockkedjor och konsensusmekanismer studeras. Sedan fördjupas arbetet i de tekniska komponenter som använts i prototypen, följt av de testfall som utförts.

Slutsatsen visar att blockkedjan kan användas för att överkomma många av de problem som uppstår hos vården i en katastrof. De viktigaste parametrarna som påverkar konsensus bryts ner till vilken blockkedjetyp det är, huruvida konsensus uppnås direkt eller probabilistiskt, om nätverket är skalbart och slutligen vilken tolerans nätverket har för illvilliga noder.

Preface

This study was conducted by Okan Arabaci in collaboration with IBM Svenska AB. The thesis was a result of the research and was a final assignment of the program Sociotechnincal Systems Engineering at Uppsala University.

I would like to thank my supervisors at Peter Forsberg and Markus Engström at IBM for guidance, fruitful discussions and for connecting me with valuable actors in the blockchain research area. I would also like to thank Kristiaan Pelckmans, my subject reader at the Department of Information Technology at Uppsala University, for all his support and for helping me plan my project.

29th of June, 2018

Contents

List of Figures

1 Introduction

The explosion of the cryptocurrency market in 2017 [1] led to a significant surge of articles that discussed virtual currencies such as Bitcoin or Ethereum, but along with that came interest for the underlying technology. Blockchain is described as a technology that will revolutionize many aspects of our society because of its many use cases besides currencies. Fundamentally, a blockchain system provides a trustworthy service for parties that do not fully trust each other [2]. By delegating the decision-making to a distributed network instead of a single entity, trust can be achieved in a way that wasn't possible in the past because each entity can control its information, but no single participant can control the entire network. More importantly, a system based on cryptographic proof has replaced the trust in an entity [3].

The capabilities that come with blockchain have sparked an interest in using it in other areas than finance which have been in the spotlight in the media craze. Governments around the world are discussing the idea of incorporating blockchain into their systems and some have launched pilots or even finished projects [4], [5]. Today, more than 100 projects in 30 different countries have launched to transform governmental processes using blockchain [5]. Past research indicates that governments need to start experimenting with blockchain in their own services to gain more knowledge of how blockchain works in a complex socio-technical aspect, and possibly even redefine their own role in the institutional environment [4].

One important government assignment is providing healthcare to its citizens. The healthcare system comprises of many different actors (hospital, doctor, patient..) and the process of receiving care usually involves most of them. This leads to fragmentation of the information since these actors do not operate on a shared information system [6]. Research is being done in the area of healthcare to incorporate distributed ledger technology and decentralized database solutions to overcome problems such as interoperability, security and health record universality [6]. Using smart contracts to manage the network flow and connecting medical devices in a

common information system are being explored [6].

In a large scale natural disaster, providing healthcare becomes even more important. Thousands of citizens are suddenly in need of immediate care and with loss of infrastructure, this is often easier said than done. Coordination is problematic which leads to consequences in the medical response where immediate care can be the difference between life and death. Accessing vital information about patients is often difficult [7]. Arguments regarding if centralization or decentralization is the best practice during normal procedure fluctuate. However, in disaster situations, the alternatives are decided by the environment which naturally makes nearly all processes distributed [7]. The level of decentralization follows the direness of the catastrophe: the bigger the disaster the more decentralized the response [7].

The consequences of a disaster from a blockchain perspective means that nodes in the network that were once active might have turned offline because of infrastructure losses. Reaching an agreement on updated information, otherwise known as reaching consensus in the blockchain can thus be troublesome in cases where the validating nodes are inaccessible. Hospital infrastructures as well as their patient records might become suddenly unaccessible. Depending on the blockchain setup and which consensus mechanism is used, the consequences can differ significantly.

1.1 Research question

The goal of this thesis is to review existing consensus protocols and use parameters from a natural disaster to test the security and resilience of a blockchain system. This is done to study if blockchain can be used in disaster medical response. The testing will be done by modeling a blockchain network using Hyperledger Fabric. Questions include:

- How compatible is blockchain as a technology solution to overcome the difficulties in disaster medical response?
- How much can a blockchain solution resist unusual behavior before existing consensus schemes behave erratically?
- What parameters are key for achieving consensus resilience and security? What is the relation between speed/accessibility and security/resilience?

2 Method

This section will describe the necessary background information needed to understand the study as well as provide insight into how the thesis was conducted.

2.1 Dictionary

Consensus	To agree within a group, to reach consensus.
PoW	Proof of Work. Consensus mechanism in Bitcoin and other blockchains.
PoS	Proof of Stake. Consensus mechanism in Peercoin.
PBFT	Practical Byzantine Fault Tolerance, protocol used in Hyperledger Fabric.
P2P	Peer to peer, distributed architecture.
DDoS	Distributed Denial of Service. Flooded traffic from different peers.
Byzantine node	Peer that intentionally tries to harm the network, also referred to as malicious.
Consensus algorithm	A protocol for reaching consensus. E.g. PoW.
Consensus finality	How consensus protocols are finalized, probabilistic or immediate.
Miner	Nodes in blockchain networks that validate new blocks.
Anchor peer	A peer that updates off-sync peers with new blocks.
Channel	A private sub-net between two or more peers in the Fabric blockchain.

2.2 System architectures

A system consists of a network of communicating nodes, each with various computational functionalities. A system is centralised (or follows a master-slave layout) if there is a single authorising node [8]. A system is decentralised if it has multiple governing nodes. A system however is distributed if such responsibility is shared across different nodes (see Figure 1 for graphical examples). Each node can be classified as honest, faulty or malicious [9]. The definition of malicious behaviour can differ in research papers. In this thesis, malicious behaviour will refer to intentional actions by participants in the network to satisfy their own selfish goals such as double-spending, stealing information, message tampering and purposeful delay.

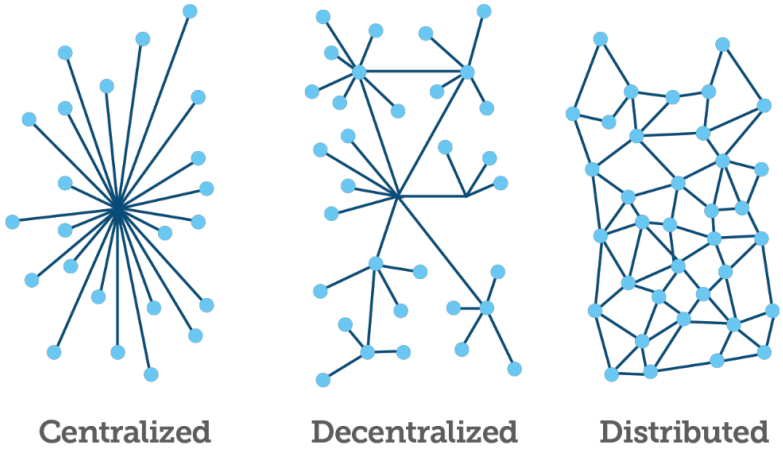

Figure 1: Centralized, decentralized and distributed systems architecture.

The biggest challenge in distributed systems is how to coordinate the nodes and how to handle behaviour that is out of the ordinary. A distributed system should work even when faulty nodes are present or when connections between nodes break, and reach the desired state. Research has been conducted in this area for years and algorithms to handle this exception are continuously proposed [9].

