BACHELOR'S THESIS

Comparative analysis of blockchain solutions for federated machine learning in healthcare

Publication Date: 2022-02-23

Author
Vinzenz ZINECKER
Karlsruhe Institute of Technology
Karlsruhe, Germany
vinzenz@zinecker.de
0x48c6A2814f03759A6b5aE36395c713c561f507fF

Abstract

As Machine Learning approaches get more sophisticated, many applications of machine learning in the healthcare sector are proposed (Mathur, 2019). However, medical data is often distributed across multiple institutions (i.e., hospitals, insurance providers etc.) and cannot be aggregated due to privacy concerns and ownership structures (Brisimi et al., 2018). Federated Learning (FL) is a technique that can be employed to enable distributed Machine Learning without sharing the underlying data (Kairouz et al., 2019; McMahan et al., 2016). There have been many approaches implementing federated learning in conjunction with blockchain technology. This avoids the single point of failure of centralised federated learning and promotes decentralisation. However, there are many competing methods and approaches on how to integrate the blockchain layer into a FL scheme. This thesis gives an overview on existing approaches, determines important requirements on the blockchain layer, then compares existing blockchain solutions and evaluates...

Keywords: blockchain, federated learning, hyperledger fabric
Methods: systematic literature review, experiments

Submission Date: 2022-02-22
Submission Contract: 0x2869a65e75B8eB3B7B96b39Bb84383b3c31BF082

Comparative analysis of blockchain solutions for federated machine learning in healthcare

Bachelor Thesis

by

Vinzenz Zinecker

Degree Course: Industrial Engineering and Management

Matriculation Number: 2067805

Institute for Applied Informatics and Formal Description Methods (AIFB)

KIT Department of Economics and Management

Advisor:	Prof. Dr. Ali Sunyaev
Second Advisor:	Prof. Dr. Andreas Oberweis
Supervisor:	M.Sc. Konstantin Pandl
Submitted:	15. April 2021

Table of Contents

Abstract

As Machine Learning approaches get more sophisticated, many applications of machine learning in the healthcare sector are proposed (Mathur, 2019). However, medical data is often distributed across multiple institutions (i.e., hospitals, insurance providers etc.) and cannot be aggregated due to privacy concerns and ownership structures (Brisimi et al., 2018). Federated Learning (FL) is a technique that can be employed to enable distributed Machine Learning without sharing the underlying data (Kairouz et al., 2019; McMahan et al., 2016). There have been many approaches implementing federated learning in conjunction with blockchain technology. This avoids the single point of failure of centralised federated learning and promotes decentralisation. However, there are many competing methods and approaches on how to integrate the blockchain layer into a FL scheme. This thesis gives an overview on existing approaches, determines important requirements on the blockchain layer, then compares existing blockchain solutions and evaluates promising candidates using practical experiments.

A systematic literature review on the use of FL with blockchain technology was conducted, after filtering, 49 approaches examined and categorised according to their mode of integrating the blockchain into a federated learning system. The analysis of requirements for the blockchain layer pointed towards a private permissioned blockchain with emphasis on security, reliability and flexibility. Afterwards, different blockchain solutions found in literature were compared. The blockchain solutions *Ethereum* and *Hyperledger Fabric* were further analysed using experiments. The analytic and experimental results suggest that the use of the blockchain solution *Hyperledger Fabric* is well-suited for facilitating a federated learning system in a healthcare setting.

List of Figures

List of Tables

1. Introduction

1.1. Problem definition

In the last years, the field of artificial intelligence, including machine learning, has received very high attention and funding. Hand in hand with these advanced methods of analysing and interpreting patterns in data goes the call for more privacy-preserving measures. Especially in the field of healthcare, there is demand for centralised machine learning on medical data (Beam & Kohane, 2018) e.g., medical image recognition or clinical outcome prediction for structured patient data. However, especially in the health care sector privacy concerns are a delicate issue, as sensitive patient data is involved (Li et al., 2020). These privacy issues can often hinder ambitions for inter-institutional machine learning. A possible solution for these privacy concerns can be federated learning, where the learning process is done locally (e.g., for each hospital), however, a global ML-model is being trained by aggregating the locally computed weights (parameters of the trained ML-model). The global aggregation process is usually done by a centralised server. This creates a single point of failure, where a corrupted central server can render the learning process useless or manipulate results. Distributed Ledger Technologies (DLT) have been proposed to replace the centralised server and the distributed, trustless mode of operation seems fit for this task and has also seen first successful applications (Pandl et al., 2020).
As integrating a blockchain layer in the federated learning system appears to be promising, many different approaches to it have been proposed. There are multiple ways to leverage the potential of blockchain-orchestrated federated learning and it is currently not clear which blockchain infrastructure is best suited for the task of federated learning and which integration method is best suited for the blockchain layer. To determine which blockchain solutions are the most useful for this task, it must be clear which requirements are made on the blockchain layer, which leads directly to the first research question:
R1: What functional requirements are there for the blockchain-layer of blockchain-based federated learning in health care?
After having established these needed characteristics, R2 arises:
R2: Which existing blockchain solutions match these requirements best?

