Blockchain in maritime cybersecurity

UNIVERSITY OF TURKU
Department of Computing
Master of Science in Technology Thesis
Networked Systems Security
December 2021
Emilia Pitkänen

Supervisors:
Meri Löfman (Brighthouse Intelligence)
Petri Sainio (University of Turku)
Antti Hakkala (University of Turku)

UNIVERSITY OF TURKU
Department of Computing
EMILIA PITKÄNEN: Blockchain in maritime cybersecurity
Master of Science in Technology Thesis, 55 p., 0 app. p.
Networked Systems Security
December 2021

Autonomous vehicular networks and safety of remotely controlled vehicles have been a widely researched topic in the last years. Smart environments have grown popularity amongs homes but in larger environments as a smart fairway system in maritime environment are still new research topics.

Sea for Value (S4V) is a multi-organization project that aims to develop a remote pilotage system within a smart fairway environment. This complex environment has many participating nodes and their cybersecurity needs to be thought carefully in terms of confidentiality, integrity and availability. Many cyber threats exist in large networks with mainly wireless communication methods which makes the remote pilotage especially vulnerable system.

Blockchain technologies have been used in various financial use cases such as creating cryptocurrencies but they have now gained more ground in different varying industries such as energy, transportation, identity management and digital signature management. Because of their decentralized architecture they create a tamper proof secure ledger with no single point of failure. These features can provide solutions for security, integrity of data and connection methods for industries and organizations that need a way to handle large amounts of data and create better solutions for privacy protection and user authentication. As new blockchain technologies are constantly evolving and new ones are being developed the industry keeps growing and changing in a fast paced manner which means keeping up with new blockchains is a constant work.

This thesis is a comparison research of three of the 2021's biggest open-source Ethereum based protocol solutions and an operating system. These protocols are compared with each other keeping in mind S4V project's requirements and environment and a possible solution is proposed.

Keywords: blockchain, maritime cybersecurity, Ethereum, IoT, cyber threats, smart contracts, fog nodes, permissioned blockchain networks, Quorum, Hyperledger Fabric, Corda, Firefly

Table of Contents

Abbreviations and Acronyms

S4V	Sea for Value
BHI	Brighthouse Intelligence
PoW	Proof of Work
PoS	Proof of Stake
PoA	Proof of Authority
IoT	Internet of Things
IP	Internet protocol
VPN	Virtual Private Network
SHA	Secure Hash Algorithm
P2P	Peer-to-peer
ECDIS	Electronic Chart Display and Information System
VTS	Vessel Traffic Services
DLT	Distributed Ledger Technology
PKI	Public Key Infrastructure
BLOB	Binary Large Object

1 Introduction

Autonomous transportation in maritime industry is a widely discussed topic and its development is expected to take leaps forward in the forthcoming years. Dimecc's program Sea for Value (S4V) [1] is one of these projects aiming for more efficient, sustainable, autonomous and safer maritime transportation. Its mission is to create service innovation for remote operations and to prepare an advanced remote pilotage system and autonomous navigation system. S4V is a large program with many partnering companies and its first steps towards this goal is a safe fairway of which ships can leave and arrive to harbour. The remote pilotage system consists of multiple hardware and software components that are distributed on the remote pilotage station and sensors on the fairway and vessel. This distributed architecture relies on secure communications from trusted parties.

Remote pilotage system faces still many unanswered questions about overall security which is handled by Brighthouse Intelligence in collaboration with other stakeholders in this "S4V Fairway" project. This Master's thesis in Technology researches the possible secure communication and authentication methods between remote pilot, bridge team and fairway sensors by using blockchain technologies.

Research questions that this thesis tries to find answers for are:

1. What benefits can the researched blockchain technologies bring to the S4V project?
2. What blockchain technology is the best for authenticating messages and data transactions between remote pilot, bridge and other fairway intelligence?
3. Can blockchain create more secure communication methods compared to VPN and if yes, how?

Because of the rapid development of blockchain technologies the scope and focus of the thesis changed during research phase. Research questions developed over the time whilst writing this thesis and the scope was re-defined since implementation and testing methods that initially were to be done got out of scope. Especially the second question about

authentication mechanism evolved from message authentication to more about authenticating new vessels and other nodes to the fairway system.

Rest of the thesis is structured as follows: Second chapter discusses current blockchain technologies and gives a general idea of their architecture and applications. Third chapter describes the remote pilotage system and fairway more in detail and defines system requirements and architecture. In the fourth chapter possible blockchain technologies for usage are analyzed and compared. Last two chapters describe results of research and conclusion of the overall thesis.

On a side note this thesis describes only the basics of the blockchain technology behind cryptocurrencies and does not discuss the financial aspect of them.

2 Blockchain

National Institute of Standards and Technology Internal Report (NISTIR) 8202 "Blockchain Technology Overview" [2] defines blockchains as "tamper evident and tamper resistant digital ledgers implemented in distributed fashion without a central authority". In the rest of the thesis term blockchain and ledger are used interchangeably. Transactions in blockchain are grouped in data structures that are cryptographically hashed together and distributed over a peer-to-peer network. These data structures are called blocks. Distribution of blocks over the peer-to-peer network guarantees resilience because no single point of failure exists. Blockchain technology provides efficiency in tracking data logs and their security, transparency of data between all users and easier device management and data synchronization.

2.1 History

In 2009 the concept of blockchain technologies emerged from Satoshi Nakamoto's whitepaper "Bitcoin: A peer-to-peer electronic cash system". [3] The paper described a financial system called Bitcoin that eliminated third parties' involvement. Nowadays Bitcoin is one of the largest cryptocurrencies in the world and probably the first thing people think when the word blockchain is mentioned because it was the first of many blockchain applications [2]. However blockchain is the technology behind these cryptocurrencies and it is also used in many other fields, than financial services, such as manufacturing, healthcare and energy industry. Blockchain technologies' core ideas originate from the late 1980s and 1990s when Leslie Lamport developed a consensus method for a case when networks or computers may be unreliable [2]. This was later applied in the development of Bitcoin. Popularity of Bitcoin was based on blockchain's distributed architecture so no single user managed all the money.

Many reasons lead to the quick increase of blockchain's popularity. Eventhough cryptocurrencies were the drivers in blockchain technologies many other industries quickly discovered its possibilities. Technologies developed quickly when organizations started researching other use cases for blockchain such as logistics, transport operations and global supply chains. Many applications for blockchain have been developed for

IoT authentication technologies since the amount of data they process is not efficiently managed in a centralized system [4]. Blockchain can be used for many different purposes for handling large amounts of data and creating better privacy protection, user authentication, tamper proof data which lead to growing interest among industries. NISTIR8202 summarizes that blockchain technology could be a suitable option if needed features are such as many distributed participants, transactional nature of workflow, cryptographically secure system, monitoring real time transactions, logs of full transaction history and mainly the want of working without a trusted third party. It also defines that blockchain is probably not the best solution if stored data has to be modified, if sensitive data such as personally identifiable information is stored or if you have only one entity that contributes to the storage or one entity is trusted enough to act as a trusted third party [2]. Blockchains have developed very quickly since NISTIR8202 report when they were only in their early stages. As blockchain technologies are changing rapidly and no one knows where they develop in the next five years it is important to firstly consider thoroughly whether it is even a viable option for some organizations.

2.2 Technology

Blockchain enables the possibility to record any transaction in a decentralised database whether it is a value, data or any other asset. This is done by a ledger that works in peer-to-peer network in which initial purpose was to remove third parties from financial transactions to create safer system. This technology is now implemented in many other industries. By removing a trusted third party the trust has to be enabled with four key characteristics in blockchain networks. These characteristics are defined by NISTIR8202 [2] as ledger, secure, shared and distributed. Ledger acts as a database which works on only append and transactions cannot be removed or overridden. The data is cryptographically secured which guarantees that the data cannot be tampered with. Shared and distributed characteristics provide more transparency and scalability to the network and guarantee that no single point of failure exist.

Blockchain is a digital ledger constructed by nodes, end-users and blocks. Each computer acting as a node needs to run a software application spesific to the blockchain.

Rajnees Gupta's book "Hands-on Cybersecurity with Blockchain" divides blockchain to five functional parts: 1. transaction preparation, 2. transaction verification, 3. block generation, 4. block validation and 5. block chained. In the first part one party creates a transaction containing receiving address, digital signature etc. This is shared to all participants of the blockchain. When this transaction is verified in the second stage its digital signature is verified by all nodes in the blockchain with the senders public key. After verification all queued transactions are transformed into a block in one of the network nodes. In the fourth part the block is validated by all the other nodes using some consensus method such as Proof of Work (PoW), Proof of Stake (PoS) or Proof of Authority (PoA). In the last part the block is chained to a blockchain after all nodes reach consensus.

2.2.1 Block and ledger

To divide blockchain into more spesific smaller technological parts we will start with a block. A block can be represented simply like an IP packet. As IP packet consist of an IP header and a payload so does the block similarly consist of a block header and block data. The header of a block has metadata such as version, timestamp, nonce, hash of previous block, Merkle root and time. Nonce keeps track of PoW algorithm and Merkle root is a hash of block transaction's Merkel tree root. Block body consists of all transactions made. It adds a digital signature and a public key to each of the transaction on the block body to identify destination [3].

A ledger consists of transactions. Usually these ledgers have been stored in a large centralized database that have been managed and secured by a third party. A blockchain ledger is distributed among many owners and computers. The distributed ownership and architecture enables better trust, reliablity and security than in centralized systems.

A node is an actor on the blockchain network that can publish new blocks, store a partial copy of the ledger or store a copy of the whole ledger and all transactions that have been made. It can submit new blocks to the blockchain and verify or reject new ones. Nodes arrange all blocks in the ledger to a chronological order to keep track of transactions. Each node can be different in software or hardware capabilities which makes them more resilient to attacks because one attack that targets a node probably wont work on another node.

Each block is chronologically connected to the previous one by having the hash of a previous block's header in their header. If a previous block is changed the hash would be different which correlates to all subsequent blocks to have different hashes.

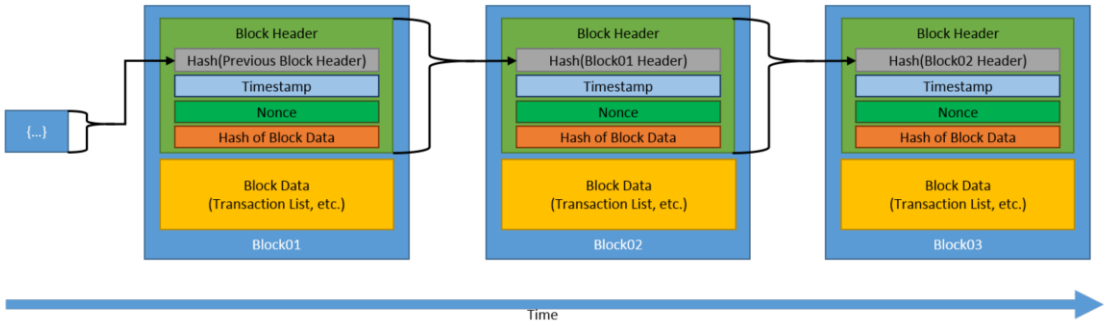

Figure 1. Description of chaining blocks together. [2] p.17

2.2.2 Cryptography

One of the most important aspect of blockchain is its cryptography. Hashing creates a backbone for blockchain security and immutability of data. They are used for encrypting data by creating a fixed-length output for a certain input [3]. Hash functions have important security aspects because they are one-way algorithms, collision resistant and they cannot produce a spesific output with any given input (in mathematical terms: for given x, finding y such that hash(x)=hash(y) is not computationally possible with today's hardware and computational power) [2]. This ensures that only one input matches to the output, no two different values produce the same output and the input value cannot be found with only output value. Ethereum's Keccak-256 and Bitcoin's secure hashing algorithm with a 256 bit output (SHA-256) are mainly used in blockchains and are NIST-approved. They belong to SHA-3 and SHA-2 families of hashing algorithms. These are used by creating a hash value for the transaction that will be sent and the receiver can always compare the received hash value to the calculated one to verify its correctness. Cryptographic hashing is used for securing block data and block header, and creating an immutable chain of blocks since every block header contains the hash of previous block's header.

Digital signatures are used for verification of the sender. When a transaction is made the sender uses asymmetric key pair to sign the transaction with a private key. The receiver can then verify transaction with a public key. Digital signatures are already

in wide use in many organizations to prevent forging messages and provide authenticity and integrity of transported data. Because each node connected to a blockchain has administrative rights, it is important to authenticate senders that no invalid data is added to the ledger.