2.3 Blockchain

Depending on your perspective, the definition of a blockchain network can differ. Authors Don and Alex Tapscott of "Blockchain Revolution" define blockchain as "an incorruptible digital ledger of economic transactions that can be programmed to record not just financial transactions but virtually everything of value" [10]. From a business perspective, a blockchain can be viewed as a distributed ledger, or record book, where participants exchange assets by performing transactions which are stored in the ledger. This is done without the need for a central authority to process, validate, secure or even hold the data. Instead, the data is held on all the

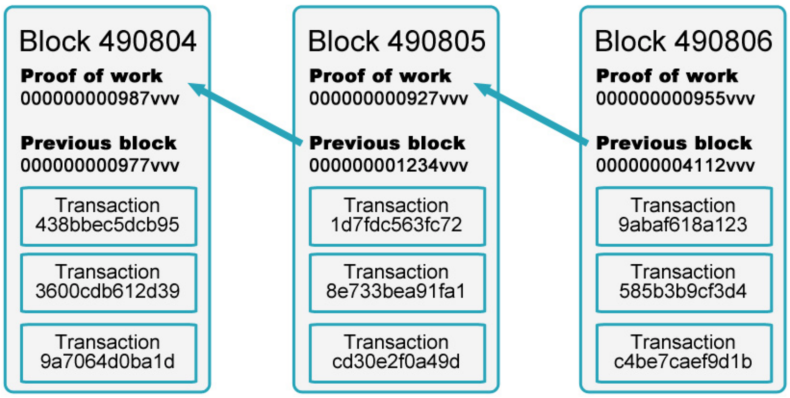

Figure 2: Simplified structure of blocks in a Proof of Work blockchain.

participants in the network [9].

From a technical perspective, a blockchain is the term for a distributed, peer-to-peer (P2P) software network. Blockchain is implemented as a layer on top of the web which can be seen as an analogy to SMTP, HTTP or FTP running on top of TCP/IP [9]. It uses cryptography to secure applications, data, and the transfer of assets that can represent real-world currency [11]. A blockchain is append-only, immutable and only updatable with the consent of the peers within the network which is performed using the built in consensus mechanism. In theory it's not impossible to change data on the blockchain. The validator would need to perform the same amount of computational work that has been done since that specific transaction was verified, and before the next block gets committed. But since it's considered extremely hard in practice, this property stands.

Continuing on the technical viewpoint, a block within the blockchain is a logical bundle of transactions (Figure 2). It illustrates a Proof of Work blockchain where each block is generated based on the hash of the previous block (number below "Proof of Work"). Depending on the design of the blockchain, the structure and size of a block can differ. While there may be additional attributes, there are some that are generally necessary for the block to function properly. These include a block header, time-stamp, nonce, pointer to previous block (hence blockchain), transaction count and the transactions [9].

2.4 Blockchain system taxonomy

Blockchain implementations can be architected in several different ways. Roughly three different categories exist: public, private and consortium blockchains [12]. The method to reach consensus is highly dependent on the architecture of the network. This section will explore the different blockchain architectures, in what setting they are appropriate to use and their constraints in that setting.

2.4.1 Public blockchain

In a public blockchain, anyone can read the ledger, send transactions and see their own transactions in the ledger. It's also possible for anyone to validate blocks - or participate in the decision making process. Depending on the design, users can get rewards based on their participation like in the Bitcoin blockchain [9], [12]. This architecture is usually referred to as a permissionless blockchain. In a permissionless blockchain, there is no single owner since all users maintain a copy of the ledger locally. It is considered to be "fully decentralized" [12].

Securing a permissionless, public blockchain without trusting a centralized authority is achieved by cryptoeconomics which essentially is a combination of economic incentives and a cryptographic verification process. This process refers to the consensus mechanism which is usually Proof of Work or Proof of Stake [12]. The underlying rule is that a user can only participate in the consensus process equivalent to the economic resources that they possess, either computational (PoW) or stake (PoS).

The limitations of a public architecture refers to many aspects, one important one being the symmetry of information. Since all the information on the ledger is symmetric, the use of a public blockchain in a business scenario can become troubling. Other businesses in the network are provided full access to the entire set of transactions, potentially providing insights to the business intelligence of other companies [13]. Hence, a public BC is not suitable for settings where confidential transactions occur or where ledger data is sensitive. Firms working with IBM to implement

blockchain have almost all demanded that the system should not be open like Bitcoin [14].

2.4.2 Consortium blockchain

In a consortium blockchain, the participants' identities are known and the consensus process is controlled by pre-selected nodes. Consortium blockchains are normally limited to the companies involved in the business scenario or a part of it. These blockchains are consequently permissioned and are usually considered partially decentralized because of the smaller number of nodes compared to public blockchains, but also because of hierarchy in the nodes. The hierarchy refers to the fact that different firms usually have different views of the data which helps securing loss of business intelligence information to competitors. [13] Reading permissions might be public or restricted to members in the network. Hybrid solutions also exist where public users can run limited queries using an API with public root hashes [12].

Consortium blockchains usually don't require the extensive security that mainstream consensus algorithms such as Proof of Work provide since the members are trusted [9]. This is a disadvantage compared to public blockchains where anyone can join because deciding who can join the network or who can commit data to the ledger is usually done in a centralized manner [15]. Also, because of the inequality of the members in the network, deciding on how to reach consensus in a secure manner is more difficult particularly in business settings where one firm has a substantially more power than others [13]. However, the legal and compliance requirements of blockchain projects in many industries (e.g. finance, real-estate, banking) will likely require the user to be known anyway, and that's why consortium blockchains mostly use other consensus protocols such as PBFT instead of PoW [15].

2.4.3 Private blockchain

This blockchain type is very similar to consortium blockchains but instead with a single organization as a participant. Many argue that private blockchains add no value and that a normal database can be used instead since there is no need to trust other parties and exchange of information between them. Gideon Greenspan has defined a set of criteria that need to be fulfilled for blockchain to be a good use case [16]. The second criteria states that there has to be multiple writers in the database which means more than one entity is required [16]. This thesis will not delve deeper into private blockchains.

2.5 Consensus

Consensus is the word for a general agreement or majority of opinion between a number of subjects [17]. It is a fundamental part of everyday life like deciding where to eat with a group of friends or which movie to watch. In a distributed system, deciding if data is accurate when the system is updated is a complex task. William Mougayar describes this decentralized scheme interestingly: "At the heart of understanding the severity of the blockchain paradigm shift lies the basic understanding of the concept of decentralized consensus" [18]. Consensus in a blockchain network refers to the process where the distributed nodes agree on the history and the final state of the data on the ledger, usually referred to as distributed consensus [9]. Since all participants in the network hold the data, they can also be a part of the decision-making. Every new block that gets added to the blockchain needs to be agreed upon according to the defined protocol so that the replication is done uniformly. To summarize, distributed state-machine replication involves two elements [2]:

State machine - implements the logic of the service that is to be replicated.
Consensus protocol - the dissemination of requests to the machines in a deterministic manner so that each node handles an identical sequence of events on its

instance.

Researchers have been developing algorithms for replication in distributed systems for many years, reaching back to Lamport et al.'s introduction of Byzantine agreement [19]. The famous analogy of the Byzantine generals problem describes how malicious behavior must be taken into account in distributed consensus models:

> This situation can be expressed abstractly in terms of a group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a common battle plan. However, one or more of them may be traitors who will try to confuse the others. The problem is to find an algorithm to ensure that the loyal generals will reach agreement. It is shown that, using only oral messages, this problem is solvable if and only if more than two-thirds of the generals are loyal; so a single traitor can confound two loyal generals. With unforgeable written messages, the problem is solvable for any number of generals and possible traitors.

A set of properties exist that are considered when describing consensus protocols: [20]:

1. Safety – This is generally considered as the most important aspect to consider. For a consensus protocol to be considered safe, all the nodes have to produce identical outputs (consistency) and the outputs have to follow the protocol rules.

2. Liveness – All honest nodes that participate in achieving consensus eventually produce a value.

3. Fault Tolerance – For a consensus protocol to be fault tolerant, it needs to recover from byzantine nodes participating in consensus.

This is the famous Fischer Lynch Paterson (FLP) impossibility theorem [20], [21], and it states that an asynchronous distributed consensus system can't provide both consistency and liveness without sacrificing fault tolerance. Dwork et al. [22] introduced the concept of eventual synchronous models which differs from the asynchronous model in the sense that messages may be arbitrarily delayed among working nodes. This delay is a result of the FLP impossibility. However, it eventually reaches synchronization and delivers the delayed messages within a fixed but unknown time interval. Eventual synchronous models do not violate their safety property when asynchronous if the right assumptions about the faulty nodes are made. The network eventually stabilizes at which point the nodes guarantee liveness again. Even though the asynchronous model is widely accepted today, many protocol designers prefer the eventual synchrony model because of its simplistic and realistic approach to network behaviour [2].