Answering these research question will benefit researchers by helping them make an informed decision about which blockchain ecosystem to use and how to integrate the blockchain layer into a federated learning scenario. For the scenario, a healthcare setting is assumed.

1.2. Research Objectives

The main objective of this thesis is to define how blockchain infrastructure can be utilised in a federated learning approach, and what requirements are made on this blockchain layer. Further it will be elaborated how existing blockchain solutions fit these requirements.
These main objectives can be split in 4 sub-objectives:

Sub-Objective 1:	What are the approaches and scenarios to use blockchain technology for federated learning?
Sub-Objective 2:	What are the requirements for blockchain solutions to enable their application to federated learning?
Sub-Objective 3:	How do different blockchain solutions fit these requirements?
Sub-Objective 4:	How do selected blockchain solutions perform in practical experiments?

Answering these questions will benefit future researchers by helping with the decision which blockchain infrastructure to use in a blockchain-assisted federated learning infrastructure.

1.3. Structure of thesis

The remaining work is organised in the following way: First, the main topics will be introduced, which are *federated learning* and *blockchain technology* and then it will be described how blockchain technology can be used to facilitate federated learning. After describing the methods, the results of the literature review will be described and core requirements for the use of blockchain in a federated learning scenario for healthcare derived. The latter are then compared to existing blockchain solutions. After selecting the most promising candidates, the results of some practical experiments with the selected blockchain solutions are presented.

2. Blockchain based federated learning

2.1. Federated learning

The term *federated learning* was coined by McMahan et al. (2016), to describe a scenario where a machine learning (ML) task is carried out by multiple entities, which each have a distinct dataset. Those clients train a ML model locally in parallel and then globally aggregate their training results into a global model, which is then again distributed to all clients for the next round of training (Kairouz et al., 2019). This principle found its first broad application in Google's GBoard (an Android keyboard app) optimisation where it was utilised to optimise word recommendations (Yang et al., 2018). The unique characteristics of federated learning allow it to perform machine learning on user input without sharing this input data, which would have violated the users' privacy. This well-known use case can be assigned to *cross-device federated learning*, where a large number of small edge devices perform the learning task. Another form of federated learning is *cross-silo federated learning*, where fewer bigger nodes perform the learning task (Kairouz et al., 2019).

Considering healthcare applications of federated learning at hand it will generally be assumed a *cross-silo* scenario, where the participating nodes could be hospitals. These form a cluster in order to use federated learning for generating a shared model without sharing patient's data records and thereby respecting the privacy of the patients.

2.2. Blockchain technology

Blockchain is a form of Distributed Ledger Technology (DLT) where a ledger is maintained in a distributed manner. The blockchain implementation of Bitcoin (Satoshi Nakamoto, 2008) solved the *double spending problem,* what refers to a classical problem with distributed ledgers where a adversary spends a token two or more times. The resulting cryptocurrency *Bitcoin* was widely adopted, a lot of research has been done in the field of Blockchain and DLT generally, and "[DLT is] one of the most promising innovations in the field of information technologies with the potential to change organization and collaboration in the economy, society, and industry." (Sunyaev, 2020, p. 265)
Blockchain technology relies on a peer-to-peer infrastructure, which records transactions between participants on an immutable ledger. These transactions are stored in blocks of fixed size which each reference their predecessor. Therefore, it is called a *block*chain. The participating parties must agree on a valid version of the ledger, which is ensured by a consensus-protocol. Most consensus models are based on the following mechanisms employed to identify the block leader:

- *Proof of Work (PoW)*: The nodes verifying transactions - "mining nodes" - try solving a mathematical task, the first one to find the correct verifiable solution broadcasts this to the other nodes and claims a reward. For Bitcoin, the task to solve is generating a random nonce which is appended to the block, so that the hash of the block is smaller than a set value λ (Satoshi Nakamoto, 2008). By changing λ the difficulty of the PoW can be controlled.
- *Proof of Stake (PoS)*: The PoW protocol consumes a lot of computing resources on the hashing task, which goes "wasted". It is estimated that in 2018, bitcoin alone consumed around 2,55 gigawatts, and may in the future rise to 7,67 gigawatts, which is comparable to the energy consumption of Austria (Vries, 2018). To solve this problem, a less consuming approach was developed: Proof of Stake. Here a random Stakeholder (i.e., a user having a positive account balance) is chosen to verify the transactions. This runs under the assumption that users having a higher account balance are interested in keeping the integrity of the blockchain – because in case of an attack also their own assets would be devalued (Saleh, 2018).
- *Delegated Proof of Stake (DPoS):* DPoS is related to PoS, in so far that the holders of tokens (stake) are assigned the power over the network. However, with DPoS, the stakeholders cannot become block leaders themselves, but can vote for candidates. These then perform the mining process when enough users voted for them. One exemplary blockchain that uses DPoS is the EOS Blockchain (block.one, 2018), where all users possessing tokens vote on the "Block Producers".
- *Proof of Authority (PoA)*: When the real identity of stakeholders is public knowledge and proven through a certification process, PoA presents an alternative to PoW or PoS. With this protocol