In addition to hashing all transactions are secured with a public and a private key in digital signatures. This is called asymmetric cryptography or public key cryptography. The public key can be published without reducing security of the cryptosystem but private key must be kept a secret. Transaction sender will encrypt data with the private key and receiver can decrypt it with the public key. This algorithm ensures trusted verifying of transactions without sender and receiver having to know or trust eachother. Blockchains use asymmetric cryptography for signing transactions with private keys, using public keys to verify digital signatures created with private keys and verifying that sender has the private key to sign transactions. Some permissioned blockchains can even use organization's existing asymmetric cryptography public keys to provide credentials by using existing directory services and sharing them to blockchain certificate authority. Blockchains store private keys securely in a software called a wallet. It is necessary that private keys are stored securely since if a user loses a private key all transactions correlating to that key will be lost. If the private key is stolen an attacker will have full access to all digital assets associated with it. Because of blockchain's immutable nature if the attacker decides to transfer all private key associated data to another account the transaction cannot be undone. This has lead to some organizations having an external hardware safe for private keys for it is crucial to keep them secure.

2.2.3 Consensus

Blockchain has a collision problem when multiple nodes try to publish different blocks at the same time. This can be solved with a consensus model. Many consensus models have been created for blockchains such as Proof of Work (PoW), Proof of Stake (PoS), Proof of Authority (PoA). They aim to define which user/users can publish the next block and create agreement on publishing system in a distributed environment. When users join the blockchain they all agree to existing policies created by the consensus model and initial system state. This ensures that all important aspects are taken into

consideration before actually joining them to the blockchain. The initial system policies are in a pre-configured block called the genesis block which starts the blockchain. Combining the initial state after genesis block and verification of each block, every user can agree to the current state of the blockchain. However temporary disagreement can happen but it is necessary if sudden changes in the blockchain happen and they must be act upon. Consensus methods are crucial for public blockchain networks because no trust exists between users. However in permissioned blockchains there exists some level of trust and in private only trusted parties can access the blockchain. This means that in these blockchain models consensus methods do not have to be so computationally heavy.

Proof-of-Work (PoW) consensus model bases on computational power. The node which solves a hard mathematical puzzle first will be the one that publishes the next block. Common puzzle is requiring that the hash of the next block header is smaller than a target value [2]. It is easy to check that a hash digest is correct but creating the hash is hard which helps other nodes to check its correctness. Bitcoin uses this consensus model and changes the target value every 2016 blocks to adjust publishing rate. Adjusting the target value keeps Bitcoin's blockchain secure by increasing difficulty of the puzzle over time. This ensures that no single node can take over all publishing and block creation. Depending on hardware and difficulty of the puzzle it can take only seconds or even hours to create the correct hash. Because of this many nodes have organized themselves into pools that work together on solving the puzzle. After a block is published and it has been verified, the block is distributed fast across the network to all participating nodes. In Bitcoin network the node that publishes a block gets rewarded with a certain amount of bitcoins.

Proof-of-Stake (PoS) bases on stake that a user has invested into the system and decision of which node can publish next will depend on it. This model differences from PoW by not having to perform computationally intensive work that demands lots of resources. PoS networks can be created the way that no new cryptocurrency is created at all. Ethereum will use this consensus model in the future even though right now Ethereum utilises PoW model. Blockchain network can use the stake in four ways: random selection of staked users, multi-round voting, coin aging systems and delegate systems [2]. These all are decision making systems for the next publisher.

Proof-of-Authority (PoA) or sometimes proof of identity is based on some level of trust between users that are linked in the real world. Nodes that wish to publish have to have their identities proven and verifiable in the blockchain. Publishing blocks is based on publishing node's reputation on behaviour. Other users disagreing with the node will affect its reputation which will lower its possibility to publish blocks in the future. This consensus model can be used only in permissioned or private blockchains since trust between users is essential where as PoW and PoS can be used in permissionless public blockchains. PoA is used in private Ethereum blockchains and it guarantees fast confirmations of transactions.

2.3 Different blockchains

As earlier mentioned different types of blockchains exist: public, private and permissioned which is a hybrid solution of public and private blockchain. At the start of blockchains' history all were public and open to everyone. In public blockchains anyone can publish a block but in private and permissioned only certain users can publish. Private and permissioned blockchains are usually deployed by organizations that can trust users participating in the blockchain and do not require mining of cryptocurrencies.

Public blockchains such as Bitcoint and Ethereum are open to public and anyone wanting can publish a block by using an open source software. Because anyone can read the whole ledger of transactions and write new transactions malicious participants can also access the blockchain. They can try to take use of the system to gain monetary profit or to subvert it [2]. To prevent this many public blockchains use a consensus system that requires maintaining or expending some resources such as processing power. Bitcoin was the first large decentralized platform but Ethereum has widened their scope to other applications also. They have created smart contracts that can be used to satisfy different restrictions to blockchain clients.

Private blockchains are maintained by single organizations that restrict participants in the blockchain to only trusted ones. Usually PoA consensus model is used since all joining parties can be trusted initially. This means that no additional computational resources have to be used when joining a private blockchain. As only certain users can join and authority is maintained by one or couple of organizations

private blockchains are not as decentralized as public blockchains. It is however necessary because usually these blockchains are used for distribution of confidential information. All participants that join private blockchains have to abide by organizations' rules and regulations. If a user misbehaves they can be removed from the blockchain or write access can be revoked.

Depending on the level of restrictions and regulations users have to abide private blockchains can be defined as permissioned, hybrid or consortium blockchains. This means that in some cases users can read the whole ledger but cannot participate in it or a user can have their write access denied for certain amount of time. They can also regulate whether a user can join via open source software or only via closed source software [2]. Participating users have to have some level of trust on each other but permissioned blockchain could be used for example in a case where organizations have to work with each other but do not completely trust one another. It is especially suitable when transparency and immutability of transactions is needed. In summary permissioned networks are the hybrid model of public and private blockchains in which authorized organization can decide the level of control and openness of the network.

2.4 Security

Cyber threats evolve fast and new vulnerabilities are found daily. Security ecosystem has evolved along with emerging threats and a zero-trust approach has evolved. Zero-trust approach means that nowadays systems and their security should be built assuming that a breach will eventually happen at some point. This means that not one user or node is trusted without a proper authentication and verification, and minimum priviledges are assigned to every participant in the network. The policy identifies sensitive data, maps data flow, creates a policy base and monitors networks [3]. This idea can be implemented in a blockchain network with separating parts of the network via smart contracts or fog nodes.

Blockchain is fundamentally a tamper proof ledger that prevents man-in-the-middle and denial-of-service attacks. Decision between what type of blockchain to use defines how to trust the acting parties, whether it is a PoW, PoS or PoA model. Trust is

a necessary factor of the blockchain network and it must be guaranteed with a consensus model as described before.

2.4.1 Smart contracts

NIST cybersecurity whitepaper [5] defines smart contracts as "a collection of code and data that is deployed using cryptographically signed transactions on the blockchain network". All nodes in the network abiding the same smart contract must derive same results from execution which are recorded in the blockchain [5]. This means that smart contracts are deterministic, all participants in the smart contract must agree to the new state of the blockchain after each execution. This minimizes malicious and accidental execution and other exceptions when contractual conditions are met. In permissioned or public blockchains smart contracts can be deployed the way that nodes outside of the contract can suggest code execution but by restricting the contract conditions such as time-limit and capacity-limit some executions can be denied or interrupted. This prevents malicious users from creating a denial-of-service attack to the system. Not all blockchains can run smart contracts but for example an Ethereum smart contract can be created to form a trusted "bubble" where only certain nodes can act. It is one of the most popular smart contracts since it provides scalable processing capabilities [3]. Each contract must be hand written code in which developer can choose all restrictions and access models in it. For example an Ethereum smart contract can be built with Solidity language which is similar to Javascript, and it can be compiled with an online compiler Remix IDE which makes the creating of a contract user friendly.

Trusted "bubbles" created with smart contracts are virtual zones in which devices can communicate securely over a non-trusted network. All devices in the network only communicate with each other and assume that outside participants are initially malicious (zero-trust approach). Depending on the configuration of the smart contract it can be limited to only some outsider participants or none at all. If no outside participants are allowed to act inside the smart contract nodes it is a partially private network which makes adding new nodes harder.

These bubbles do not define which kind of devices can join and adding new devices is easy when smart contract allows outside participants to join the network [4]. However one smart contract over the whole system consisting of admins, end users, fog

nodes and cloud is easier to use. Smart contract includes a list of registered devices and their associated IoT devices. All access control and authentication functionalities are managed inside the smart contract. Smart contract is created by a “Master” node which can be thought similar as a certification authority. After master node all other nodes will be follower nodes that implement an Elliptic curve cryptographic key pair. SHA-3 Keccak hash is used for the public key to keep it secure. Signatures in the smart contract are called Elliptic Curve Digital Signature Algorithms (ECDSA) and are thought to be more secure than RSA algorithms in IoT devices [4].

For enterprises that do not want to focus on private networks only, permissioned network solutions such as Hyperledger Fabric are good since they base on trust of having known participants only in the network and they do not demand a payment for joining into a smart contract. Other methods of preventing malicious behaviour are then applied such as revoking access or time-limits [2].

2.4.2 Fog nodes

In addition to smart contracts fog nodes can be implemented to a blockchain network. Fog nodes are a part of the blockchain and one node acts as a data storage for a group of selected sensors. This way sensors are not part of the blockchain, only their data is which increases scalability and eases the workload on sensors [6]. As R. Almadhoun etc. suggest in [6] “a decentralized and scalable authentication mechanism that utilizes blockchain-enabled fog nodes with connectivity to Ethereum smart contracts for authenticating user access to IoT devices whereby access tokens are issued by the smart contracts with no intermediary or trusted third party” would be a viable solution for multi-party fairway systems that rely on sensors on the fairway.

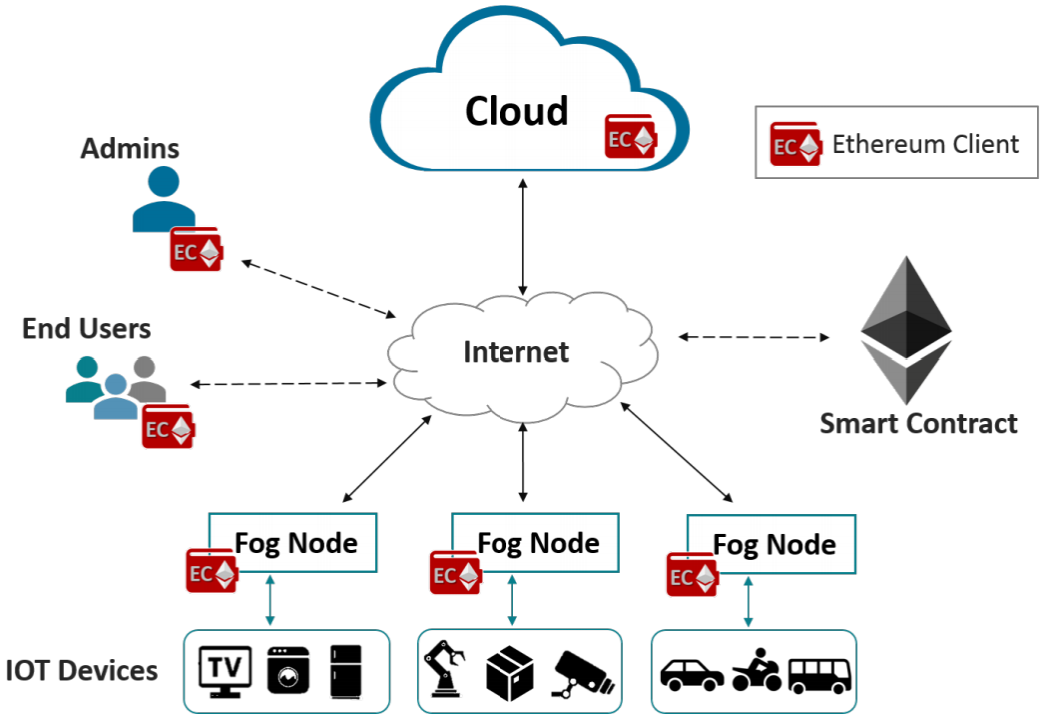

Figure 2. Proposed architecture of fog node enabled network [6]

This figure however is a purely hypothetical figure of how a fog node enabled network could work in a multi-party environment. Most essential part of fog nodes is the labour of computational work they lift from actual sensors on the fairway. On-chain devices can propose changes to the network immediately but off-chain devices need to request access. This access is controlled by the smart contract which will allow or deny access to a certain IoT device depending on the signed message that has the information of user public key and smart contract token which are encrypted by its private key. With this information the fog node can verify or deny this user by verifying its signature and attributes of the token received with the fog node event.