2.6 Consensus protocols

This section will provide a review of current protocols and their security aspects. A working consensus protocol preserves the quality of the data on the ledger since strict rules apply when updating the ledger. Consensus protocols can be formally analyzed by focusing on certain design parameters such as node identity management,

consensus finality, scalability (number of nodes participating in consensus), performance (latency, transactions per second), adversarial behaviour tolerance, network assumptions and lastly existence of correctness proofs [15].

2.6.1 Proof of Work (PoW)

In the famous paper by Satoshi Nakamoto [3], state-machine replication is not explicitly mentioned. However, the anonymous writer(s) describe the process of consensus in the Bitcoin protocol as nodes voting with their CPU power to validate or reject blocks. In the Bitcoin blockchain and most other public blockchains, reaching consensus is achieved using the PoW algorithm. Block validators have to prove that they have performed some amount of computational work which is done by finding a hash value that is less than the difficulty level set by the network [20]. Bitcoin's difficulty is set according to the total hash power of the network to create a new block every ten minutes. Solving the PoW puzzle and finding the value is also known as mining [20]. The first node that finds the value lower than the difficulty wins the mining race and gets its proposed block added to the blockchain along with a mining reward which is paid in Bitcoin [20]. Security in PoW blockchains is thus achieved by the miners consuming electricity and therefore having an economic interest in following the rules.

Because the network is fully distributed and asynchronous, temporary forks can occur where more than one validator finds a winning hash value (Figure 3). This means that each validating node adds the proposed block to the blockchain and broadcasts it to the network, thus creating diverging branches. As more blocks are added, the protocol will solve this problem by selecting the longest branch (the branch with maximum PoW) and the other branches will be discarded [20]. Consensus finality can thus not be ensured [15].

The PoW consensus model is well suited for environments where no authentication of participants is needed and where anyone can join as a validator. PoW is however vulnerable to dominance attacks, or more commonly known as 51% attacks. In a

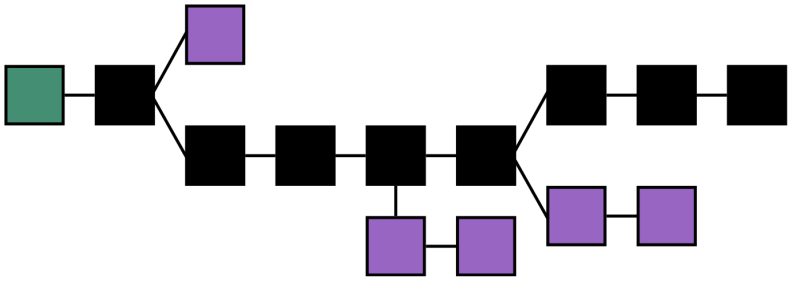

Figure 3: Illustration of a blockchain fork where the purple blocks are the blocks of the diverging branch and the black blocks are part of the original chain.

51% attack, the attacker (a person, group, organization..) takes control of the network by controlling more 51% or more of the computational power in the consensus protocol. Getting this majority can be done in many ways, but usually by convincing the community that their chain is the valid one and should be considered the longest chain. By having this power, the attacker(s) can choose to disregard valid transactions and accepting invalid transactions such as double-spending a cryptocurrency [20].

Another problem with PoW is the slow confirmation rate which is around 7 transactions per second [15]. Compared to VISA's 10 000 transactions per second, this throughput is far from usable for commercial purposes [20]. Since mining requires computational work, mining gear is needed to participate in the validation of blocks. As Nakamoto stated in the paper, nodes vote using their CPU. However, actually getting a reward using CPU mining today is not feasible with the vast amount of competition. Special mining gear called ASIC's (Application Specific Integrated Circuit) or at least normal GPU's are required to actually get a reward as a miner. This of course involves additional cost beyond the electricity consumption which was estimated to be between 0.1 and 10 GW in 2014 [15].

Moreover, PoW can support a working consensus with at most 25% of network power acting maliciously [15]. Up until recently, the understanding was that PoW could handle everything below 50% and work properly but that was lowered to 25% after further research [23].

2.6.2 Proof of Stake (PoS)

The disadvantages of PoW have sparked interest in designing algorithms that don't have nearly as high electricity consumption in the mining process. PoS replaces the computational work in the mining process with using the validators stake/ownership of a certain cryptocurrency. Instead of spending money on electricity and mining gear, validators can instead buy cryptocurrency and use that currency to engage in the validating process to get rewards and keep the network operating. The amount of currency at stake by the validator corresponds to their probability of being chosen as a validator [20]. However, PoS algorithms pseudo-anonymously choose a validator through a randomization algorithm to ensure that no validator can be certain that they are chosen.

The security is not achieved by consuming electricity, but instead by putting up economic value in the form of a stake in cryptocurrency which can be lost by malicious acts by the validator. Thus, PoS is more penalty focused compared to PoW which is reward based to ensure security [24].

2.6.3 Practical Byzantine Fault Tolerance

The Practical Byzantine Fault Tolerance (PBFT) algorithm was developed by Castro and Liskov [25] to solve the problem of reaching consensus in the face of Byzantine nodes. It was the first practical implementation of Byzantine fault tolerant consensus algorithms. PBFT uses state-machine replication and replica voting for changing the state in the network. Nodes acting as validators in a PBFT network all have equal votes and the validation is run through multiple rounds to reach consensus [26]. PBFT is used in private settings where no authentication is needed. Characteristics that optimize the communication are signing and encrypting messages between replicas and clients, and minimizing the number of exchanged messages. This is to make the algorithm usable in the face of Byzantine faults. It requires $3f + 1$ replicas to tolerate f Byzantine nodes [25]. This can also be formulated as less than 33% of the nodes can be faulty for consensus to be reached.

2.6.4 Kafka

Apache Kafka is a platform that aims to achieve high throughput and low latency in distributed communication. Even though it is used as the consensus mechanism in Hyperledger Fabric blockchain, Kafka was not specifically designed for blockchain systems. It has a publish/subscribe interface similar to a message queue system which applications can interact with [27]. Kafka stores streams in a fault-tolerant way and is often used to communicate real-time reliably between different systems or applications. Kafka's usage in Fabric is permissioned and voting based like PBFT.

The platform consists of broker and consistency nodes and to achieve redundancy, a set of brokers process each message stream in parallel. A consumer subscribes to a topic to receive messages (see pseudocode below). The brokers store the messages in partitions and if more than one message is sent, they are stored in separate partitions. A ZooKeeper instance is running on the consistency nodes for coordination of the brokers [28]. If they crash or go offline, ZooKeeper ensures that new brokers are spawned. Thus, Kafka inherits the resilience against crashes from the ZooKeeper ensemble that respawns nodes. Kafka does however not tolerate byzantine failures [27].

```
int interval = 100 ms   # arbitrary frequency for messages

for(every(interval))
    if(consumer.gotMessage())
        if(messages > 1)
            for(messages[i])
                storeInSeparatePartition(message[i])    # one msg per partition
                consumer.processMessage(messages[i])
                consumer.sendAckToBroker()
                kafka.changeOffset()
        else:
            store(message)
            consumer.processMessage(message)
            consumer.sendAckToBroker()
            kafka.changeOffset()

    # repeat until consumer stops request
```

2.7 Hyperledger Fabric

Hyperledger is an umbrella project that foster open source blockchain frameworks and tools. It's hosted by The Linux Foundation together with front runners in industries such as finance, banking, IoT, supply chains and manufacturing as a collaborative effort with a common goal: develop cross-industry blockchain technology [29]. Blockchain solutions with different use cases hosted by Hyperledger include Sawtooth, Iroha, Burrow, Indy and lastly, the most prominent one, Fabric with its recent v1.0 release.