a small group of actors (Committee) assumes leadership and verifies the transactions of all participating parties. The small group of validators (Committee) is to be monitored and is required to operate independently (Xiao et al., 2020).

2.3. Blockchain based federated learning in literature

Instead of using a central server for the aggregation and distribution of the trained weights, a blockchain infrastructure can serve as a decentralised alternative. The main motivation for replacing the central server in the FL architecture is to eliminate the single point of failure. As depicted in Figure 1, the central coordination of the server is relayed to the blockchain, where the business logic (weight aggregation, rules for participation, …) can be implemented via smart contracts.

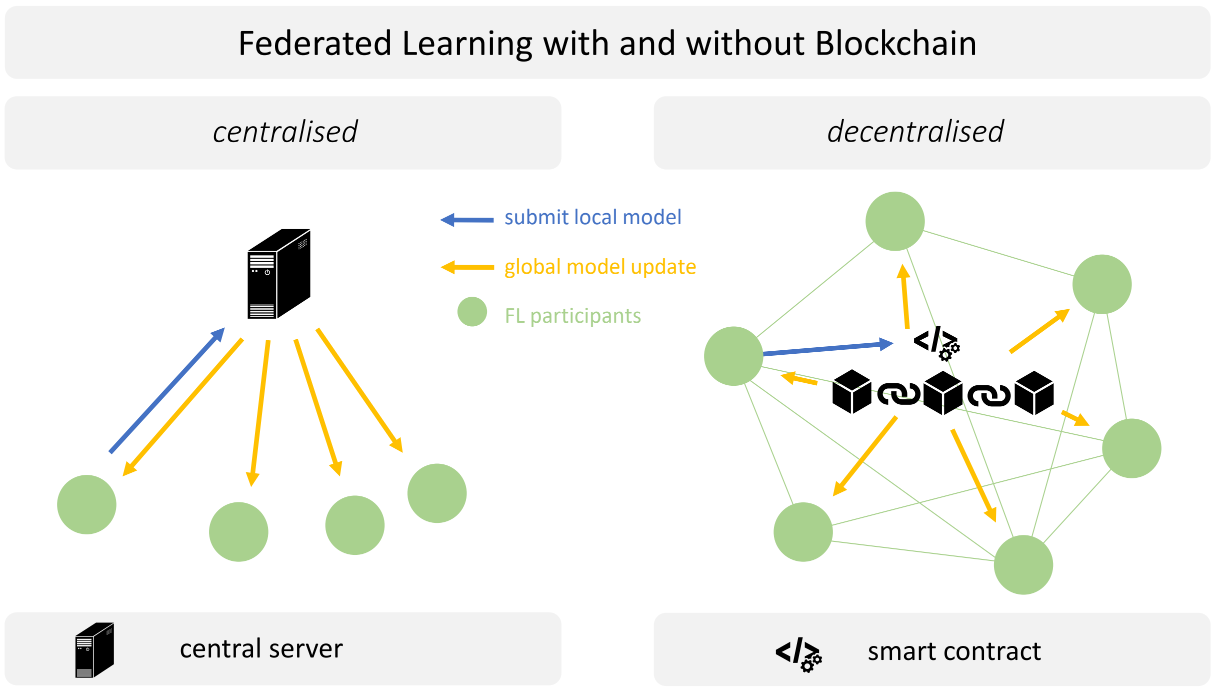


Figure 1: Architecture comparison between centralised FL and blockchain-based FL

This implies that all participating nodes maintain a full node of the blockchain, so as long as the distributed ledger stays synchronised, no further communications between peers is necessary. While submitting new weights via a smart contract changes the state of ledger and therefore requires a transaction and broadcasting of the changes to the other nodes, a lookup of weights stored on the blockchain can usually happen without generating a transaction and changing state, depending on the implementation details.