All fog node function handling happens inside the smart contract. The smart contract has to implement three main functionalities: addition and deletion functions and events, and authentication functions and events [6]. Via properly configured smart contract confidentiality, integrity and availability can be ensured if proper authentication and encryption schemes are provided. Confidentiality is essentially provided by blockchain architecture that relies on PKI-architecture and SSL-communications. To ensure integrity of data and prevent Man-in-the-Middle or replay attacks each message in the blockchain are cryptographically hashed and nonce are added with a timestamp. Availability also is ensured in Ethereum solution since it is resilient to DoS attacks and other service disruption attacks.

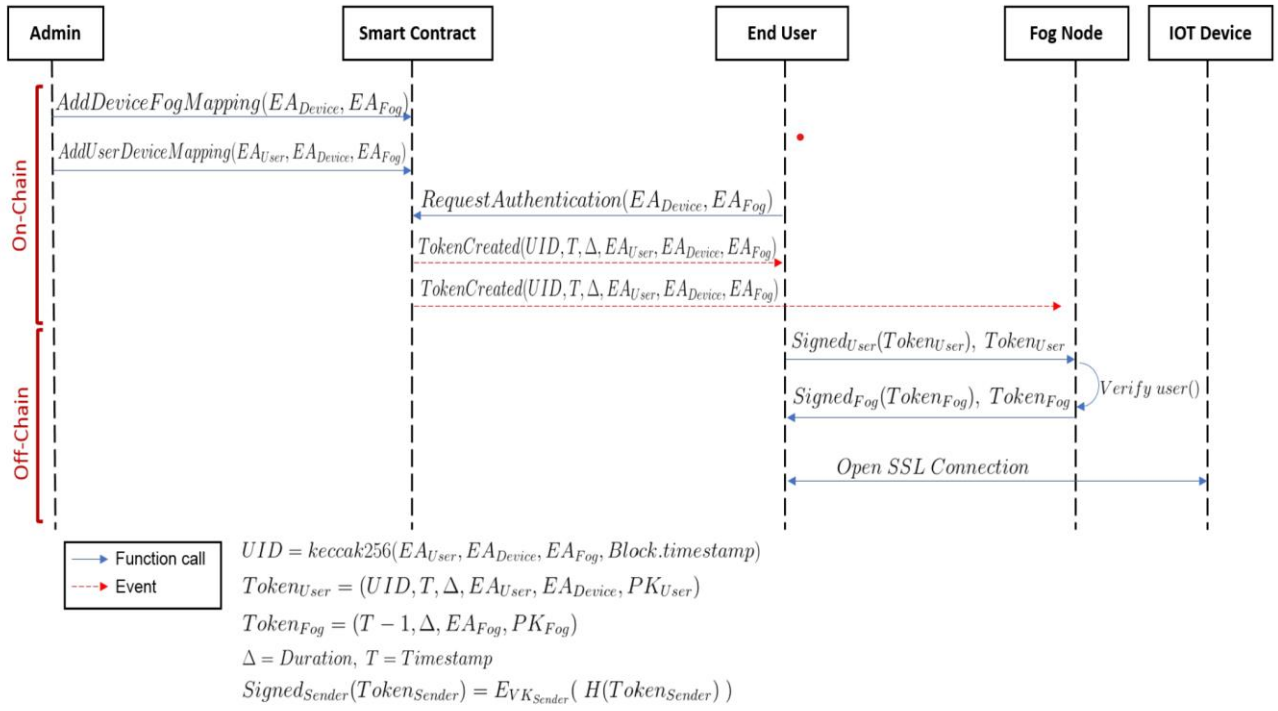

Figure 3. Proposed authentication sequence between end-user and IoT device [6]

Authentication scheme for IoT devices is important to ensure secure connection between the end-user and IoT device. R. Almadhoun etc. suggest the authentication scheme shown in figure 3. Inside smart contract the admin has registered all participating IoT devices and mapped them to a certain fog node. This list of authorised IoT devices is controlled by the admin and if an end-user wants access to a device an authentication request using the devices address has to be made to the smart contract. If

Ethereum smart contract is used then every device has their own unique Ethereum address (as seen in figure 3). When the user is authorised to access the IoT device the event will be created and broadcasted across the network. This event will log all available information such as user id, fog node and device address and block timestamp using the Keccak256 hashing algorithm. Fog node can also restrict the time a user can access the device but after a successful authentication a normal SSL connection will open between the user and the IoT device.

2.4.3 Attacks against blockchains

Ethereum Classic was under attack on January 2019 when an attacker gained access on over half of working nodes in the network and started to rewrite transaction history [7]. This made it possible to spend gained cryptocurrency more than once and the attack was noticed only after 1.1 million of dollars was gained. Overall attackers have succeeded to gain over 2 billion dollars worth of cryptocurrencies since 2017 which leads to growing concerns about safety of public blockchains. As blockchains have many unique qualities security-wise they also have very unique vulnerabilities and risks that have not been yet fully researched. Many attack vectors exist because in addition to the ledger blockchains work by downloading a software client that works as a node. For example Bitcoin's main client Bitcoin Core had to fix a bug in September 2018 that could have lead to millions of fraudulent bitcoins being mined by attackers. However private and permissioned blockchains that are set up correctly are very difficult to hack and not so financially alluring to attackers since no cryptocurrency is being mined.

3 Maritime environment

Cybersecurity in maritime industry grows in importance as remote pilotage systems and autonomous vessels are being developed rapidly. It is predicted that in 2022 autonomous docking and transit could be possible and by 2030 a fully autonomous vessel could be on route [8]. These remote and autonomous vessels are very sensitive to cyber attacks because of their complex systems. Secure gateways between remote vessel, fairway sensors and pilotage system must be reliable and data confidentiality, availability and integrity must be guaranteed.

Maritime industry has many attack vectors for remote vessels because both information technology (IT) and operational technology (OT) must be secured [9]. IT in maritime consists of networks, administration, management systems and electronic certificates. OT instead consists of propulsion, ballasting, navigation and communication systems. In IT mainly reputation and finance are at risk but in OT environment, property and even life is at risk [9]. Vulnerabilities in remote vessels lie in bridge systems, communication channels, access control systems and in operational systems such as power control systems and machinery management systems. These vulnerabilities consist of human error and hardware malfunctioning. Especially data from different sensors might be faulty sometimes. Unreliable data might be generated unintentionally or by accident if camera, lidar or other sensor data is misinterpreted.

IoT systems can be targeted with many different kind of attacks such as denial-of-service (DoS), man-in-the-middle (MITM), message or executable code injection, authentication tampering and GPS spoofing which all are hazardous for a remote pilotage system. These attacks take advantage of the system's wireless nature but physical environment has to be also secured. All these attacks will have severe impacts. By attacking the physical layer attackers could modify sensors' data, network attacks could enable hackers to have remote access and inject malicious instructions and by targeting application layer of the pilotage system sensitive data could be leaked. In most severe cases ships could be held as hostage if ransomwares start targeting these new systems. All these flaws in security may lead to financial loss, reputational damage, loss of customer and industry trust and environmental damage.

Defence in depth model ensures layered physical, network and system security. Network security protects connections between zones and system security provides encryption, authentication, backup and recovery protection. Blockchain can solve many of these issues by creating a tamper proof private ledger where all actions are recorded. It can be used for many purposes in remote pilotage systems such as authentication system for messages, secure communication channel between bridge team, pilot and sensors on the fairway and a trusted log of events.

3.1 S4V system architecture

The most important aspect of remote pilotage system is its communications. When remote pilotage starts secure communication channel must be established between the vessel and remote pilot and attending parties must be authenticated. This authentication could be completely automatic but before that a security token is needed to provide reliable authentication methods. After successful authentication all connections have to be reliable and always available since the pilot of the vessel is not physically on the bridge. There are many ways that communications between pilot and bridge team can happen: VoIP, microphone, ECDIS-type of view etc. Environmental conditions are currently available from public interfaces but in the future more accurate sensor data helps vessel navigation. Sensor data from the fairway is useful when the vessel cannot provide enough accurate information and if vessels GPS is jammed or spoofed.

S4V fairway has a complex architecture that can be best explained in high abstraction level in figure 4. Many different operating parties, software and hardware modules that are distributed on the fairway and vessel make the environment complicated to manage.

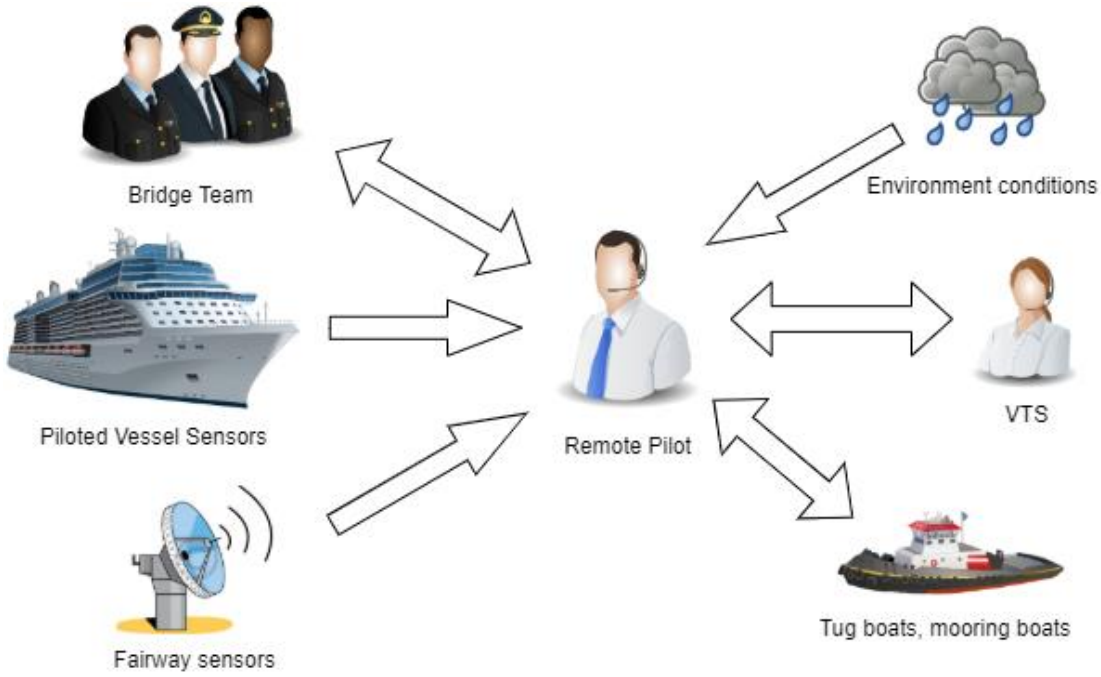

Figure 4. Complex environment of a remote vessel [10]

Implementing a remote pilotage system to this complex environment comes with many yet to be solved issues such as proper communications between bridge team and pilot, missing standards and procedures and tackling with human error. Huge amounts of data is being handled in maritime environment. Different parties communicate with each other remotely and in a fast paced nature and the importance of reliable data increases when adding a remote operating system. Figure 4. represents the main parties that exchange information with the remote pilot. Also other vessels on the fairway could be attending to this communications but we will assume that is not the case yet. Figure 5. explains the data architecture more in depth.