Initially developed by Digital Asset and IBM, Fabric was a result of a hackathon. Its main purpose is to act as a foundation for developing modular architecture solutions by allowing consensus and membership services to be pluggable [30]. It means that consensus policy can be chosen and designed after the needs of the application and

its context. The next subsections will explore the Fabric blockchain capabilities that will be used in the prototype.

2.7.1 Endorsement policy

During the consensus workflow, peers decide if a transaction is valid or not by checking against the endorsement policy. Upon receiving a transaction, the peer invokes VSCC (Validation System Chaincode) to check if the transaction is valid. VSCC checks three things: all endorsements are valid, the number of endorsements are valid, endorsements are from authenticated members. The endorsement policy is a flexible parameter that can be set according to how the business network operates normally. For example:

AND('Org1.member', 'Org2.member', 'Org3.member') requests 1 signature from each of the three organizations
OR('Org1.member', 'Org2.member') requests 1 signature from either one of the two organizations
OR('Org1.member', AND('Org2.member', 'Org3.member')) requests either one signature from a member of the Org1 or 1 signature from a member of the Org2 and 1 signature from a member of the Org3.

2.7.2 Smart contract/chaincode

Hyperledger Fabric, Ethereum and some other blockchains offer a smart contract functionality. Smart contracts are autonomous programs on top of the blockchain that encapsulate business logic and code which are executed when certain conditions are met [9]. The flexible and programmable nature of smart contracts are highly desirable features since it means that they can be implemented based on most business scenarios.

In Fabric, smart contracts are referred to as chaincode but the underlying principle

is the same. Two types of chaincode exist: system chaincode and application chaincode. The code is run in secured Docker containers and is invoked by users through transactions to manage the ledger state [30]. So if an end user wants to update a key value in the ledger, it has to do this by invoking chaincode.

2.7.3 Channel

Fabric provides a unique mechanism for members of the network to perform transactions in complete confidentiality in respect to the rest of the network. The channel is essentially a subnet of the blockchain intended for communication between two or more specific members. Elements in a channel include [30]: members (organizations), anchor peer(s) per member that is responsible for communication with other members on behalf of the organization, the shared ledger, chaincode and the ordering service. For each channel, a new shared ledger is created only with the involved participants and with chaincode that is bound to a specific channel. The ledger can be extended to include the entire network if the participant is a member of the "common channel" [30]. Chaincode can be installed only on the peers that will access the asset state and perform read/write operations. The information including transactions, members and channel policy are invisible and inaccessible to other network members that are not part of the channel [30].

When creating a new channel, a client SDK (Software Development Kit) sends a request to the system chaincode and in that request, specifications of anchor peer(s), members and policies of the channel are included [30]. When the chaincode receives the request, a genesis block for the channel ledger is created which stores the configuration information about the channel policy, members and anchor peers. When adding new members to the channel, the configuration block is shared with the new member. If the channel policy needs updating, a new configuration block can be appended to the blockchain.

2.7.4 Identity and membership management

Fabric being a permissioned blockchain, identity management is required. Hyperledger Fabric provides a membership identity service that manages IDs and authentication of participants which consequently means that members know each other [30]. It allows for different organizations and participants to use their own certificate authority, implementing various cryptographic algorithms for signing/verifying/identity authentication. This is done through an Membership Service Provider (MSP) process running on both the ordering service nodes and the channel. The MSP is a set of cryptographic mechanisms and protocols for issuing and validating certificates and identities throughout the blockchain network [30].

2.7.5 Node types

In Fabric, three different types of nodes exist and they serve as different roles in the network [30].

Client is the node that submits the transaction request to the endorsing peers and has the responsibility of broadcasting transaction proposals to the ordering service nodes. The client is acting on behalf of the end-user by communicating with peers and the ordering service to interact with the blockchain.

Peer is the node that maintains the state and holds a replication of the shared ledger. It is the node that commits transactions to the ledger. Peers can also take up another role, endorsing peer or endorser. An endorsing peer will endorse transactions before they are committed according to a defined endorsement policy in that channel. The policy states the required conditions for accepting a transaction as valid, which is usually a set of endorsing peer signatures.

Ordering service node (OSN) is running the communication service that implements a delivery guarantee, such as atomic or total order broadcast. This means that the output from the channel will be identical to all peers, and in identical order.

The ordering service provides the consensus aspect of the Fabric network. It can be implemented as a centralized service which is usually done in development and testing, or it can be distributed to target different network models.

The ordering service is the communication channel to client and peer nodes. A client connected to a channel can broadcast messages on that channel and the messages are then sent to all the peers. Peers connect to the channel by the OSN using the API interfaces that are provided.

2.7.6 Gossip protocol

By dividing the workload between the endorsing and committing peers and the ordering service nodes, Fabric provides optimal performance, security and scalability. It does however mean that the dissemination of data need to be secure, reliable and scalable. These requirements are met by the implementation in Fabric called gossip data dissemination protocol [31].

Peers in the network use the gossip protocol to broadcast ledger and channel data in a scalable manner [31]. This protocol is running continuously to keep all peers up to date on newly committed data from multiple peers. The gossiped messages are signed by the sender's key and Byzantine nodes are thereby easily identified [31]. Peers that go offline or are subject to delays will get updated to the current ledger state by communicating with other peers that already are up to date.

2.8 Healthcare industry

The healthcare industry includes a wide range of companies and non-profit organizations (NPO) that are responsible for providing medical services, equipment and pharmaceuticals to individuals or a community [32]. Physicians, dentists, patients, hospitals, pharmacies, research and educational organizations are examples of participants. Moving from the ever decreasing paper based documentation of informa-

Figure 4: Frictions that challenge growth and innovation in the healthcare industry.

tion, digitized medical records are getting more common in the healthcare industry. Estonia is an example of a country that has digitized the medical records of all inhabitants using blockchain technology [33].

Multiple stakeholders create a complex ecosystem of sensitive interactions which leads to security challenges and inefficiencies in operations even with digitized medical records [6]. The most common problem stated by past research indicates that a shared source of information is needed to simplify access to health data in an efficient way and replacing the fragmented system that exists today [6], [7], [34]–[36]. Moreover, clear ownership of the data and restricting access to the medical records only to allowed parties are two critical aspects of the medical industry [6]. Today, the patients do not actually own their medical records since they reside within the database of the hospital or a similar organization. The records can theoretically be read or updated by anyone with access without the consent of the patient.

The design of electronic health record (EHR) systems was not taking into account the multi-institutional complexity [34] and "long data" that captures the lifetime of a patient [35]. Thus, the processes are too elaborate and costly today. Figure 4 illustrates the various frictions that hold back the medical industry from growing and innovating [35].

2.8.1 Healthcare Disaster

Even though large-scale disasters are a rare phenomenon, there's a need for a resilient EHR infrastructure that can handle the delivery of healthcare to the public during such events [36]. In the event of a natural disaster or a bioterrorism attack, it is vital that the inhabitants can access their medical information quickly and share it with caregivers. Absence of a robust infrastructure that provides access to the information will undermine attempts to construct a reliable plan to handle the disaster and its consequences on the public health [36]. Emergency medical services (EMS) must have mechanisms to generate and communicate data that are valid, reliable and accurate [37]. They should also collaborate with other health care providers and community resources to develop integrated information systems [37].

The security of an EMS information system is vital and can be split into two areas: security and confidentiality of the patient's information, and the system's information [37]. Any information system that collects data from multiple systems should have policies and procedures defined to prevent the identity of medical systems from being disclosed to any outside parties, or the public, without consent [37]. Patient security is also critically important to an EMS information system. Policies and procedures should be developed and implemented to provide appropriate access by personnel in need of patient data but also to protect the patient from unnecessary exposure.