Short et al. identify five main aspects of the federated learning process that are advanced through the use of a blockchain in a FL setting (Short et al., 2020, p. 1185):

1) “Data integrity and Reliability”
   By replacing a vulnerable central server with “inherently secure” blockchain technology, stored data is protected against manipulation. Here Short et al. refer to the mechanism of blockchains where newer blocks refer to a checksum (hash code) of older blocks, which prevents unnoticed changes on older blocks.
2) “Reliability”
   Because of the redundant structure of the blockchain, the failure of a single node would not affect the system’s reliability as all nodes store all information. So not affected nodes would compensate if one or multiple nodes are not responding or faulty.
3) “Trust”
   Blockchain also enables the cooperation of untrusting parties, through its “security and traceability properties”. By that Short et al. refer to the consensus mechanism, which enables finding a stable consensus in the presence of faulty nodes or adversaries.
4) “Possibilities for incentives or rewards”
   Blockchains’ first application, digital assets, can be exploited to support incentive, for example in the form of reward distribution for good training results.
5) “Auditability – Traceability – Accountability”
   Because every transaction, and with it every action, is recorded on the distributed ledger, a system fault can be traced back and the malicious clients can be held accountable. This is possible because all past transactions are stored on the blockchain and users can not deny the participation in wrongdoing, since transactions are signed by the initiating party (*non-repudiation*).

As the benefits are now established, it arises the question as to how the blockchain can be integrated into the FL architecture. In the located literature there are three main ways identified for how the blockchain layer can be integrated into a federated learning setting.

1. **Model on Chain: Sharing the model parameters between participants through a designated blockchain with specialised node infrastructure.**
   With this mode of deployment, the blockchain serves as a decentralised storage infrastructure that stores the model weights or similar data, and therefore contains the ML model. Due to large models and storage space restrictions on some custom blockchain solutions, a custom built blockchain seems to be used more often for this. Building a custom blockchain where the node infrastructure as well as the consensus mechanism are specifically designed for the task of federated learning and developers have more fine-grained control over the design of blockchain and its characteristics has advantages that might be worth the extra effort.
   Generally the implemented blockchains can be characterised as private permissioned (nodes have to be verified) or public permissionless (anyone can join the network) according to the scheme developed by Kannengießer et al., 2020. The public permissionless approach is more often used in an IoT context, see for example ur Rehman et al., 2020. In a healthcare scenario however, a private

permissioned setting will be assumed. As the users of the system will be hospitals or similar entities, new entities have to prove their identity before read & write permissions are granted.
However, the storage of the model can theoretically also be done using existing blockchain solutions. For example Ramanan & Nakayama, 2019 store the ML-model on a Ethereum Blockchain by chunking it into multiple transactions (24 kB each).

2. **Key to Model: Sharing only a pointer to the model parameters file through a blockchain using smart contracts on traditional Blockchain 2.0 platforms, while the real model data stays on a decentralised storage solution.**
   Because the size of the model's data can be quite large, especially for large Convolutional Neural Networks (CNN) which are often used for the analysis of (medical) images, and storage on the blockchain is very expensive, a solution can be using a third-party storage system, that is accessed with a cryptographic key. The storage of the model's parameters can therefore be performed by the distributed storage system and only the key is stored on the blockchain to save space. An often cited distributed storage system in the context of FL (Arachchige et al., 2020; Awan et al., 2019; Cai et al., 2020; Hu et al., 2020; Kumar et al., 2020; L. Liu et al.; Martinez et al., 2019; Mugunthan et al., 2020; Passerat-Palmbach et al., 2019; Ratadiya et al., 2020; X. Wu et al., 2020) is the *InterPlanetary File System* (IFPS) (Benet, 2014), however other distributed storage systems can of course work as well. With IFPS, every file is stored on the distributed network and can be located using the files' hash value. This way all federated learning participants can access the global model parameters after retrieving the hash value from the blockchain. As on-blockchain storage restrictions are bypassed, this also facilitates the use of existing blockchain solutions, and a wide range of systems have been developed this way.

3. **Audit trail Blockchain: Recording meta-information on a blockchain system.**

Some approaches also use the blockchain as a complement to centralised federated learning, for example to store a trace of all actions or a record of participants' reputation for reference, while the actual FL process is happening through other means.
For example Kang et al. store information about participants' reputation on the blockchain. This ensures that model updates are verified, and malicious participants can be identified (Kang et al., 2020). The blockchain can guarantee immutability and verification in these scenarios.
This can also be in conjunction with another blockchain for handling the model data – for example Desai et al. developed a hybrid scheme, in which a permissioned blockchain handles the aggregation of model data while reputation records are stored on a public blockchain which is not directly involved in the learning process.

When it comes to the type of blockchain solution in place, three scenarios can be identified:

1. Specifically designed custom blockchain
2. Use of smart contracts on existing blockchain solutions
3. Hybrid solutions, employing multiple blockchain solutions in the system

In Figure 2, a quantitative overview over the identified literature is given, depicting what mode of operation was chosen and how it was implemented (n=49). Citations of the underlying papers can be found in Appendix A.