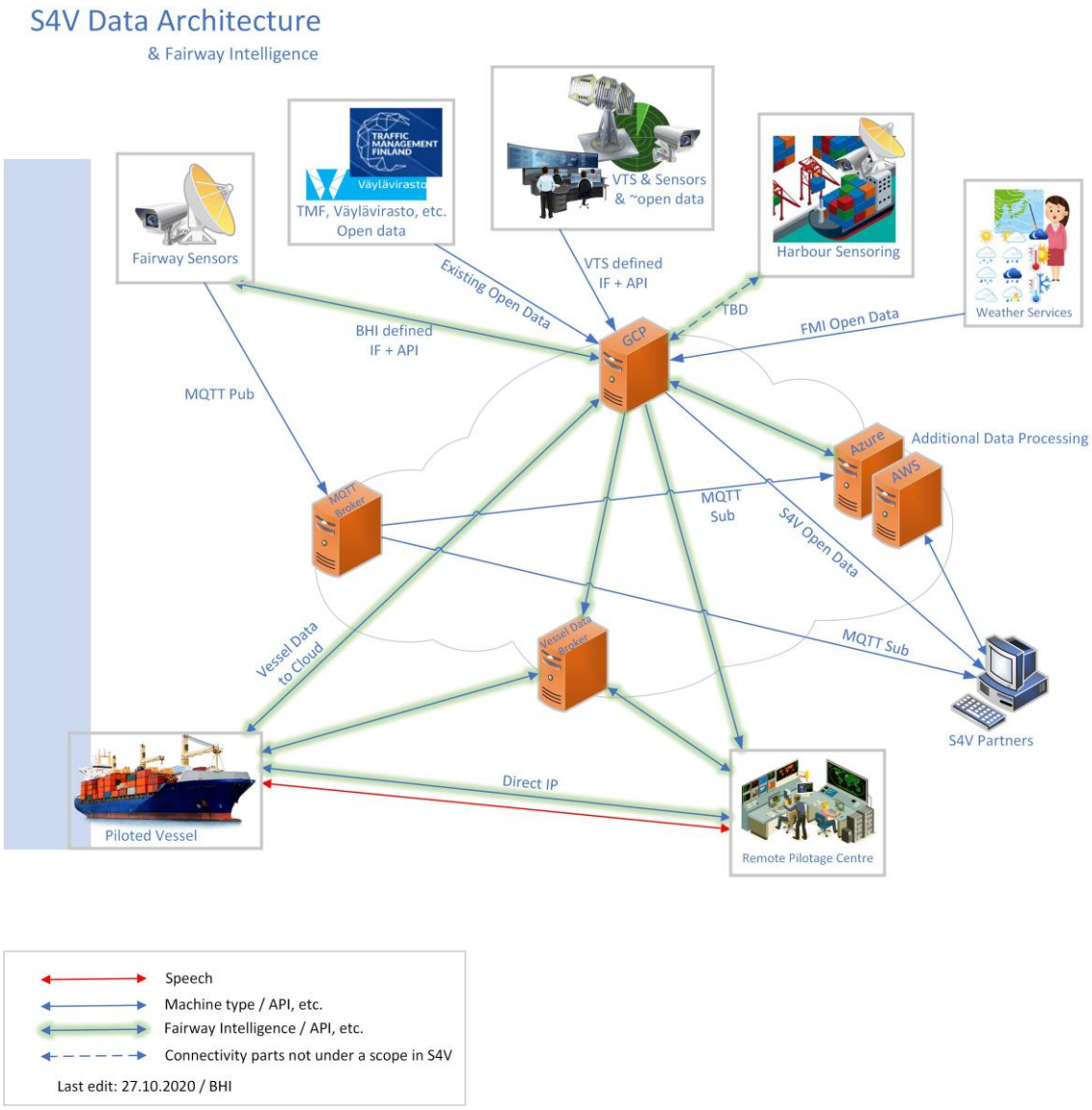

Figure 5. Communication architecture [10]

In complicated communication architecture as shown in figure 5, data is received from multiple sources such as onboard sensors, external sensors and other parties participating in the fairway network. Connections for transporting all data must be secure. Participating end users must be authenticated to ensure that communication is taking place with the intended party. Communication between all attending parties must be encrypted and done via secure protocols. Networks are segregated with firewalls and password protection is enabled to create more secure wireless environment. Intrusion detection systems and intrusion prevention systems are placed within the networks and to some hosts to detect anomalies and prevent security flaws or attacks.

Fairway and vessel data are sent to bridge team and remote pilot however they do not have a shared view per se. They both have an ECDIS-type of view and if it is truly shared communications could be established through it. Markings could be done to ECDIS-view and these “commands” could be the new “language” between pilot and bridge team. Trust is necessary to be established between pilot and the bridge team so that sudden changes can be verified to create an undisputable communication and logging system. On the fairway environmental conditions can change rapidly and pilot might have to change passage plans. These must be confirmed by the bridge team.

In challenging environments such as dense archipelago the pilot usually takes over but in remote piloting bridge team needs to handle situations with only instruction from the remote pilot. This means data that remote pilot receives must be real-time and trusted. Remote pilot receives data from different sensors on the fairway. Currently radars are utilised by VTS and with multiple radars there is some data fusion to make them more accurate. Lidars could be used in narrow areas and placed in bridges or other structures on the fairway. These provide more accurate data about distance to shore and a 3D image of the vessel. Cameras are used for checking the accurate orientation of the vessel and weather conditions. All these sensors send large amounts of data to the pilot and bridge team and integrity of it must be guaranteed since vessel sensors and fairway sensors are the primary source of information.

Since most procedures are done via information systems it is possible to record all verbal communication, ECDIS-view markings, vessel control commands, sensor data etc. This data must be stored in a secure ledger where no changes of data is possible so that whole remote pilotage event has kept its immutability and integrity. Data from remote pilotage can be later used in quality control or in investigations if an accident has happened.

3.2 S4V system requirements

For this thesis it is not necessary to focus on all various connections as shown in figure 5. To simplify needed communication channels we aim to research security of connections between remote piloting center, vessel and fairway intelligence. Fairway intelligence will then cover all different data sources such as lidars, cameras, smart buoys, TMF, open source intelligence etc.

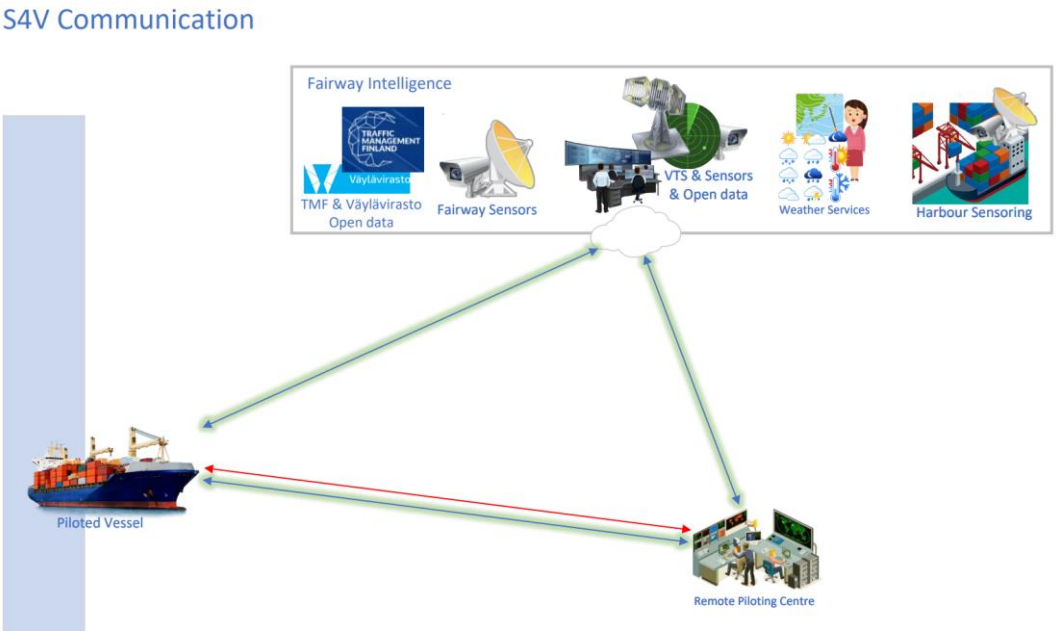

Figure 6. Simplified communication architecture (modified from [10])

This communication architecture simplifies the requirements that are needed for this thesis. Most important is the security of communication between remote piloting centre and piloted vessel. Additionally fairway intelligence is needed so that accurate information about weather conditions and position of the vessel can be aqcuired and trusted. Many mandatory requirements for the system are VTS radar location, FMI weather data, cameras for blind spots, ECDIS route, sensors collecting data of temperature, humidity, air pressure, wind and direction.

Initial simple list of system requirements that was thought were reliability, security, communication performance, maintenance and cost. Reliability and security of communications is one of the key requirements for remote pilotage to work. Other requirements such as low cost, amount of needed resources and maintenance were

considered as different blockchain technologies were searched but other varying options are also presented.

Authentication mechanism between a new vessel that arrives to the fairway and an authentication mechanism of new sensors or fog nodes must be ensured. Smart contracts and fog nodes provide a way for the master node to add more authenticated sensors to a fog node and it bases on Proof-of-Authority consensus so that the sensor must be added by a trusted party. This way only trusted sensors are authenticated to the system and malicious parties cannot tamper with them. Communication security between the fog node and sensor is ensured with standard encryption. However there still exists the question of how to authenticate completely new vessels that arrive to the fairway the first time ever. This can be done via blockchain smart contract if the arriving vessel has earlier informed about their arrival and given then the private and public keys to establish connection to the blockchain. When the connection is established blockchain provides a secure communication channel between the vessel and remote pilot.

Other authentication mechanisms also exist to ensure that the assumed party is who they say they are and Proof-of-Authority is established outside the blockchain. Li et al. [11] propose a reputation based system for authenticating new nodes to an ad-hoc vehicular network. They research a possible three party system consisting reputation centre, access points and vehicles. This study is quite old and they rely on centralized system with public networks which is not applicable in our system. However they propose an interesting idea for a possible blockchain system's authentication mechanism where a commonly used reputation system would exist. All nodes of the blockchain network would know other nodes' ratings and affect them, when some threshold value would be exceeded the system could alert about their trustworthiness when the masted node of the blockchain can remove this party from the network.

In addition to authentication of new devices scalability is important for the project's future prospects. With highly scalable system this fairway can be expanded over time and new parties added to the architecture to improve its performance and make the system more modular. Future in mind also maintenance issues need to be tackled. It is important to create a stable and scalable system that uses state-of-the-art technologies that are guaranteed to have maintenance and development in the future.

This can be done by relying on a blockchain technology and building an own solution which can be easily maintained by in-house developers, or by trusting a reputable client software of a blockchain technology that has a proper development team behind working to improve it in the long term. These clients need to have various features or pluggable services for scalability reasons.

Performance and reliability of communications is an important aspect since a small loss of connectivity could have serious results. Bandwidth, communication speed, the amount of data moving through the channels, the type of data etc need to be considered when thinking of the performance of the blockchain client. As maritime environment is hectic and information moves fast it is important that the communication channels can keep up. Reliability of these connections is essential especially between the remote pilot and the vessel. Other connections have some fault tolerance depending on their importance for navigation systems. Some parts of the system are more essential for the vessel such as the ECDIS-view and cameras but for example if one of the air pressure sensors lose connectivity it might not be that essential. Overall reliability of data and communication is more important so that no critical data is lost if some connection is lost. Blockchain provides more secure ledger than VPN when connection issues arise.

4 Blockchain technologies

This chapter gives an in depth technical review of the base blockchain Ethereum and the three blockchain protocols that were chosen. Also the chosen operating system for these protocols is introduced and its functionalities explained.

4.1 Ethereum

Ethereum is one of the biggest blockchain technologies in 2021 that has a broad use case market in different industies such as capital markets, decentralized finance, digital identity, government sector, insurance, healthcare and supply chain management. Because of the large variety of blockchains, choosing the one that suites the purpose best is a hard task. Bitcoin is probably the most known cryptocurrency blockchain but it is focused on changing financial banking systems. Ethereum instead has focused on decentralization of heavy computer systems and changing the current server-client infrastructure. The Enterprise Ethereum Alliance is the world's largest business blockchain consortium with over 450 members including Microsoft, JP Morgan, Accenture, ING, Intel, Cisco, and many more [12]. Ethereum is widely used through all types of industries from finance to supply chain management and digital identity. For example Ethereum has brought many possibilities for supply chain industries in terms of traceability, transparency and tradeability. For identity management Ethereum brings the advantage of decentralised identity management and embedded encryption. Service providers can verify the identity of a user which can then be granted access by giving them a private key to identify later again.

Since there are many different options of blockchain technologies to be used in maritime environment, this thesis focuses on comparing them and analyzing which one could be the best possible solution. There are many ways of implementing a blockchain technology such as creating one's own network solution from the scratch and writing own smart contracts. This is however a lot of work and many easier solutions exist such as software clients of different blockchains. Usually they come with already existing consensus models and security methods or pluggable extensions for these actions. Quorum for Ethereum is one of these solutions. Also a modifiable client Firefly by

Kaleido exists that allows user to decide which blockchain technology is implemented. The “big three” between Ethereum Alliance, Quorum, Corda and Hyperledger Fabric can be chosen. More about Firefly in chapter 4.4.