Transparency is also an important aspect to consider. Even with swift governmental and NGO reactions, the lack of transparency among care providers harm the humanitarian efforts. The dire 7.0 magnitude earthquake in Haiti in January 2010 is an example of this [38]. 20 nations and NGOs were within 24 hours actively trying to help the disaster struck island with medical support, food and water. Each organization brought their resources and specific knowledge and all together combined their endeavor to achieve higher efficiency. It was, however, revealed afterwards that the full potential of the provided efforts could not be utilized because of the absence of a shared communication channel [38].

2.9 Analysis method

This section describes the method used to conduct the research and a discussion of chosen method.

2.9.1 Timeline

Before entering the field of blockchain, several research questions were devised. One aspect was however established fairly early which was to focus on consensus mechanisms. During the first 8 weeks, the work mostly involved an extensive literature study. Both the healthcare and the government sector using blockchain was explored. It was done to better navigate the research context which was fairly unfamiliar. Research papers regarding various consensus mechanisms were studied to establish a set of relevant protocols to discuss along with the design parameters that describe their behavior.

The first use case involved exploring the use of blockchain in governmental processes. This was the natural first decision since the thesis supervisor at IBM has close contact with defense and intelligence customers. A workshop together with two representatives from the defense/intelligence sector was conducted where the goal was to reach a feasible project idea that involved a real industry problem. Even though the workshop stimulated fruitful discussions, it didn't result in a use case for the thesis since the information about network participants, assets and transactions are too confidential to share with students without proper security clearance.

Since the government use case didn't yield any direct project ideas, the healthcare industry was made the focus of the thesis. Several blockchain projects that operate within the medical industry were explored to get an idea of how their solutions were designed. This involved reading whitepapers and research papers. The lack of research in consensus mechanisms linked to societal problems sparked an idea in using a natural disaster as a simulated stress test on the network. This would require building a prototype that has enough capabilities to be tested with this scenario.

After discussions with the thesis partner and supervisors, a network design was proposed and initial sketches were made.

The next step involved developing the actual implementation which was done in the next 4-5 weeks. During this time and the remaining period of the study, the thesis was written continuously. The last weeks involved testing and preparing the presentation of the thesis.

2.9.2 Qualitative research

Qualitative research involves a range of methods such as interviews, participant observation, document analysis and can include research in the field, using a focus-group or working in the office [39]. Qualitative research is suitable when the researcher wants to approach the topic in a holistic manner and where the researcher wants to make sense of the subject [39]. It focuses on giving a deep description of the context and is usually developed from a problem in the field.

Many blockchain projects being developed tend to focus too much on the technical aspect and foregoing the necessary connection to a societal issue [4]. It's important to keep in mind what the problem to be solved is, and if blockchain is the proper technology to be used with it. Blockchain advocates present the technology as a solution to almost any type of information system related issue and don't consider the cumbersome process that can come when trying to implement it, or the criteria that need to be fulfilled to actually derive value from it. This can be linked to the immaturity of the technology and the reduced knowledge of the potential value gained from it [4]. There is a need to incorporate public values into blockchain architecture to fairly represent the environment. A qualitative study is therefore suitable for this thesis to expand on the research question and link it to a wider, more generalized issue in our society. A qualitative approach is also suitable when the researcher is personally interested in the subject but hadn't a valid reason to conduct research in it [39].

2.9.3 Material

The majority of material that was gathered in this study was a result of the literature study. To get an overview of blockchain, consensus mechanisms and the healthcare industry, research articles were read mostly from the Uppsala University library and Google Scholar. Past theses in the subject of blockchain were also examined to acquire useful references and get an idea of how research in blockchain has been done in the past. Additionally, many useful articles were sent by the developers working on Hyperledger Fabric and in a few cases, technical consultation was also provided through conference calls.

Technical material such as code and documentation was gathered from Github repositories and the Hyperledger Fabric documentation. This was used as a starting point for the development of the Hyperledger Fabric prototype blockchain network that was used in the testing. The resulting material of the development was a Fabric v1.1.0 blockchain network running with 6 organizations, a smart contract that handles creating, retrieving, updating and deleting medical records and a RESTful API to interact with peers in the network. In some test cases, the testing tool Caliper was used to stress the network with requests. Finally, bash scripts were created to automate processes such as adding new members and launching/stopping the network. See (https://github.com/okaara1/fabric-disaster-prototype) and (https://github.com/filippoboiani/fabric-disaster-kafka).

2.9.4 Method discussion

When designing and developing the medical records blockchain system, there was no input from industry representatives. The information about how a healthcare network is set up, how medical records are managed and other information about the current work flow was instead gathered from literature reviews. To get a more realistic problem based approach, the better alternative would've been to discuss the proposed thesis idea with representatives in the industry together with a literature review to make the basis of the design more scientific. It would also enable the

researcher to conduct the study with a more specific case in mind which would possibly yield more realistic results.

The results from the tests were obtained by simulating a disaster in the network by trying to model the disaster as close to reality as possible. Even though a realistic simulation was a priority, it's still a simulation which means that the results should be interpreted with this in mind. Since a real disaster is unfeasible to produce, this was however the only alternative at hand. In a real disaster, other parameters and consequences that were not considered might exist.

2.10 Consensus testing

This section will describe the architectural choices made based on the literature review of blockchain setups and consensus mechanisms regards to the medical industry and disaster response management. Until now we have explored different blockchain setups and consensus protocols. However, as stated before, it's important to keep the link to a societal issue to get a reference point of the problem and the results obtained from tests. The assumptions will thus be made by using information about a natural disaster and reusing the parameters in the testing.

Looking at the different blockchain types makes it clear that in the medical industry, a permissioned blockchain is best suited because medical information such as patient records are confidential. A consortium blockchain would be a good architecture where hospitals and other actors involved in the healthcare chain can join the network and share the information between each other. To test the consensus, a set of test cases will be created using Hyperledger Fabric. The network will consist of 4 organizations, of which 2 are hospitals, one is a disaster response organization and one is the Ministry of Health (see Figure 5). They will all operate on a common channel in which they share information between each other. A simple chaincode will be run on validating peers that handles the medical records.

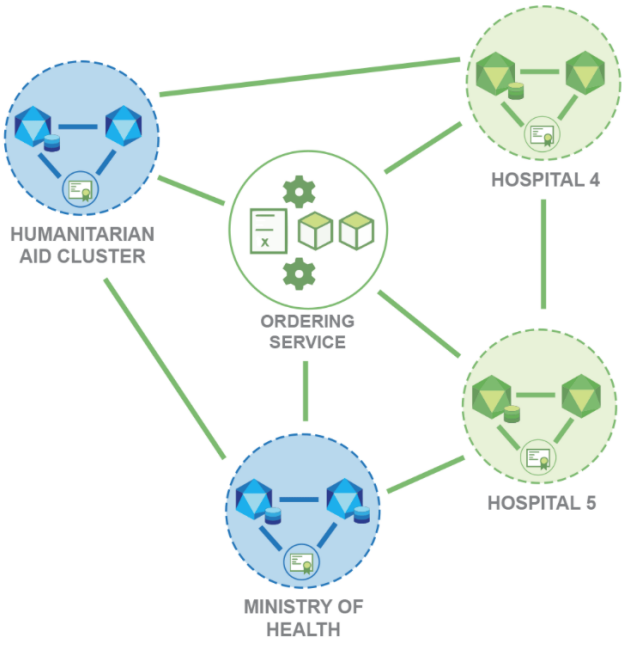

Figure 5: Network architecture.

Test case 1

During a disaster, peers may go offline either quite shortly or up to hours or even days. What is the consequence of a peer going offline and coming back online? While the peer was offline, the blockchain may have been updated several times and the ledger is outdated for the peer that was offline.

To simulate a peer going offline, the Docker container in which it runs will be paused for a period of 2-3 minutes while the rest of the network appends new blocks to the network. The log for the peer will be checked during the entire process. Will the peer get synced up with the rest of the network after coming back online?