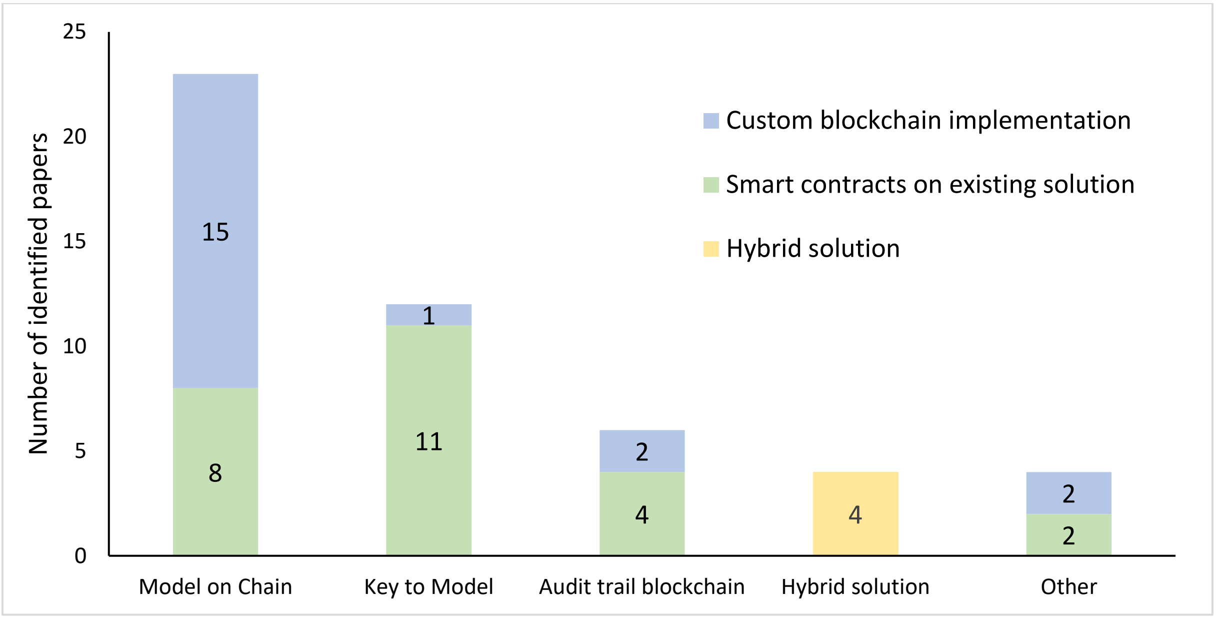


Figure 2: Quantitative overview of literature using blockchain solutions for federated learning.

As can be seen, for the “Model on Chain”-approach, a tailored blockchain solution seems to be more common, while implementations using existing DLT designs (with smart contracts) often use the “Key to model”-approach. This can be explained by the storage restrictions on traditional blockchain clients, while a custom implementation has no limits for the size of stored data, other than performance issues.

2.4. Attack vectors and countermeasures

Although the federated learning infrastructure theoretically keeps the information used for training the model inherently private, there are remaining risks, which will be addressed in this subchapter. A federated learning system processing sensitive healthcare data should by design be secured against privacy breaches – following the guidelines of *Privacy by Design* (Ann Cavoukian, 2011). To reach this goal, the algorithm as well as the data must be protected accordingly (Kaissis et al., 2020).

2.4.1. Attack vectors on the algorithm

The federated learning algorithm is usually guided by a central server which would be able to identify malevolent participants (anomaly detection). Due to the envisioned distributed architecture, the algorithm must be protected by other means against attacks. One attack strategy on federated learning is **model poisoning**, i.e., manipulating the joint model either in a destructive way or for the final model to behave how an attacker wishes for a subset of training data (Bagdasaryan et al., 2018, p. 4). In a healthcare scenario, this attack is relevant, one could imagine for example a rogue healthcare provider or hackers submitting poisoned model data to manipulate the outcomes of a recommendation system (e.g., leading to the recommendation of more expensive drugs). Countermeasures to model poisoning attacks include the review of submitted weights, this will be further discussed in chapter 4.4.
In a public unpermissioned system **sybil attacks** must also be accounted for. Sybil attacks refer to an attacker creating multiple participating accounts to skew voting results in their favour or poison the model (Fung et al., 2018). However, in a private permissioned scenario sybil attacks can usually be neglected, unless it is taken into account that multiple verified participants can get corrupted by a third party.