Options for which blockchain technology to use varies a lot whether private, permissioned or public blockchain is needed. Private networks restrict adding new nodes to the network which limits scalability and flexibility for the environment architecture. Public networks instead are open for everyone which excludes the option of Proof-of-Authority consensus model and creates a vulnerable system for sensitive network communications. For a smart fairway environment a permissioned blockchain is the best solution for scalability reasons and considering future work. J. Zhang from Kaleido provides a comparison in [13] between Ethereum, Hyperledger Fabric and R3’s Corda which are the leading permissioned blockchain technologies in 2019. They argue in the first chapter that Corda “is custom designed for financial industry” [13] which might be old information since Corda’s page describe their solution applicable to also energy, trade finance, telecommunications and supply chain markets. However all main points and differences between these three technologies are gathered in an extensive table comparing their aspects. They have many similarities in cryptography, identity management, maintaining team in numbers, privacy of transactions etc. The main differences between the three blockchains that can be highlighted in this chapter, before explaining each in depth, are languages that can be used writing smart contracts, smart contract engine and smart contract’s lifecycle. Ethereum’s own smart contract language Solidity, contract engine EVM (Ethereum’s Virtual Machine) and immutable lifecycle make the blockchain easy to configure for developers. However other clients such as Corda and Hyperledger Fabric use more common languages such as Java. These have accepted the base functionalities from Ethereum however creating their own distributed ledger technology solutions. The base blockchain Ethereum brings its best qualities to these permissioned or private clients to protect users from malicious parties trying to gain advantage from public blockchains. But as J. Zhang concludes in [13] “Choosing one over the other is likely not the best strategy at the moment” meaning that at this currently evolving situation all options are best to be kept open instead of just choosing one blockchain client or a blockchain to use. This is why this thesis focuses on the

comparison aspect for the future S4V work that could be in implementation during summer 2022.

4.2 Quorum

Consensys Quorum is a blockchain platform for enterprise solutions. Many large enterprises in fields such as finance, trading and energy have trusted Quorum with their blockchain needs and namely organisations such as Microsoft and J.P. Morgan. Quorum was acquired by J.P. Morgan in August 2020 with Consensys which means continuous development and management of Quorum services with on-demand support. Quorum is an open-source protocol layer that enterprises can develop in-house applications on top of it. It bases on Ethereum and comes with integratable product modules from Consensys [14]. Open-source version of Quorum includes a private key manager EthSigner, a private transaction manager Tessera and two available client softwares Hyperledger Besu or GoQuorum. The enterprise stack of Quorum is more complex solution that comes with additional modules such as Codefi Assets, Codefi Markets and Codefi Payments which are not yet necessary for the fairway project environment. With Quorum either permissioned or private network can be created which guarantees data integrity, confidentiality and availability with fast transactions and operational transparency. Quorum provides extensive guide on implementation and additional documentation of Hyperledger Besu, Tessera and EthSigner. On this thesis only Hyperledger Besu is covered but more information of GoQuorum can be found from [14]. All three can be downloaded as a bundle when downloading Quorum. Prerequisites are Node.js, Docker and docker-compose and a Linux machine [14].

4.2.1 Hyperledger Besu

Hyperledger Besu is a software client written in Java for Ethereum networks. It implements Proof-of-Work and Proof-of-Authority consensus mechanisms. Hyperledger Besu acts as the logging, monitoring, running and maintaining client and smart contracts and decentralized apps can be created and maintained with it. It can be run just as a command line interface or with JSON-RPC API.

Architecture of Hyperledger Besu is divided in three parts: storage, Ethereum core and network. The storage consists of the actual blockchain and then the history of states and the current overall state that consists of code and account states. Ethereum core handles the transaction functionalities by validating blocks/transactions and applying consensus. Network handles the basic peer-to-peer communications. Different Quorum plugins can be added to this architecture such as better monitoring and encrypted storages to improve security.

Different consensus models can be applied to Hyperledger Besu client. There exists one Proof-of-Work consensus model Ethash and four options for Proof-of-Authority consensus: Clique, IBFT 2.0, Quorum IBFT 1.0 and QBFT. For private and permissioned networks PoA consensus relies on trust between participants. They also have much faster throughput than in PoW consensus based networks. Differences between PoA consensus models are also wide. Finality is an issue when many nodes try to do transactions at the same time. IBFT 2.0 and QBFT have immidiate finality which means every transaction block is appended to the blockchain it is valid. Clique however has a voting system for transactions and their validity which can fork the blockchain momentarily. This creates reorganisation in the blockchain occasionally. Validators of a transaction can vary. In Clique only a single validator can be chosen but IBFT and QBFT need at least four validators for a transaction to go through and be appended to the blockchain. It is persumed that all validators will fulfill their responsibility and validate a transaction but in case some do not respond Clique is more fault-tolerant because only half of the validators need to sign the transaction.

To secure privacy between nodes Hyperledger Besu has a private transaction manager Tessera. Each node that wants to communicate transactions privately has their associated Tessera node. On a side note private transactions are still an early feature on Clique or PoW consensus models. Each Tessera node and Hyperledger Besu node have their own public and private keys and they use these to authenticate transactions that need to be private. All communications use TLS. Private groups additionally have their nonce value. Privacy groups are identified by their ID and handled by Tessera (more in chapter 4.1.3). These groups can be flexible if their memberships are handled inside smart contract. Hyperledger Besu however collects the state of all privacy groups in addition to the state of the blockchain. All transactions going through Tessera are

delivered privately directly to the recipients instead of sending them to transaction pool. However a privacy marker transaction is sent to the pool and later verified and added to the blockchain if nonce value stays correct, if not then the transaction will not be executed.

Since permissioned network is the most flexible but still secure solution Hyperledger Besu offers permissioning of nodes and accounts. Node permissioning means that only trusted nodes can access the network but account permissioning can restrict account's access to some nodes and it can be used to enforce identity requirements. Permissioning can be defined either locally or on-chain. Local permissioning happens inside one node and it can control who has the access. This is especially important in case of a threat or an attack because node owner can immediately protect it. On-chain permissioning affects the whole network and can take a while to coordinate the update through every node.

Transactions create logs that should be monitored but smart contract storage costs too much for extensive logs. Pluggable features can be added to Hyperledger Besu to improve monitoring and logging of network and node events. Plugins include commercial and enterprise versions such as Prometheus, Elastic stack, Quorum Hibernate and Splunk. All data can be visualized then in Grafana, Kibana or in Splunk's own graphical interface. There are two ways of changing logging: basic log level change and advanced format and output change. Log rotation can be changed to comply with regulations and standards.

4.2.2 Tessera

Tessera manages blockchain's privacy for GoQuorum and Hyperledger Besu with transaction manager and enclave. Transaction manager handles distribution of all private transactions, creates a peer-to-peer network and transfers transactions to and from ledger. More about private transaction lifecycle can be read from [15]. All private and public key access and handling happens in enclave. Eclave manages keys and encryption and decryption processes that are needed in transaction manager. Separation of these two acts as containerisation of sensitive data for security purposes that no leaks to unauthorised parties can happen. Logical separation always exists between enclave and transaction manager but they can be stored locally running in the same process.

Enclave can also be stored remotely in a HTTP process using RESTful endpoints and even in an additional environment for security purposes.

Tessera's enclave handles the main security parts of Quorum's blockchain. It can fetch the identities, which is a node's public key, and distribute identities of forwarding nodes and overall handling all identity management. It also creates encryption of transactions and other payloads and decryption of these. All these actions can be monitered with Prometheus or InfluxDB and visualized in Grafana. This makes key management and access control easier for administrators.

Privacy groups are managed by Tessera and there exists three types of groups: legacy, pantheon and resident [15]. Legacy private group is created always when someone sends private data to another node. The group id for lecagy group is created by hashing participants' keys. Pantheon is created before hand of transactions and the group id is hashed similarly as legacy's but an additional seed is added to the group id. Resident group is very different since it is managed by Tessera configuration file and the id is just the name of the group.

4.2.3 EthSigner

In comparison to Tessera that has transaction manager and enclave that handle the functionalities of network and key distribution, EthSigner signs transactions with private key and all private keys can be stored either in a cloud or encrypted locally. It is a proxy server that generates signatures for transactions using a private key and forwarding them. All keys can be stored in a local V3 keystore, HashiCorp cloud or Azure cloud. All connections between clients and cloud are secured with TLS communication. Similarly to Tessera all connections and events can be monitored with Prometheus and visualized in Grafana.

4.3 Hyperledger Fabric

Hyperledger Fabric is a distributed ledger software established by Linux Foundation in 2015 that helps enterprises develope modular applications and solutions. It is part of the same Linux foundation's Hyperledger family of solutions as Hyperledger Besu that

provide tools for easier blockchain deployment. Main purpose of Hyperledger Fabric is to make blockchain technologies for business solutions that need pluggable services with resilience and confidentiality. It is a permissioned blockchain that has identities for all participating nodes and policies that define transaction logic. Similarly to Hyperledger Besu this client uses Proof-of-Authority consensus mechanism so that only trusted parties can access the permissioned network and its tools. As for smart contracts that are created with Solidity programming language in Ethereum contracts, Hyperledger Fabric has a bigger overall smart contract called “chaincode” which handle business logic that is agreed upon members of the network. Chaincodes are like smart contracts that can be written in general purpose programming languages such as Node.js, Golang or Java and can be deployed inside isolated containers such as Docker.

4.3.1 Chaincode

As earlier presented smart contracts make the transaction logic of a blockchain network. In Hyperledger Fabric the term for this network wide smart contract is chaincode. The chaincode can consist of many smaller smart contracts with their own additional logic. In general it is however assumed that each chaincode has only one smart contract unless they are closely related. Chaincode or smart contract has access to the blockchain’s history of records (transactions and blocks) and the current state with a history of previous states. Basic “get”, “put” and “delete” methods are included in chaincode but Hyperledger Fabric’s chaincode comes with many APIs that can be used.

Every chaincode has to have an endorsement policy that defines that only for a transaction to be valid, previously defined organizations have to approve it. This policy obligates every smart contract inside the chaincode to follow it also. This usually means that inside every smart contract there are a few organizations that approve transactions from their peers. These cannot access the policies outside their own smart contract nor validate other transactions than what happen inside their own smart contract. Hyperledger Fabric differs from Ethereum with these policies because only transactions from trusted parties can be validated and/or executed. On the contrary Ethereum gives every node the same opportunity to generate transactions. Other policies can also be defined for stricting nodes from performing certain functions such as querying the ledger or executing transactions.

Consensus in Hyperledger Fabric provides validation of correctness of the whole transaction flow from proposal to appending to the ledger. As it works in Proof-of-Authority model a trusted Membership Service Provider (MSP) guarantees that a node that is part of the network can be trusted as MSP handles their identities using PKI. Also other consensus models can be plugged into Fabric such as Byzantine Fault Tolerance (BFT) consensus or crash fault-tolerant (CFT) consensus model. As earlier mentioned Hyperledger Fabric is designed to have modular architecture which means many pluggable consensus models, identity management protocols (LDAP, OpenID Connect) and key management services can be added. However in conclusion chaincodes and endorsement policies provide trust between peers and structure for transaction validation which means the consept of consensus is thereby achieved by ongoing verifications of transactions.

4.3.2 Transactions

Multiversion concurrency control [13] is used when same parameters are being changed by different parties. Smart contracts are upgradeable when new devices are added to restrict control and access if needed. It is possible that many concurrent smart contracts are deployed in Hyperledger Fabric networks. Most Ethereum based smart contracts work in order-execute architecture meaning they need to be deterministic. This might limit scalability and resiliency to which Hyperledger Fabric has implemented a new order-execute-validate architecture. It provides a solution for issues in flexibility, scalability and performance by executing orders before reaching an agreement of their proper order [16]. This architecture allows transactions to be executed with an endorsement of only a subset of peer nodes which can be separately defined in the chaincode. Parallel execution enhances scalability and flexibility whilst erasing the problem of determinism by filtering any inconsistent results. Before admitting a transaction to the ledger many versioning checks are made to ensure validity of the transaction and to prevent double spend operations. This is a part of transaction's lifecycle starting from chaincode when certain policies of endorsement are created. The latest research shows that Hyperledger Fabric is currently able to do 20 000 transactions per second [16].

Transaction flow has a preset of assumptions where the user initiating it has already identified to the trusted certification authority (CA) using PKIs and chaincode with endoresement policy is applied. Identity management happens in Membership Service Provider that grants identities for trusted users. After this the user initiates a transaction proposal by utilizing Hyperledger Fabric's one of the provided APIs and SDKs and a unique signature for the transaction is created. Nodes that are part of the transactions endorsement subset verify transaction's signature, its formality and that it has not been submitted before (replay attack). Participating nodes in the endorsement subset sign the transaction with their private key so that it can later be verified that all needed parties have validated the transaction. Also user's identity must be confirmed and whether or not they have access to the channel. After endorsement of the peers the chaincode is executed with transactions details to create results such as read/write set but no actual changes are made yet to the blockchain. Versioning checks and signature checks are then made that no collisions, invalid data or replay attacks can happen. If the transaction is commited to the ledger it will then be broadcasted to all the nodes in the channel as a block. All transactions are validated to follow endorsement policy and if not they are simply tagged as invalid. These invalid transactions are added to the blockchain but they do not change the state of the blockchain. Finally every channel's blocks are appended to the blockchain and a notification is sent to every participant.