Test case 2

In the Haiti disaster, there were a lot of NGO presence in the area. Getting them up to date on medical information and coordinating the new members takes a lot of time normally. Using Fabric, this test case will examine how the consensus process of involving new members into the response affects the security of the network and

if the time aspect can be improved compared to today.

Test case 3
In the Haiti earthquake, out of 49 hospitals 30 were either totally destroyed or seriously damaged which can be rounded to 60% [7]. How would a 60% drop-off in network participation affect the consensus scheme? Will it still operate normally or will a drop-off cause the network to crash?

Test case 4
Naturally, right after disaster hits, the request for healthcare and information regarding healthcare will be significant. During this high flow of information, will the system hold or crash? What vulnerabilities exist during this scenario?

3 Result

The results from the study and testing will be presented in this section.

3.1 Protocol summary

Developing consensus protocols is not a trivial task. The solution should optimally be secure enough to be resilient against a wide range of adversarial behaviour and at the same time not affect the functionality to the point that it can't accomplish its primary task. It has to prevent, deter, withstand or tolerate behavior that can prevent the user from performing wanted actions. Still, validating the security of the model is often made in a non-formal manner which results in vague security claims [2]. Experiments can't test the consensus protocol in all possible scenarios and often standalone only shows the failure of the protocol security. To prove that the solution is secure under any scenario stated by the trust assumption, mathematical reasoning and formal validation is also required. Evaluating if the model is appropriate and covers the scenario in which it operates, human judgment is also

needed in the form of reviews, study, validation and expert agreement [2]. Open discussion between developers, investors and users in the industry should replace the hype that surrounds blockchain and cryptocurrencies.

The trust assumption is a fundamental aspect of a blockchain protocol. It defines the environment for which the protocol is designed and in which it satisfies its guarantees. Assumptions should cover all elements in the system such as the network and the expected behavior or misbehavior of the nodes. The purpose of a trust assumption is to represent an idealization of the environment in which the protocol operates. If the model lacks some aspect of the real world behavior, it can affect the system negatively and the security must be revised.

To summarize the different parameters of each consensus protocol and provide a basis for further analysis, a table describing the different aspects of each protocol has been created (Table 1). Consensus finality refers to how the new blocks are "calculated". In PoW, the miners have to solve a puzzle which is probabilistic. It depends on the computing power, the network latency and other parameters. But in PBFT and Kafka, the finality is immediate since there is no puzzle to calculate.

Table 1: Summary of consensus protocols.

	PoW	**PoS**	**PBFT**	**Kafka**
Blockchain type	Permissionless	Both	Permissioned	Permissioned
Node identity	Anonymous	Anonymous	Known	Known
Consensus finality	Probabilistic	Probabilistic	Immediate	Immediate
Scalability (# of nodes)	High	High	Low	Low
Performance (tx/s)	Low	High	High	High
Byzantine tolerance	$\leq 25\%$	Depends*	$\leq 33\%$	-

*PoS algorithms have different implementations. Based on which one is used, the system has different tolerances for Byzantine nodes.

3.2 Test case 1

In the first part of the log, we see that the gossip protocol notices that the peer is offline and tries to establish communication in a frequent manner. The time period, 2 minutes and 24-25 seconds, in which the peer is offline is also shown. Lastly, a warning appears in the log that shows the peer isn't responsive anymore.

```
2018-05-14 13:30:34.650 UTC [gossip/discovery] getDeadMembers -> WARN 042 Haven`t heard from peer1.org1.example.com:7051 for 2m24.1294292s
2018-05-14 13:30:34.662 UTC [gossip/discovery] getDeadMembers -> WARN 048 Haven´t heard from peer1.org1.example.com:7051 for 2m25.8292851s
2018-05-14 13:30:34.653 UTC [gossip/comm] func1 -> WARN 043 peer1.org1.example.com:7051 isn`t responsive:
rpc error: code = Unavailable desc = transport is closing
```

After getting back online, the out of sync peer reads the blocks from other peers and commits them to its own ledger. This means that the peer is now in sync with the rest of the network.

```
2018-05-14 13:30:34.725 UTC [kvledger] CommitWithPvtData -> INFO 051 Channel [mychannel]: Committed block [6] with 3 transaction(s)
2018-05-14 13:30:34.753 UTC [kvledger] CommitWithPvtData -> INFO 054 Channel [mychannel]: Committed block [7] with 3 transaction(s)
2018-05-14 13:30:34.770 UTC [kvledger] CommitWithPvtData -> INFO 056 Channel [mychannel]: Committed block [8] with 2 transaction(s)
```

3.3 Test case 2

Adding new members to the network requires the new member to receive the resources needed to communicate with other peers securely. Mainly, this means that the new peers need to enroll with the Certificate Authority and receive the required certificate and keys. These keys will be stored locally in the organization's machines. They also need to join the channel in which the rest of the network operates and install the chaincode. When the newly joined peers attempt to instantiate the chaincode on the channel, it will be done in the form of a transaction request that the rest of the network needs to approve according to the endorsement policy. Once the network has approved the member, it can now send requests to other peers and create, retrieve, update and delete values on the ledger.

The time aspect depends on how prepared the system was beforehand. Fabric requires the Docker files to be configured for all members in the network, so adding

new members requires the configuration to already be prepared beforehand. To get a numeric value for the time, a bash script was created and a timer was set to clock the process execution time of adding a new member to the network while the network is running. The resulting execution time to add a new member to the network was found to be around 20 seconds when using the pre-constructed script. The most important functions of the bash script in the correct order of events is shown below.

```
# Generates organization certs using cryptogen tool
function generateCerts() {
        cryptogen generate --config=./organization-crypto.yaml
        ...
}
# Generate channel configuration transaction
function generateChannelArtifacts() {
        configtxgen -printOrg OrgMSP > ../channel-artifacts/organization.json
        ...
}
# Create the configuration transaction needed to add Org to the network
function createConfigTx() {
        ...
}
# Join channel on peers
function joinChannel() {
    peer channel join -b $CHANNEL_NAME.block
    ...
}
# Install chaincode on peer(s)
function installChaincode() {
    peer chaincode install -n mycc -v ${VERSION} -l ${LANGUAGE} -p ${CC_SRC_PATH}
    ...
}
# Instantiate chaincode on peer(s) on channel
function instantiateChaincode() {
```

```
        peer chaincode instantiate ...
        ...
}

function getMedicalRecord() {
        peer chaincode query -C mychannel -n mycc
    -c '{"Args":["queryMedicalRecord","940220-1234"]}'
}
```

3.4 Test case 3

The result of this test case is highly dependent on the endorsement policy that was defined beforehand. A 60% drop-off in network participation can be devastating to the consensus mechanism in some cases, but not in others. If the endorsement policy relies upon endorsing peers that are offline, the consensus mechanism will not execute properly. For example, if the endorsement policy is set up as

AND('Org1.member', 'Org2.member', 'Org3.member')

which requires one signature from each organization, and all peers in, for example Org1 are offline, the consensus mechanism will not execute properly and give an error. However, if the endorsement policy is defined as

OR('Org1.member', 'Org2.member', 'Org3.member')

and all peers in Org1 are offline, the consensus mechanism can still execute properly because it only needs one of the organizations to sign off on the transaction.

3.5 Test case 4

The network was tested with create and retrieve requests between 100 and 500 transactions per second respectively. These parameters were based on an assumption as real data was unavailable. It is however realistic to assume that no more than 500 requests per second will be done for patients needing care in a disaster in a country such as Haiti in a disaster. During the simulation, the logs were reviewed continuously. As Table 2 shows, in all the request cases there were not any failed transactions. For the queries, the tx/s and the throughput was relatively similar. However, in the create requests, the tx/s and the throughput differs significantly. The latency is very high in the create requests compared to the queries.

Table 2: Result of request cases when creating and retrieving medical records.