2.4.2. Attack vectors on the data

Model inversion attack: Although the sensitive data stays with the separate parties conducting the federated learning, an attacker who gets access to the model can use it reconstruct the used feature vectors - this is called *model inversion attack*. From there a *reconstruction attack* could even, to an extent, reveal underlying raw data that was used for training the model (Al-Rubaie & Chang, 2019). The *model inversion attack* is only interesting for models where inverting the direction of the model could reveal sensitive information. For example it was shown for a face recognition model, which takes image data as input and predicts identities, that images of a person can be recovered given a person’s name, even though a black-box model was employed. This was achieved by minimising the confidence values for different generated input images (Fredrikson et al., 2015, p. 1323). For anonymised data and normal classification tasks, this attack can hardly gather sensitive data. Furthermore, the quality of reconstructed data points could be significantly reduced by applying small amounts of obfuscation e.g., rounding confidence values given with model predictions (Fredrikson et al., 2015, p. 1323).
Membership inference attack: In a membership inference attack an adversary having knowledge of the model data can infer whether a specific datapoint was used to train this model, which is concerning in a healthcare scenario: when an adversary can learn for which ML-model a person’s data records were used for training, this could implicate this person’s clinical history. This attack can effectively be prevented with *Differential Privacy* (DP), which was originally invented for database systems but can also be applied for machine learning scenarios: DP guarantees that adding one element to a set *D* does not

alter a query's result, except for the privacy factor ε, which means a query cannot infer whether a specific element is present in the set, in other words *membership inference attacks* are not possible. Formally DP states that the output R_0 of the randomising function f behaves the same (except for the privacy factor ε) when applied on the Datasets D_1 or D_2, which differ at only one element (Dwork, 2006, p. 8):

$$\Pr[f(D_1) = R_0] \leq \exp(\varepsilon) \times \Pr[f(D_2) = R_0]$$

DP is usually implemented by injecting noise into the data, which produces an inherent trade-off between privacy and precision, however, especially for medical data, this bargain should be worth it. Differential Privacy can even guarantee privacy when the modified data is publicly released, which was for example applied in the case of genomic data (Raisaro et al., 2018).

Other additional **countermeasures** from the toolbox provided by PPML can be added to the system design, the most often used approaches include *Homomorphic Encryption* (HE), *garbled circuits*, *secret sharin*g and *secure computation* (Al-Rubaie & Chang, 2019, pp. 53–54).

2.5. Related approaches

DLT concepts other than blockchain can be used to facilitate federated learning in a similar fashion. Directed Acyclic Graphs (DAG) have been used by Schmid et al. to implement a federated learning framework. They use a DAG because "unlike a blockchain, a tangle does not rely on one single chain being the single source of truth, the idea is rather to reach consensus on a partially ordered set of transactions", which led to easier synchronisation in their IoT scenario (Schmid et al., 2020, p. 854). Lu, Huang, Zhang, et al. also made use of a DAG in a Federated Learning setup for Vehicle to Vehicle (V2V) networks. They propose using a DAG on a local level to coordinate model training rounds while additionally using a permissioned blockchain to share the new model globally (Lu, Huang, Zhang, et al., 2020a).

Furthermore, there are a few alternatives to Federated Learning aiming to share model data without compromising the underlying data. In contrast to Federated Learning, where participants train a model in parallel and their updates are aggregated, Institutional Incremental Learning (IIL) lets each institution train a ML model once. Then the model is passed on to the next institution that continues the training with their data, until all institutions have trained on the model (Sheller et al., 2019). Cyclic Institutional Incremental Learning (CIIL) only trains for a fix number of epochs per institution before going to the next institution (after a full cycle the next epochs are trained), so that the data is more balanced, which combats "catastrophic forgetting". This refers to unlearning features learned before and can occur when the model is trained in a sequential way (Kirkpatrick et al., 2017). Chang et al. also evaluate different heuristics for training on distributed data, including methods similar to IIL and CIIL, but also a heuristic for combining different models trained in parallel by different institutions after the learning process ("ensemble single institution models") (Chang et al., 2018, p. 947). Federated Learning is however still considered the best performing method for learning on distributed data, Sheller et al. note that "[other examined methods] fail to match the performance of federated learning" (Sheller et al., 2019, p. 1).

Another pre-printed paper discusses using the computations of the FL process to replace Proof of Work for a blockchain system (X. Qu et al., 2019), although the paper does not go into implementation details, future research in that direction could in theory reduce the waste of energy by Proof of Work to the benefit of federated learning.

3. Methods

3.1. Literature Review

For a holistic overview of the literature on the topic, a systematic literature review (Snyder, 2019) was conducted. To find available articles on the topics, multiple academic databases were searched, and the search results then screened for relevant works. Because the topic is recent, not yet peer-reviewed *pre-prints* from arXiv[1] were also included. As not many papers on the topic are focused on the healthcare domain, all papers using blockchain technology for federated learning had to be considered. The search string "'blockchain' AND 'federated learning'" was modified for each database using the LitSonar[2] service and following databases were searched (hits are the remaining ones after removing duplicate papers that were already found in the other databases):

- **IEEExplore** *(https://ieeexplore.ieee.org)*: 47 hits, 41 relevant
- ***AIS*** *(https://aisel.aisnet.org/)*: 3 hits, 0 relevant
- ***ArXive*** *(https://arxiv.org/):* 18 Hits, 13 relevant
- ***ACM*** *(https://www.acm.org/)*: 9 hits, 3 relevant
- 2 additional papers were identified because they were cited by other papers.