4.3.3 Privacy

Privacy and confidentiality is an issue in permissionless networks but since Hyperledger Fabric is a permissioned blockchain client the architecture provides channels and "private data" for the participating nodes that need privacy. Nodes that belong to a channel can see each other but they cannot see other channels or nodes connected to those channels without additional calling of other smart contracts in the same channel or other. These channels are made by two or more organizations that agree on the same endorsement policy within one smart contract and nodes can be participants of one or more channels. These help parties in the blockchain to have private channels with many other other parties/nodes but also coordinate together when needed. As usually only one smart contract is deployed in a chaincode, channels provide a separate channel for private communication if no separate endorsement policy is needed.

Private data storage can be created within the channel to keep transactions logically separated from channel and private so only a subset of nodes in the channel can access this data. This data storage consists of the private data that is stored in state database that is private between only the participating parties and a hash of the private data is stored in all the nodes in the channel. The purpose of storing the hash in all nodes in the channel is purely for audit purposes and private data can later be shared with other channel members. One node can be a part of different private storages similarly as being a part of different channels. In addition data can be cryptographically encrypted with common algorithms such as AES. All private data storage transactions and sharings with other that storage members happens via PKI. This creates a level of privacy that can be leveraged now and in the future if private pilotage is needed and when for example vessel sensor data needs to be private from other participating nodes. Especially in case where many vessels communicate with the remote pilotage centre at the same time some conversations between the pilot and vessel can be kept private and confidential. If transactions must be kept confidential to only some peers then private data storages are a good option because then the data is only stored peer-to-peer but if all transactions must be kept confidential from other organizations then separate channels are recommended.

4.4 Corda

Corda is one of the newest blockchain technologies for businesses since it was founded in 2016 by R3 software firm. Corda networks have been developed across business areas such as banking, trade finance, telecommunications, supply chain and insurance [17]. Corda networks base on private transactions between participating parties and are flexible and agile for multi-platform systems. It is open-source and has a community of developers constantly making enhancements. The fact that Corda is such a new DLT means many possible changes in their technology and networks. For example their test network was decomissioned on April 2021 and they offered other solutions after. This chapter gives an introduction to Corda functionalities and how they differ from Hyperledger Fabric and Quorum but more can found from [17]. Corda is a semi-private network (not the same as permissioned) which means that participating node needs to

have real-world legal identity provided by ISP and a public key. Node identity for the network is only provided by know-your-customer processes by the ISP.

Network solution in Corda differs from other blockchain technologies from the fact that usually network broadcasts the transaction to all nodes of the network but Corda uses point-to-point transactions so that only the nodes that need the information will get it. This means that the ledger is different from each node's point of view and no node will have information of the whole ledger. Two peers that communicate share the same version of a ledger whilst communication and these two will save the conversation data. Since there is no central figure that handles all data all nodes are responsible for their own data. It is however possible to broadcast information to all nodes by looping the network service. Data is delivered in "states" that contain all transaction information. Two kinds of states exist: generic states and reference states [17]. Instead of updating the ledger data, reference states are added to reference state list that need to refer to a contract which makes them behave a bit differently during transaction lifecycle. Corda has named their node spesific ledgers as "vaults" that store current and historic states.

Unspent transaction output model is implemented which means every transaction updates the ledger but historic versions are immutable. Transactions consists of an input, which is the earlier parameter, and an output which is the proposed new version of the parameter. Input state includes the hash of the previous input and an index. Additionally transactions contain a time-window, commands, a notary service and attachments. Commands give more in depth description on intent of the transaction for example with the list of needed signers. Notary pool gives transaction lifecycle its finality for when the assigned notary entity is not present, no transactions are validated. Similarly to Quorum and Hyperledger Fabric all transactions need to be signed by required parties before commiting them to the blockchain. The validity is checked in the process so that it is contractually accurate, signed and no same transaction has been made before. Reference states need also a notary to confirm them.

Process of updating transaction to the ledger and other common tasks can be automated via flows. All transactions must contain data about what information is sent and to what parties within a time-window. Corda automates these by guiding the steps a node needs to do to achieve the ledger update it wants [17]. This helps to remove

networking and concurrency issues from node user. After flows have been determined to a node users can always start them with RPC calls. Flows handle all communication between nodes and each node has more than one flow that responds to incoming flows. This means that whilst execution of transactions there exists many concurrent flows. Flows are serialized to achieve this and if a flow waits for validation then some other flow will be executed in the mean time.

Even if transaction is signed by all needed participants it needs to abide by a contract. Transaction state defines what contract it is under and each of the transaction's input and output must be contractually valid. Contracts are deterministic to reach consensus in the blockchain network. Corda's sandbox of contracts restrict the use of some libraries that might lead to non-determinism. In addition two types of consensus must be achieved for a transaction to go through. Validity consensus requires that needed parties have verified contractual validity and the validity of a transaction and its preceeding transactions. Uniqueness consensus is checked by notary service which verifies that transaction's input has not been used by another transaction [17]. Notary service has pluggable consensus models which means structure and size of a notary service can change by using different consensus. This brings benefits to load balancing and latency lowering. Also privacy is enhanced because nodes can choose what consensus model notary service they want to use.

4.5 Firefly

Previously introduced Ethereum, Quorum, Hyperledger Fabric and Corda are blockchain protocols which can be used to create blockchain network for business solutions. Firefly instead is an operating system that provides an opportunity to use any of the previously mentioned protocols behind it, however Corda's implementation capabilities are still at research phase. It provides an easier and more flexible solution for the complex architecture layers from low level blockchain configuration to business level management. This gives developers a faster solution to create blockchain applications for enterprises. Their documentation [18] emphasizes how easy developing solutions for business their operating system is in terms of pluggability of services, visibility and control of data flow and custom transaction flow with easy to operate

functionalities and transaction submissions. Especially the complexity of protocols in application and private data layer need usually lots of familiarization before implementation. Since multiple use cases across industries can take advantage of blockchain technologies Firefly has focused on providing developer friendly API based approach with UI/UX to mobile friendly applications and auditing operations. All Firefly functionality is handled inside YAML configuration file where the chosen blockchain client is defined. This helps business innovation and motivates developers to create system of trust faster because there is no need for multiple separate configuration files.

4.5.1 Node

One node is a bundle of micro service runtimes with a single HTTP/Websocket API [18]. The architecture can be divided into Firefly core, connectors and infrastructure runtimes. The core is the API and the event server, connectors consist of blockchain interface and public storage interface and infrastructure runtimes consist of blockchain nodes, data storage and private data communication channels. Core handles all lifecycle orchestration, broadcast manager and private storage. It is the host where applications connect with API and UI. Private messaging can be established via Kafka, RabbitMQ or ActiveMQ. All connector runtimes are pluggable, since all protocols are pluggable, such as all Firefly connectors can be written in Java, Python, Go or Node.js etc and consist of file transfer and end-to-end encryption. They can also use any network transportation protocol such as UDP instead of only HTTPS/WebSockets. Infrastructure runtimes handle multi-party activities and have many options blockchain services such as Hyperledger Fabric, Ethereum and Corda, cloud and containerization services such as Kubernetes PCV, Azure and AWS S3, and storage services such as PostreSQL, SQLite etc. Firefly offers pluggable UI for blockchains that can be chosen to meet the requirements of one's system and event driven programming. This makes developers' job easier because they can use REST API to access blockchain functionalities and still do not have to configure on-chain logic that in depth. Firefly offers a possibility for combining custom on-chain logic with smart contracts and a solution that does not require creating custom smart contract logic.

Plugin architecture for Firefly makes the multi-party system more extensible and scalable. Connectors provide separation for features that might have differences in programming languages, transport methods and high availability architectures. Databases are pluggable in Firefly core, connector plugins must be 100% FLOSS code not GPL and lightweight in mapping. Infrastructure runtime is in its essence pluggable since different blockchains and cloud environments can be chosen to be used.

4.5.2 Privacy

In a multi-party system some organizations want to have private data exchange that is not broadcasted to all nodes in the network. Privacy of data can be preserved with either zero knowledge proofs or trusted compute environments. Inputs and outputs can be cryptographically secured and shared with these methods without the other participant's knowledge of the internal data. Private message channels are pluggable features that use TLS for all communication, authentication mechanisms such as Java Web Tokens or private key infrastructure, synchronous and asynchronous transports such as HTTPS or Kafka to help communications online or offline, peer-to-peer communications or hub, end-to-end encryption and compression of large data transfers. These messages that go through private channels, such as in Hyperledger Fabric, are hashed and only the hash pin is added to the blockchain. Messages contain at least sender and receiver information and additional information. Receiver can be a group of participants whether many parties attend the same private channel. Fundamentally after a user sends some data via blockchain as a BLOB some other user can request the full data to be sent via private channel. After the initial user has authorized the request and retrieved data by its hash, full data is sent and no blockchain is a part of this data transaction. This way the record of the request is stored in the blockchain.

To establish a secure ledger of transactions in global order some data, such as reference data, needs to be broadcasted to all participating nodes. This shared data is stored off-chain but its hash is stored on-chain for future deletion purposes. Off-chain data needs to be stored securely that it does not leak outside the network. Broadcasted information usually is organizational identities, nodes, data types and namespace data. Broadcasts usually contain large amounts of different types of information which means for efficiency reasons these messages are batched.

4.5.3 Multi-party event flow

Determinism of a consensus algorithm depends on the complexity of requirements in transactions. Technological maturity, data privacy, auditing purposes and cost of implementation in means of developer skills and maintenance create levels of importance in consensus algorithms. Since Firefly offers different protocols for different use cases these factors can differ a lot. Blockchains offer solutions for determinism in form of a predetermined sequence of events whether it is bi-lateral or multi-lateral and identification of a message based on its hash value.

Parties can communicate with each other on-chain or off-chain. Each party has their own private history that consists of off-chain message data as well as blockchain data. To achieve consistency and agreement between multiple parties same datatypes need to be agreed on. Firefly's importance is shown in handling the local events from processing, confirming and sequencing them whilst handling other events happening at the same time. It aggregates data into multiple runtimes and deliveres it to multiple locations using REST APIs. These transaction flows need to be deterministic no matter if they are local messages, local applications using blockchain or globally to all applications when blockchain pins events [18]. Firefly instance stores transactions and every instance sees them in the same sequence as others. Concurrency issues are dealt with message queues that assemble private and public transactions that arrive to the blockchain in different orders. This job is handled by inbound aggregator that manages triggering events in sequential order. If some transaction does not arrive before some other transaction that needs it before, both transactions are discarded. Processes can be created with event triggers, inputs and outputs with common step sequence rules. These steps can be automated with common tools such as REST APIs, WebSockets and WebHooks which makes history data querying fast. Partial history is stored in the blockchain so that private metadata and sensitive information does not leak into blockchain or outside the network.

5 Results

Many variants of a blockchain technology exist with similar capabilities, for example Hyperledger and Ethereum, because the main blockchains have evolved clients that assimilate some features of the bigger underlying blockchain technology. Quorum, Hyperledger Fabric and Corda are one of the biggest open-source blockchain protocols of today. These provide possibilities for maritime cyber security environment by providing no single point-of-failure network and a tamper proof ledger. In this chapter the research questions will be answered, the three protocols are evaluated against the requirements that S4V project has, protocols are compared with each other and a suggestion of a possible solution is given. Finally it is described how this thesis can be used in the future and what future work will be done.

5.1 Research questions

The introduction chapter presented the research questions that this thesis aimed to research answers to. First one was what benefits can the researched blockchain technologies bring to the Sea for Value project? Second question was that what blockchain technology is the best for authenticating messages and other data transactions between remote pilot, bridge and other fairway intelligence such as fairway sensors? Finally it was thought that can blockchain create more secure communication methods compared to VPN and if yes, how? These questions evolved during the time thesis was written and especially the second question about authentication mechanism evolved from message authentication to more about authenticating new vessels and other nodes to the fairway system.

Blockchain can benefir S4V project by creating a secure, immutable, tamper proof ledger that stores all wanted data. Not all fairway intelligence data is reasonable to store on the ledger but especially communication between remote operations centre and vessel is important to be stored. This way if some node drops from the network or communication channel drops down for a moment, other participants will not lose connections, data recovery is easy and the fault is easy to track down. It is especially important in case of an incident that all remote pilot instructions are recorded.

Permissioned blockchain network keeps data private from outsiders and new nodes can be later added to the network with Proof-of-Authority consensus model.