Type	Success (requests)	Failure	Tx/s	Avg Latency	Throughput (Tx/s)
create	5000	0	100	1.06 s	99
create	5000	0	200	10.89 s	163
create	5000	0	300	15.63 s	164
create	5000	0	400	21.54 s	161
create	5000	0	500	18.29 s	177
create	4975	25	700	23.21 s	152
query	5000	0	100	0.03 s	100
query	5000	0	200	0.01 s	200
query	5000	0	300	0.04 s	300
query	5000	0	400	1.71 s	379
query	5000	0	500	4.65 s	381

Even though the system did not lead to transactions failing until 700 tx/s, the log below shows a trend which is that the unfinished requests are increasing. It means that the system is putting requests on hold in the buffer until it has finished processing the current requests.

```
[Transaction Info] - Submitted: 21464 Succ: 20000 Fail:0 Unfinished:1464
[Transaction Info] - Submitted: 21972 Succ: 20000 Fail:0 Unfinished:1972
[Transaction Info] - Submitted: 22471 Succ: 20000 Fail:0 Unfinished:2471
[Transaction Info] - Submitted: 22970 Succ: 20091 Fail:0 Unfinished:2879
[Transaction Info] - Submitted: 23472 Succ: 20091 Fail:0 Unfinished:3381
[Transaction Info] - Submitted: 23972 Succ: 20091 Fail:0 Unfinished:3881
[Transaction Info] - Submitted: 24473 Succ: 20091 Fail:0 Unfinished:4382
[Transaction Info] - Submitted: 24979 Succ: 20091 Fail:0 Unfinished:4888
```

At 700 tx/s in create requests, the failure of transactions start to occur (see log below). The transactions are dropped and the network responds with ”undefined” since the transaction cannot be found in the ledger.

```
Failed to complete transaction [c38fa...]:undefined
Failed to complete transaction [38906...]:undefined
Failed to complete transaction [c6372...]:undefined
Failed to complete transaction [e9a5d...]:undefined
```

4 Discussion

Before developing a blockchain system, identifying the actual network, the transactions within the network and the assets that are transferred is required. This is important since there are many different blockchain systems that are suited for different use cases. The systematic study of different blockchain systems and consensus mechanisms in this thesis is a result of this. In this case, the network consists of hospitals, a human aid organization and the Ministry of Health that together create, retrieve, update and delete medical records. Since the asset is sensitive information, a public blockchain is not well suited. It would lead to the information being symmetric: everyone in the network has the right to see the same level of information and anyone can join the network. Limiting the members in the network in this case is a preferable feature to enhance network security. The better alternative is a consortium blockchain. Reaching this choice was also a result of the transactions that are done within a network of healthcare actors. To model realistic transactions, a blockchain providing smart contract functionality is required. This is another reason why a more advanced blockchain system like Hyperledger Fabric is suitable.

Still, Fabric has its down sides. A consortium blockchain will never achieve the security and resilience against attacks as a public blockchain will because the network is relatively centralized. Fabric can not scale to the number of nodes as for instance the Bitcoin or Ethereum network because the network will be extremely slow if not erroneous. So there is a trade-off that has to be decided before choosing the blockchain platform: do we value scalability or confidentiality in our business scenario? The security is created by having distributed ledgers which are harder to manipulate. Design choices determine if users are anonymous or have an iden-

tity. For many government applications, identity management will be an important aspect to consider. Blockchain should be connected with identity management systems which might be at the expense of other benefits like privacy and confidentiality. Identity controls will be one of the challenges when having a huge amount of users.

The summary of the different protocols show a relation between node identity and consensus finality: anonymous blockchain networks have probabilistic consensus finality compared to known identity blockchains have immediate finality. This is natural since blockchains with known identities do not need the excessive security precautions because the user identities are known. It allows for the communication to be immediate. And if some actors behaves maliciously, it can easily be banned from the network and if needed, legal actions can be taken. Another interesting relationship is that low scalability allows for high performance, except for PoS which allows both. With a low amount of nodes, the performance can be higher because the latency of the network is be low. But a lower amount of nodes means the security will be affected because it is easier to attack. A network such as Bitcoin is hard to attack with its high amount of users compared to a Fabric network with, say, 5 validators. This is why public blockchain networks are usually considered more secure and resilient against attacks since the attacker will have to convince a lot more users to accept their malicious block.

In test case 1, we see that the gossip protocol solves the problem that occurs in a disaster. If infrastructure losses cause peers to break connection with the network, the rest of the network can still operate and when the peer receives connection, it will get updated on the missing blocks. This will also be immediate since the consensus finality using Kafka in a Fabric network is not probabilistic like PoW or PoS. This assumes that the endorsement policy does not require the peer that has gone offline to participate in the consensus. If that is the case, the policy will not be fulfilled which would lead to consensus failure.

Test case 2 shows that joining new members to the network follows the same rules as a normal transaction. It is dependent on the endorsement policy to approve the member and generate the required resources for the new member to interact

with other peers. It means that the biggest security flaw that can be identified is if the newly joined member mishandles their certificates and keys locally so that an attacker can impersonate them to communicate maliciously with other peers. This means that the attacker can commit malicious information to the ledger and read medical records. The gossip protocol solves the problem if someone without the proper keys and certificates tries to communicate information to other peers, but it can't if the right keys/certs are held by someone who shouldn't have them. That is up to each organization to store them securely.

Case 2 also shows that the time can be significantly reduced compared to the process today. There is no standardized procedure to brief NPOs and other organizations that come to the disaster struck site to aid in the medical response. Using Fabric and preparing the system for a possibility of adding new members in case of a disaster with a pre-developed script, the medical response can achieve greater results and consequently, more lives can be saved.

In test case 3, we see that the endorsement policy is a parameter that decides if the consensus mechanism will execute properly. This shows how important the trust assumption is for the blockchain protocol because understanding the environment in which the system will operate highly affects how the endorsement policy is created. It also shows how important it is to have close contact with users in the industry to get reviews and evaluation of the model. If this study were to focus only on consensus mechanisms without looking at the specific use case of natural disasters, it would be difficult to know how resilient the network would be in a realistic scenario and how to set the endorsement policy so that the consensus mechanism does not behave erratically.

Looking at test case 4, it is clear that the network can handle enough requests in the event of a disaster where in a short period of time, a lot of requests for medical data will be made in a short time period. But as the log shows, the unfinished transactions pile up which means that the network will reach a limit where it can't store transactions in the buffer anymore. This will lead to transactions being dropped and the client will have to submit the request again. It can be viewed as a DDoS attack

on the network. This is especially visible in the create requests where the latency is really high. The latency is higher than the read queries because creating medical records requires much more processing power than reading them. But there is still significant latency in the read queries. This can be a problem as discussed in the FLP impossibility theorem. If a system is distributed, latency will occur which can lead to inconsistency in the data because it's impossible to have a geographically distributed network without latency.

5 Conclusion

Blockchain solves a lot of the problems regarding disaster medical response. It reduces the time to involve new actors in the response, enhances confidentiality of records, gives the patient the ownership of his/her data to quickly share it with consent and it can handle the load of requests for medical records. Knowledge about the effects of disasters are known, yet the systems today are not prepared for them. The information is scattered across databases in different physical locations and when they are destroyed, the data is lost. Blockchain also provides more valid information since the ledger can only be updated through the consensus mechanism which consists of a set of rules that must be followed. Additionally, it can serve as a template for sorting out responsibilities afterwards. Information about who treated what patient at what time is logged in the ledger and in the event of malpractice, the source can easily be found.

Compared to a real disaster scenario, the blockchain can persist even with a relatively high amount of transactions in a short time period. PoW is a consensus protocol that cannot reach the levels of performance compared to consensus protocols in permissioned blockchains such as PBFT and Kafka. PoW cannot tolerate more than 25% byzantine nodes whereas PBFT can tolerate up to 33%. Kafka does not tolerate byzantine nodes but in its implementation in a blockchain it can be adjusted according to the scenario in which it operates.