The resulting 59 relevant papers were then examined with regard to their blockchain-based FL approach: For each system implementation the used blockchain solution, the mode of implementation, main characteristics and the domain of application were extracted. Because ten of the papers did not clearly state their approach, but were of more theoretical nature, 49 approaches were fully analysed in the end.

3.2. Requirement analysis

In this thesis, *requirement analysis* will refer to the process of finding out what external requirements exist and what characteristics would be positive for the blockchain solution employed in a federated learning setup.

In the literature review, mentions of beneficial characteristics for the blockchain layer of the system were extracted to distil basic requirements. When no concrete requirements or reasoning was stated, general requirements, for example "smart contract-functionality" were extracted. As stated before, there

[1] ArXiv is a free distribution service and an open-access archive for scholarly articles: https://arxiv.org/
[2] LitSonar offers modification of search strings for many databases: http://litsonar.com/

are different ways of how a blockchain layer can be integrated into a federated learning system. Obviously, the requirements differ for each type of implementation, so this chapter will first define basic requirements and then explain differences dependent on the type of implementation. Corresponding to the scheme provided by (Kannengießer et al., 2020), it was systematically described which characteristics a blockchain should have to be employed in the described scenario.

4. Requirement analysis

4.1. Scenario and basic requirements

The envisioned scenario consists of multiple (~ 5-10) medical organisations (i.e., hospitals, insurance agencies, …) which decide to collaboratively train a ML-model. As laws forbid the sharing of the underlying medical data, they decide to use federated learning to share insights about their data without sharing the underlying data. To ensure traceability, accountability and transparency while eliminating the single point of failure of a central coordination server, they use a blockchain layer to facilitate a distributed federated learning process. Related to this scenario, following circumstances are assumed:

- All federated learning participants are known.
- Trust is not assumed as participating entities could be corrupted/hacked.
- An always-online central server is not a solution, because this single point of failure is what we want to eliminate.
- Public verifiability is not required, only participating parties verify transactions.

Following the scheme provided by K. Wüst and A. Gervais, this scenario points towards a **private permissioned blockchain** (K. Wüst & A. Gervais, 2018, p. 47), which means only accepted entities can read and write on the blockchain and new participants have to pass a verification before they can participate in the network. Given this assumption, there are multiple basic requirements on the blockchain layer, which are in the following ordered according to the DLT characteristics and properties identified by (Kannengießer et al., 2020, 42:12 - 42:15). Table 1 gives an overview of which DLT characteristics are especially important for facilitating a federated learning system in a healthcare setting, depending on the mode of implementation (scale: 0 = "not important" to 3 = "very important"). For the modes, the most common variants were selected: 1) Sharing the model data using a custom-built blockchain; 2) Utilising smart contracts on an existing blockchain solution with external storage infrastructure and 3) Running a complementary blockchain in parallel to the FL process, to record participants actions or reputations.
The main requirements on characteristics of the employed blockchain layer are described in the following:

DLT Properties	DLT Characteristics	Model on designated blockchain	Smart contracts on existing blockchain	Audit trail/reputation blockchain
Flexibility	Interoperability	2	2	2
	Maintainability	1	1	1
	Turing-complete Smart Contracts	0	3	3
	Token Support	1	1	2
	Transaction Payload	3	1	1
Opaqueness	Traceability	2	2	3
	Transaction Content Visibility	1	1	3
	User Unidentifiability	0	0	0
Perfor-mance	Throughput	3	3	1
	Latency	2	2	2
Prac-tical-ity	Easy Node Setup	1	1	1
Security	Authenticity	2	2	2
	Availability	3	3	1
	Confidentiality	3	3	3
	Integrity	3	3	3
	Non-Repudiation	2	2	3
	Reliability	2	2	1
	Strength of Cryptography	3	3	3

Table 1: Overview of the importance of DLT characteristics for selected scenarios

Flexibility

Good *Interoperability* plays a role in healthcare, as legacy systems must be able to contribute their outputs to the FL system in order to minimise the initial cost of adopting the FL system. An accessible API is necessary to enable the use of the system. A common API is for example to use remote procedure calls (RPC) over HTTP, which should be compatible with most computer systems. A high degree of M*aintainability* is also beneficial to enable continuous operation of the system over longer timeframes. The ability to implement *smart contracts* is a necessity when an existing solution is used, with a custom solution however, the application logic can be realised in the node infrastructure. Therefore smart contracts are not necessary in that case. When a reward mechanism is to be implemented, *Token Support* is also necessary to enable a pay-out functionality. The *Transactions Payload* determines how large data packets associated with transactions can be. This is a key factor if an implementation plans on storing the ML model on the blockchain. Especially models handling high dimensional input such as images – for example Convolutional Neural Networks (CNN) - can easily outweigh the size limitations imposed by some existing blockchain solutions, so these use external solutions (IPFS).**Opaqueness:** Due to the nature of the envisioned private permissioned system, a high degree of *traceability*, *visibility* and *node controller verification* is necessary, especially so if the blockchain is to be used for auditing purposes.