For the question what blockchain can be used there exists different possibilities for future scenarios. Ethereum blockchain technology was chosen as the main base blockchain because it has focused on decentralized applications for different industries compared to most large blockchain technologies such as Bitcoin that are focused on financial industry and cryptocurrencies. Since blockchains are still very new technologies and constantly developing, it depends on the project's later implementation during summer 2022 which Ethereum based blockchain protocol client will be deployed but this thesis proposes three protocol options Hyperledger Fabric, Corda and Quorum and an operating system Firefly for multi-party systems. This research question will be answered more in depth in chapter 5.3 where proposed solution is explained and possible future uses for this thesis are thought.

Second research question of what blockchain to use for authentication of messages and transactions between remote pilot, vessel and sensors evolved during the writing because blockchain itself cannot properly handle authentication mechanisms in S4V environment. Proof-of-Authority consensus model is the only proper one for maritime fairway environment since others are computationally too heavy and parties are assumed to know each other when they join the fairway. This means authentication has to be done before joining the blockchain network by other means such as PKI and then added to the list of trusted authorities. After joining the network all communication is encrypted and delivered over secure channels. All transactions and messages are verified, signed and validated by predetermined parties of the network. All three blockchain protocols have different mechanisms for this but more about proposed blockchain solution is described in chapter 5.3.

The final research question about if blockchain can create more secure communication mechanism compared to VPN was one of the initial research questions that later on turned out to be a bit out of scope. VPN and blockchain are very different technologies and probably both are later used on S4V project. Blockchain does create a more secure communication mechanism because there exists no single point of failure, because the ledger is secure in decentralized fashion, and private data channels and transaction logic can be created within one system. If one communication channel is lost

temporarily all other communication persist, in comparison to VPN where if connection is lost to the server or the server is down then all communication is down and logs might be lost too.

5.2 Protocol comparison

Initial requirements had an influence on the decision of the three chosen protocols. Permissioned network solution was chosen for future flexibility and scalability reasons. It was important that mining cryptocurrencies was not required, preferably them to be open-source or at least not costly and have on-going maintenance. Popularity, community, use cases and encryption made a difference also. The chosen three are all used widely across industries which makes them easily mutable and scalable for different use case scenarios.

Even though the base blockchain functionalities exist in all protocols each have brought their own additions to benefit different organizations and use cases. All protocols have some similarities in programming languages, privacy management, public key infrastructure and consensus models. All have a possibility to be deployed in a containerized environment for example in Docker with docker-compose and all support permissioned network solution. Also every protocol has their own monitoring system for example Kibana or Grafana etc that make log management easier to access and monitor. This makes deployement easier for developers and secures development process and deployement better. However Hyperledger Fabric and Quorum have more similarities overall than Corda since it is backed by R3 and has a bit different viewpoint how a ledger should work. Quorum's core ledger is Hyperledger Besu that belongs to the same Hyperledger family as Hyperledger Fabric. Quorum provides other applications additionally for privacy and private key management. Corda instead has a peer-to-peer blockchain network where there is not only one big blockchain backing everything up but nodes construct the whole ledger with different parts that they store. The table chart on the next page about requirements and how they are met in every protocol gives a simple preview of all the similarities and differences that these Hyperledger Fabric, Quorum and Corda have.

Requirements /features	Quorum	Hyperledger Fabric	Corda
Encryption of data	AES	AES etc	RSA, PGP, nodes responsible for their own encryption
TLS	Enabled client-server	Enabled, one-way or two-way authentication	All communication
Access control	Tessera	MSP	ISP
Languages	Java, Go	Java, Go, Node.js	Java, Kotlin
Pluggable features	Consensus models, logging and monitoring services	Consensus models	Consensus models in notary services
Transactions/s	100	20 000	15-1678
No single point failure	Yes	Yes	Yes
Documentation	Large, partially on Github, separated Besu, Tessera, EthSigner docs	Extensive, bit scattered	Made for non-tech persons, not so extensive, inconsistent
Maintenance	On-going	On-going	New technology, seeking contributors
Concurrency	Clique consensus provides some	Channels, smart contracts	Flow concurrency
Cost	Open-source / Enterprise	Open-source	Open-source / Enterprise

Table 1. Features of blockchain protocols

5.2.1 Main differences

This chapter explains the main differences between the three protocols and explains table 1 more in depth. Encryption and TLS mechanisms differ slightly but not so much

that it would make a big difference in S4V maritime environment. Access control however is managed very differently between Hyperledger Fabric, Quorum and Corda. Quorum has its own application Tessera that handles all transaction control and identity management by controlling private key usage. Hyperledger Fabric has a Membership Service Provider that acts as a trusted authority which creates and manages idententities of participating nodes after they have identified themselves to a certificate authority. In S4V this authority could be for example the remote operations centre. Corda has a whole different system because they have a semi-private network instead of permissioned which means a real world authority, which usually is an internet service provider, authenticates nodes and ties them to a real world known person or organization. All three protocols use PKI after authentication of new participants has been done.

Most popular programming languages are used in smart contracts but their consensus models are very extensive and pluggable with different protocols. Quorum has the possibility for PoW consensus but PoA is more reasonable and IBFT 2.0, QBFT and Clique are considered. Clique provides the most flexible solution which means the blockchain can fork temporarily and it accepts transaction with lesser validators which makes it flexible. This means blockchain reorganization is needed sometimes. This process makes emergency situations easier to handle in comparison to IBFT or QBFT that need at least four validators. However IBFT and QBFT have immediate finality but transaction validation might take longer time.

Hyperledger Fabric defines consensus as the sequence of required steps that certain parties have to do to achieve consensus. It has basic PoA model but other pluggable consensus method can be implemented into Fabric such as Byzantine Fault Tolerance (BFT) consensus or crash fault-tolerant (CFT) consensus model. Its chaincode and additional smart contracts define their own endorsement policies that require some subset of participants inside of the smart contract of the network to sign transactions before they are validated to form a block.

Corda has PoA consensus model but very different pluggable consensus methods. After transaction is signed by all needed participants it needs to abide by a contract. Contracts are deterministic to reach consensus in the blockchain network. Two types of consensus must be achieved for a transaction to go through. Validity consensus

requires and uniqueness consensus is checked by notary service which verifies that transaction's input has not been used by another transaction [17] and it is not double-spent. Notary services differ with every transaction because every transaction can choose which notary service to use in their peer-to-peer network where only certain participants have same transactions as them. All nodes have stored only their transaction history with other nodes and no central ledger exist which is the main difference between Corda and other solutions.

One of the biggest requirements was throughput or in other words how many transactions could be sent in one second. This parameter differs a lot between these three protocols and since throughput is especially important in fast paced maritime environment Hyperledger Fabric seems the best solution with 20 000 transactions per second. Second best option is Quorum and lastly Corda with differing throughputs from 15 to 1678 transactions per second. Considering all nodes and organizations that are sending data to the blockchain it is important to have high throughput especially if all data is wanted to be saved in the blockchain. For example some sensors might send new information every second and they might strain the blockchain's capabilities.

Extensive documentation makes the blockchain protocol easier to deploy and maintain. Comparing all three documentations it is easy to see which protocol clients are newer in the industry and what kind of community they have behind. Quorum's documentation was first very hard to read in June 2020 but when J.P. Morgan acquired them the whole documentation and support behind the client changed to more maintained and documentation became more clarified and simple to read. Quorum has extensive documentation of each application it provides. Hyperledger Besu, Tessera and EthSigner have their own documentation pages that explain their connectivity. Since Hyperledger Fabric is backed by the Linux foundation it has on-going maintenance and a big community of developers behind it. Hyperledger Fabric has only one extensive documentation that is a bit scattered between "Getting started" and "Architecture" but still very easy to navigate. Corda has taken a more modern solution for documentation and created a divided documentation to their web pages. It is scattered and inconsistent which makes the reader jump back and forward between pages a lot. It is obvious Corda is not designed for purely technical persons since documentation does not go very deeply into programming aspects and only explains the basic functions of Corda's

workings. Corda is still a very new blockchain in the market which means that more extensive documentation about configuring flows and contracts are possibly yet to come.

5.2.2 Concurrency

Concurrency and parallel processing is a big issue in a multi-party system where many participating nodes could try to update same parameter at the same time. Especially in blockchain networks many concurrent updates on the ledger are being done by different nodes and these need to be organized and analysed that all transactions are executed in the right order and no collisions happen.

Corda has different concurrency issues compared to Hyperledger Fabric or Quorum since the ledger is stored in multiple parts between nodes. Flows handle all transaction management between Corda nodes and if many concurrent flows happen at the same time they will be serialized. If some other flow waits a parameter update from another flow their execution will be delayed until the parameter is updated or their execution will be dismissed. In the mean time other awaiting flows will be executed. Processing of flows can be automated to make networking and concurrency management easier.

Hyperledger Fabric has more options in concurrency management. It supports multiple concurrent smart contracts and channels. Multiversion concurrency control is used when same parameters are being changed by different parties. Hyperledger Fabric has implemented a new order-execute-validate architecture that executes transactions before reaching an agreement of their order. Only a predefined subset of validators need to verify that the transaction is correct and fulfills the endorsement policy which increases flexibility of system. Parallel execution enhances scalability and flexibility whilst erasing the problem of determinism by filtering any inconsistent results.

Quorum provides concurrency control via consensus methods. IBFT and QBFT have more strict transaction management since at least four validators need to verify the transaction as valid before it is added to the blockchain. Clique consensus model instead has a voting system for transactions validity which makes the blockchain fork momentarily and reorganization must be done afterwards. Only one validator can be

chosen and if there exist more than one only half of validators need to verify the transaction. This way flexibility is improved and fault-tolerance increases.

In a fast paced environment such as a smart fairway collisions and concurrent transactions is evidently going to happen. This creates a risk if misinformation gets through or if the throughput is significantly slowed down. Only needed nodes should be added to the blockchain network so that it does not grow to be too computationally heavy. Hyperledger Fabric has the biggest transactions per second rate and the best multiversion concurrency control. It can handle parallel execution with many concurrent smart contracts and channels and throughput is increased with orded-execute-validate architecture which makes it the best option for our environment.

5.2.3 Privacy

Participants in the fairway environment have needs for private messaging and for transaction data to secured from parties that are not included in the network. Sometimes there is also a need for private communication inside the blockchain environment for example of remote operation centre's instructions for operated vessel.

Quorum offers private transaction logic via private message groups. Tessera is a private transaction manager which handles all privacy group management and private key control that are used to authenticate private messages. Each node that wants to communicate transactions privately has their associated Tessera node. A peer-to-peer network is created for private transactions and they are encrypted. Logical separation of private transaction signing and encryting works as containerisation of sensitive data so that no leaks to unauthorised parties can happen.

Hyperledger Fabric has many ways of providing privacy for participating organizations. Private channels can be created within a smart contract and contain two or more organizations that have agreed on an endorsement policy. Nodes that belong to a channel can see each other but they cannot see other channels or nodes connected to those channels without additional calling of other smart contracts in the same channel or other. Hyperledger Fabric provides additionally private data storages channel to keep transactions logically separated from channel and private so only a subset of nodes in the channel can access this data. Storage is stored peer-to-peer with nodes that need it and can be cryptographically secured by storing only a hash of it in the nodes.

Depending on how many participant the storage needs sometimes channel is better solution for private transactions.

Corda offers privacy only by notary services because every notary service can use any consensus model and nodes can choose which notary service to use. Privacy is automatically created as point-to-point network sends transaction to only the nodes that need them. However this shows that Hyperledger Fabric and Quorum have more advanced privacy methods and larger use cases for them. Logical separation privacy management and private transaction storage creates a more secure environment that protects sensitive data from leaking to unauthorised parties. Encryption of all private transactions and signing them with private keys helps incident response when the sender is easy to recognize. This is important in large systems with multiple different participants. Hyperledger Fabric gives many ways for participants to send and store transactions privately which makes it a preferable solution in S4V project. Quorum provides as good solutions as Hyperledger Fabric but might have a bit more complex environment since Tessera nodes have to be created and communication established with Hyperledger Besu and EthSigner.

5.3 Solution and future work

Previously compared protocols give two very good options for S4V project's fairway environment. Considering Corda's blockchain solution it is too recently developed technology with uncertain future and difficult ledger constructs for it to be a useful solution for S4V project. Corda is designed more for financial and banking industries from which the use of notary service becomes useful. Hyperledger Fabric and Quorum have different features from each other but bring quite equally good solutions to secure communication channels from outside malicious parties. Privacy methods are implemented in a professional matter with encryption of data, logical separation and private messaging channels. However when comparing concurrency and throughput issues that are especially important in a multi-party maritime environment it becomes clear that Hyperledger Fabric will perform better with high data amounts and with fast paced information flow. Eventhough only needed nodes are added to the blockchain the amount of data that moves between remote operations centre, vessel and other fairway

intelligence is huge and some sensors might have to be configured to send less data. Twenty thousand transactions per second is still a bit slow throughput but after proper configuration of environment Hyperledger Fabric can provide a secure way to establish connections.