The parameters that are important and effect consensus resilience and security are:

- Blockchain type: permissioned or permissionless (public or private). If nodes are anonymous, the security will affect the performance since you can't trust anyone to be honest and security precautions will have to be taken to assure that the updated blocks of information in the blockchain are valid. In a permissionless blockchain, the node identities are known and the same security precautions are unnecessary.
- Consensus finality: is the distribution of data in the consensus mechanism probabilistic or immediate? In probabilistic protocols, the delay can cause forking and lead to byzantine behavior whereas in immediate communication, the data is shared immediately which makes the system more secure from attacks.
- Scalability: if the network can scale to a high number of nodes, the network is more secure because an attacker will have a harder time to outperform the rest of the network in a PoW blockchain. In permissioned blockchains, the scalability is lower because the network communication between the nodes will be too slow when
- Byzantine tolerance: at what percentage of byzantine nodes can an attacker take control of the blockchain? This differs a lot between consensus protocols and is a very important parameter.

6 Future Work

What blockchain technology can offer is that instead of the government handling the transactions, they can be handled directly by the P2P network that are enabled by government organizations. In this case, several questions are interesting to study: who will set up the architecture? Who will maintain the blockchain? Most likely, it will still be the government that handles this while the transactions might be

performed without the government. Nonetheless, it is an important subject to research in the blockchain field.

The results from this study makes it clear that emergency medical responses can be greatly optimized compared to today. An interesting study would be to connect with emergency response actors and test if blockchain, or more specifically Fabric, could be used after a disaster has struck.

Fabric is a flexible blockchain where the endorsement policy can be set according to which environment it will operate. However, the optimal solution would be to have a consensus protocol designed specifically for medical services that optimizes the processes according to the problems that occur in the industry today. An interesting yet complex study could make an attempt to create or at least design a solution for this.

References

[1] Coinmarketcap, "Global charts," English, [Retrieved 15-03-2018]. [Online]. Available: https://coinmarketcap.com/charts/.

[2] C. Cachin and M. Vukolic, "Blockchain consensus protocols in the wild," English, *IBM Research - Zurich*, 2017.

[3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," English, 2008.

[4] S. Ølnes, J. Ubacht, and M. Janssen, "Blockchain in government: Benefits and implications of distributed ledger technology for information sharing," English, *Government Information Quarterly*, pp. 355–364, 2017.

[5] M. Jun, "Blockchain government - a next form of infrastructure for the twenty-first century," English, *Journal of Open Innovation: Technology, Market and Complexity*, 2018.

[6] IBM, "Blockchain: The chain of trust and its potential to transform healthcare," English, 2016. [Online]. Available: https://www.healthit.gov/sites/default/files/8-31-blockchain-ibm_ideation-challenge_aug8.pdf.

[7] A. Sarcevic, L. Palen, J. White, K. Starbird, M. Bagdouri, and K. Anderson, "Beacons of hope in decentralized coordination: Learning from on-the-ground medical twitterers during the 2010 haiti earthquake," ACM, 2012, pp. 47–56.

[8] I. Global, "What is a centralized system," English, [Retrieved 15-03-2018]. [Online]. Available: https://www.igi-global.com/dictionary/centralized-system/3590.

[9] I. Bashir, *Mastering blockchain*, English. Packt, 2017.

[10] D. Tapscott and A. Tapscott, *Blockchain revolution*. Portfolio, 2016.

[11] C. Dannen, *Introducing ethereum and solidity: Foundations of cryptocurrency and blockchain programming for beginners*. Apress, 2017.

[12] V. Buterin, "On public and private blockchains," 2015 (Retrieved 12-03-2018). [Online]. Available: https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/.

[13] D. E. O'Leary, "Configuring blockchain architectures for transaction information in blockchain consortiums: The case of accounting and supply chain systems," *Intelligent Systems in Accounting, Finance and Management*, pp. 138–147, 2017.

[14] N. Popper and S. Lohr, "Blockchain: A better way to track pork chops, bonds, bad peanut butter?," 2017. [Online]. Available: `https://www.nytimes.com/2017/03/04/business/dealbook/blockchain-ibm-bitcoin.html`.

[15] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. bft replication," English, 2016, pp. 112–125.

[16] G. Greenspan, "Avoiding the pointless blockchain project: How to determine if you've found a real blockchain use case," English, 2015 [Retrieved 16-03-2018]. [Online]. Available: `https://www.multichain.com/blog/2015/11/avoiding-pointless-blockchain-project/`.

[17] Dictionary, "Consensus definition," [Retrieved 02-05-2018]. [Online]. Available: `http://www.dictionary.com/browse/consensus`.

[18] W. Mougayar, *The business blockchain: Promise, practice, and application of the next internet technology*. 2016.

[19] R. S. L. Lamport and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, pp. 382–401, 1982.

[20] A. Baliga, "Understanding blockchain consensus models," *Persistent*, 2017.

[21] M. Fischer, N. Lynch, and M. Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the ACM (JACM)*, pp. 374–382, 1985.

[22] C. Dwork, N. Lynch, and L. Stockmeyer, "Consensus in the presence of partial synchrony," English, *Journal of the ACM*, pp. 288–323, 1988.

[23] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," English, 2014, pp. 436–454.

[24] V. Buterin, "A proof of stake design philosophy," English, 2016 (Retrieved 16-03-2018). [Online]. Available: `https://medium.com/@VitalikButerin/a-proof-of-stake-design-philosophy-506585978d51`.

[25] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," English, *ACM Transactions on Computer Systems (TOCS)*, pp. 398–461, 2002.

[26] T. T. A. D. et al., "Untangling blockchain: A data processing view of blockchain systems," pp. 2–3, 2017.

[27] Apache, "Introduction," English, Retrieved 14-03-2018. [Online]. Available: `https://kafka.apache.org/intro`.

[28] ——, "Introduction," English, Retrieved 14-03-2018. [Online]. Available: `https://zookeeper.apache.org`.

[29] Hyperledger, "About," English, Retrieved 17-01-2018. [Online]. Available: `https://www.hyperledger.org/about`.

[30] ——, "Hyperledger fabric documentation," English, Retrieved 15-03-2018. [Online]. Available: `https://hyperledger-fabric.readthedocs.io/en/release-1.0/`.

[31] ——, "Gossip data dissemination protocol," Retrieved 25-05-2018. [Online]. Available: `https://hyperledger-fabric.readthedocs.io/en/release-1.1/gossip.html?highlight=gossip`.

[32] Investopedia, "Healthcare sector," English, 2018 [Retrieved 19-03-2018]. [Online]. Available: `https://www.investopedia.com/terms/h/health_care_sector.asp`.

[33] Capgemini, "Insights into blockchain: Opportunities and challenges across multiple industries," English, 2017. [Online]. Available: `https://www.capgemini.com/wp-content/uploads/2017/07/audience_insights-blockchain.pdf`.

[34] A. E. et al., "A case study for blockchain in healthcare: "medrec" prototype for electronic health records and medical research data," English, 2016.

[35] IBM, "Healthcare rallies for blockchains: Keeping patients at the center," English, 2016.

[36] P. T. et al., "Personal health records: Definitions, benefits, and strategies for overcoming barriers to adoption," English, *Journal of the American Medical Informatics Association*, 2006.

[37] D. C. C. et al., *Emergency medical responses: Clinical practice and systems oversight*. 2016.

[38] J. Rohr, "Blockchain for disaster relief: Creating trust where it matters most," English, 2017 [Retrieved 19-03-2018]. [Online]. Available: `http://www.digitalistmag.com/improving-lives/2017/11/23/blockchain-for-disaster-relief-creating-trust-where-it-matters-most-05527536`.

[39] S. J. Tracy, *Qualitative research methods: Collecting evidence, crafting analysis, communicating impact*. Chichester, West Sussex, UK;Malden, MA; Wiley-Blackwell, 2013.