Firefly is an operation system that helps developers to create blockchain applications without having to build complex smart contracts with computationally heavy business logic. It eases developers' work by not having to learn all different layer protocols because REST API can be used for accessing blockchain's functionalities. Firefly can deploy any of the previously mentioned blockchain technologies but Hyperledger Fabric has the biggest support in Firefly nodes. Corda is still in development phase as a protocol and as an implementation to Firefly. Quorum is another viable option for Firefly but as earlier researched Hyperledger Fabric would perform better in a smart fairway environment with remote pilotage operations. This combination of Firefly's tools for blockchain building and Hyperledger Fabric's protocol is the most scalable, high performance, flexible and secure solution for S4V project. It creates an easily configurable multi-party system with pluggable services, visibility and control of data flow and custom transaction flow with easy to operate functionalities. Developers use a lot of time to familiarize the complexity of blockchain protocols to which Firefly gives an easier solution. Plugin architecture makes the multi-party system more extensible and scalable and gives an opportunity to have communications on-chain and off-chain with private data storages and messaging. Hyperledger Fabric's private message channels are pluggable for on-chain or off-chain communication. Multiple different programming languages, data transportation methods and high availability architectures can be used simultaneously because connectors separate them from the main core. To achieve consistency and agreement between multiple parties same datatypes need to be agreed on. Firefly sequences transactions whilst handling other events happening at the same time. Concurrency issues and problems with parallel execution is there by efficiently solved.

Fog nodes are a great safe way to balance work load from sensors on the fairway. Not every camera, lidar or weather sensor need to have blockchain running in them which is not even possible in some scenarios. A middle node that runs the blockchain instance is placed to collect data from smart contract listed sensors and send

it to the blockchain. This middle node could be a piece of hardware such as a RaspberryPi. It creates a secure mechanism to manage sensors on the fairway and creates a mechanism for trust where all data coming from sensors has remained their integrity. Fog nodes guarantee integrity of data and prevent Man-in-the-Middle and replay attacks as each message in the blockchain are cryptographically hashed and nonce are added with a timestamp. Assurance of correct node data is also improved with fog node technology since all sensors are listed in the agreed upon smart contract which means no malicious party can add new nodes to the blockchain. Also misbehaving sensors can be easily removed from fog nodes without direct access to the sensor.

There are many use cases for both Firefly and Hyperledger Fabric as their own since Firefly can implement any of the three protocols presented in this thesis. Hyperledger Fabric has many successful case studies across industries varying from healthcare to finance and cyber security. Even certificate authorities have chosen Hyperledger Fabric as a solution for private key management [19]. Since Hyperledger Fabric is only the protocol behind many decentralized solutions it is more important to focus on the success stories that Firefly has accomplished with differing blockchain solutions and whether it has accomplished its main purposes.

Kaleido is the organization behind Firefly and has customers ranging from energy, transportation, finance and identity management industries. Namely organizations such as WWF, World Bank and Komgo have managed to create new assets and increased gains with Firefly blockchain solutions. Digital identities have been one of Kaleido's success showcases with Gorilla Hash digital signature system and Andes blockchain that manages access control to restricted places. Hashes of requirements for restricted area are stored in the blockchain and self-sovereign identity-based access control system will verify the identity of a user and whether or not they have access [20]. Handling digital identities with IDs and signatures is increasingly important in today's data driven world. These showcases can be used as examples and inspiration for future work in S4V fairway environment's identity management and authentication schemes.

This thesis focused on researching blockchain solutions for S4V remote pilotage project. Comprehensive comparison was done to evaluate the best open-source blockchain protocol of 2021 that meets the requirements of the environment. In the

future during summer 2022 implementation and deployement of Firefly and Hyperledger Fabric will be done. This solution should be tested with proper hardware and network architecture but it however requires re-evaluation of these blockchain technologies to make sure that they are still working the same way that they are during the writing of this thesis. Blockchain technologies evolve in a fast paced manner and some of the compared protocols might have gotten better or worse. However this thesis provides three viable blockchain protocol options for future work. This research can be used as guideline to what kind of features and requirements a complex remote pilotage system in a smart fairway has and how to meet them with blockchain technology. S4V communication architecture was simplified for this thesis which means as an obvious future work is to think how blockchain network is expanded to the whole environment and how to agree on one blockchain between all other participating organizations. Firefly is working on the prospect of a multi-party system where organizations can choose their own blockchain solution and these could be compatible with each other. Using Firefly as a starting point for other researches for whether a complete one-in-all solution for these kind of environments is possible could be done since for now possible solutions are very mix-and-match. Security and privacy of the three blockchain protocols and the operation system have been explained in depth which creates possibilities to expand this research into different kinds of environments with multiple participants that need secure communication channels and ledgers and a no point-of-failure network architecture.

6 Conclusion

Blockchain technologies can be used for many different purposes from handling large amounts of data to creating better solutions for privacy protection, user authentication and a tamper proof ledger which lead to growing interest among industries. Smart contracts, fog nodes and different consensus methods create a scalable environment to secure multi-party connections with equal trust of participanting nodes' identity. Different blockchains have multiple options for methodologies to use in different environments. This thesis has focused on Ethereum based open-source solutions that fit the remote pilotage environment the best.

Autonomous vehicular networks and remote operatable devices have been a popular research topic in the last few years. Remote pilotage in maritime environment is persumed to reach its full potential with fully autonomous vessels in ten years which makes the topic interesting for all researchers. However cybersecurity in these environments is especially important because incidents can lead to financial loss, reputational damage, loss of customer and industry trust and environmental damage. These complex environments also have multiple attack vectors because of the systems wireless nature. Denial-of-service (DoS), man-in-the-middle (MITM), message or executable code injection, authentication tampering and GPS spoofing are one of the most usual attacks against large IoT systems. This is why blockchain can be used for creating a tamper proof environment with no single point-of-failure.

After extensive research about best performing blockchain technologies Ethereum seemed the most preferable for decentralised maritime environment. In comparison to most of 2021 blockchain technologies that have focused on financial industries and cryptocurrencies, Ethereum has focused on decentralizing applications within many different industries. This thesis provides three Ethereum based blockchain protocol solutions and one operating system for these protocols. All have different features that add to the base blockchain technology but after extensive comparison two of these protocols perform better in means of concurrency and privacy. Hyperledger Fabric and Quorum provide many ways of tackling privacy, concurrency and parallel execution issues with consistent high throughput levels. However Hyperledger Fabric has far better throughput and concurrency management. This makes the solution of Firefly operating

system with Hyperledger Fabric blockchain protocol the most preferable solution in complex remote pilotage fairway environment.

As I was researching materials for this thesis I could not find any similar technical papers or researches where someone else would have similar research questions of how to create a secure remote operations network by using blockchain technologies. Initially this thesis was to be an implementation of a blockchain technology to S4V project but the scope was reduced to researching possible solutions for a remote pilotage system in a smart fairway environment. As blockchain is still such a new technology it is constantly developing which meant that research methods required many changes as this thesis was under work. For example the initial blockchain client that was researched got purchased by another large blockchain company that rewrote the software client from scratch. They also changed their private transaction manager to another one again in the middle of the thesis. This means that during summer 2022 when the project is being implemented a re-evaluation of these blockchains needs to be done for the purpose of validity of the proposed solution.

There currently exist many different blockchain technologies and their clients that are maintained and developed by different large companies. Because of this large amount of options the task of finding an existing solution for multi-party maritime environment is difficult since many blockchains are either public and transparent, or are meant for financial purposes and mining cryptocurrencies. In 2021 almost 8 out 10 the most popular blockchain technologies were purposed for mining cryptocurrency or making other financial transactions. But as earlier mentioned J. Zhang concludes today's blockchain technologies shortly as: "Choosing one over the other is likely not the best strategy at the moment". This is why this research can be used for defining requirements in remote pilotage system or smart fairway environment and as a base for future implementations of the three protocols.

Since this thesis was a year long research where blockchain technologies changed during the writing process it is almost impossible to say what the research outcomes would have been if the first preferable systems were tested with the requirement assumptions. The research would probably have been much shorter and might have not given any possible solutions and outcomes for future researches. As blockchains develop so fast it is important to keep up with the preferable solutions' development and case

studies since the implementation phase is yet to come. Research questions developed over the time whilst writing this thesis and the scope was re-defined since implementation and testing methods that initially were to be done got out of scope and this thesis would have been prolonged for too much time. Comparison research of the current most popular blockchain technologies instead of an applied research was a better solution for the project's deployement during summer 2022. This way the thesis can be used as guidance for future work in blockchain technologies.

References

[1] DIMECC. Sea for Value S4V. February 12. 2020. [Online] Available: https://www.dimecc.com/dimecc-services/s4v/

[2] D. Yaga, P. Mell, N. Roby and K. Scarfone. "Blockchain Technology Review". NISTIR8202. NIST National Institute of Starndards and Technology. [Online] Available: https://doi.org/10.6028/NIST.IR.8202

[3] R. Gupta. *Hands-on cybersecurity with blockchain*. Packt Publishing. Birmingham, Mumbai. 2018.

[4] M. Tahar Harri et al. "Bubbles of Trust: A decentralised blockchain-based authentication system for Iot". LTCI, Telecom Paristech, France. 2018.

[5] L. Lesavre, P. Varin, P. Mell, M. Davidson and J. Shook. "A Taxonomic Approach to Understanding Emerging Blockchain Identity Management Systems". NIST Cybersecurity Whitepaper. January 14, 2020. [Online] Available: https://doi.org/10.6028/NIST.CSWP.01142020

[6] R. Almadhoun et al. "A User Authentication Scheme of IoT Devices using Blockchain-enabled Fog Nodes". Khalifa University of Science and Technology, Abu Dhabi, UAE. 2018.

[7] M. Orcutt. "Once hailed as unhackable, blockchains are now getting hacked". MIT Technology Review. February 09, 2019 [Online] Available: https://www.technologyreview.com/2019/02/19/239592/once-hailed-as-unhackable-blockchains-are-now-getting-hacked/ Accessed 15.04.2021

[8] B. J. Vartdal, J. Blomhoff, Ø. Goksøyr and S. Adams. DNV. Webinar. "Digital transformation in the maritime industry. Smart, data driven and secure operations". March 18, 2021.

[9] J. Blomhoff, G. Smefjell, S. Einarsson and A. Vorreiter. DNV-GL. Webinar. "Maritime cybersecurity". March 25, 2020.

[10] J. Alamaunu. Brighthouse Intelligence. "S4V logical connectivity architecture". Set of Brighthouse Intelligence communication architecture plans. 2020.

[11] Q.Li. et al. "A Reputation-Based Announcement Scheme for VANETs". IEEE Transactions On Vehicular Technology, Vol. 61, No. 9, November 2012.

[12] ConsenSys. "5 Reasons Why Enterprise Ethereum Is so Much More Than a Distributed Ledger Technology". December 12. 2018. [Online] Available:

https://media.consensys.net/5-reasons-why-enterprise-ethereum-is-so-much-more-than-a-distributed-ledger-technology-c9a89db82cb5

[13] J. Zhang. "Enterprise blockchain protocols: A technical analysis of Ethereum vs Fabric vs Corda". kaleido.io. October 24, 2019. [Online] Available: https://www.kaleido.io/blockchain-blog/enterprise-blockchain-protocols-a-technical-analysis-of-ethereum-vs-fabric-vs-corda

[14] ConsenSys Quorum. *Build on Quorum, the complete open source blockchain platform for business*. [Online] Available: https://consensys.net/quorum/

[15] Tessera. *Private transaction manager Tessera*. [Online] Available: https://docs.tessera.consensys.net/en/stable/ Accessed 21.10.2021

[16] Hyperledger Fabric. *Docs Introduction*. [Online] Available: https://hyperledger-fabric.readthedocs.io/en/latest/whatis.html#hyperledger-fabric Accessed 20.10.2021

[17] R3. *Corda Key Consepts*. [Online] Available: https://docs.r3.com/en/get-started/corda-key-concepts.html Accessed 24.10.2021

[18] Hyperledger Firefly. *Hyperledger Firefly documentation.* [Online] Available: https://hyperledger.github.io/firefly/

[19] Hyperledger Foundation. "Thales and DigiCert team up to increase cybersecurity for Hyperledger Fabric". Case study. [Online] Available: https://www.hyperledger.org/learn/publications/thales-digicert-case-study

[20] Kaleido. *Andes blockchain*. [Online] Available: https://www.kaleido.io/solutions/andes-blockchain