Policy

Here a trade-off between patients' data security (*compliance*) and *auditability* has to be considered. The system must be auditable and verifiable, but actual patients' data cannot be examined by a third party, as preventing this is the whole point of this system. The *incentive mechanism* is also important, this is further described in Chapter 4.4.

Performance:

Because of the cross-silo approach some performance aspects are not as important as in a cross-device scenario, however a good *throughput,* which is usually measured by how fast the DLT design can execute transactions, is influencing the training time and should be very high to not slow down the FL process. A low *latency* is also helpful.

Practicality:

An *easy node setup* could lower the entrance barrier for participating parties, however here a trade-off with *node controller verification (opaqueness)* exists, which is necessary for the permissioned mode of operation (Kannengießer et al., 2020, 42:17).

Security:

Security is a crucial aspect of the blockchain layer in the envisioned system because sensitive health data is concerned. High *Authenticity* and *Integrity* must not be compromised to prevent corruption of the ML model. High degrees of *Availability* and *Reliability* are important, especially when the system relies on the blockchain and the FL process cannot be performed without it. Generally only blockchain solutions providing flexible and strong security mechanisms are to be considered. *Non-Repudiation*, referring to the prevention of the ability to deny actions performed by participants is also favourable, especially for the audit-trail mode, so that accountability is guaranteed.

4.2. Requirements for designated blockchain solutions

Sharing the model parameters through the blockchain is a common approach, which is mostly realised by implementing a designated blockchain framework for this task (Bao et al., 2019; Chai et al., 2020;

Kim et al., 2018; Kuo & Ohno-Machado, 2018; Lu, Huang, Zhang, et al., 2020c; Lugan et al., 2019; Ma et al., 2020; Majeed & Hong, 2019; Pokhrel & Choi, 2020; Y. Qu, Gao, et al., 2020; Y. Qu, Pokhrel, et al., 2020; Shayan et al., 2018). This implies slightly different requirements:

Because whole model updates are stored on the ledger, the implementation using smart contracts and traditional Blockchain 2.0 solutions do often not work, because the file size of the model parameters is larger than the *block size* or the *transaction payload*, as most traditional blockchain solutions were not designed for storing large data. Therefore, the custom implemented solution has to ensure sufficient transaction payload and block size. With custom blockchain solutions, the logic (weight aggregation) can happen in the node infrastructure layer, so that smart contracts are not needed in this case. As high amounts of model data have to be synchronised between nodes, low *latency* is especially important.

4.3. Requirements for existing blockchain solutions

To implement a blockchain based federated learning system there is no absolute necessity to implement a new blockchain, as there are already plenty existing blockchain implementations. Especially Blockchain 2.0 solutions which encompass smart contracts are very flexible and can be used to enable blockchain-based federated learning.

There are scenarios where the FL process is done using a public blockchain such as the public Ethereum chain. This approach can be especially useful in a mobile edge computing (MEC) scenario (ur Rehman et al., 2020). However, many implementations prefer to use existing blockchain solutions in a private setting, as in a public setting computationally expensive additional encryption can be necessary to prevent information leakage (Arachchige et al., 2020). The scenario at hand is not really suited for a public blockchain, as only verified participants should have access. Therefore, a private permissioned scenario will again be assumed, where only validated nodes get access to the blockchain network. This implicates the first requirement: the used blockchain solution must provide an option to implement standalone private permissioned blockchains.

To allow its use for federated learning, an existing blockchain ecosystem generally also must have a way to implement *smart contracts*. Within these smart contracts, federated learning functions can be implemented.

As the *transaction payload size* is given by the used ecosystem, many solutions resort to the use of an external distributed infrastructure such as IPFS (Benet, 2014), only storing the key to the external model data on the blockchain. This allows for a lightweight implementation where the transaction payload size is no longer a bottleneck.

Another important requirement is sufficient *flexibility*:

- The ability to freely choose a consensus mechanism.
- The ability to implement an incentive mechanism.

Those can help with customising the blockchain solution to fit better in a FL-architecture.

Also, a high degree of *security* should be achieved with the blockchain solution, as discussed above. Private key authentication with certificate authorities as implemented by some blockchain solutions could for example secure the access to the network.

4.4. Mechanism design

To improve the efficiency of the system, incentives for good behaviour can be integrated. A common way of realising rewards is via the consensus protocol: Novel block leader selection / consensus mechanisms designed for FL have been proposed to ensure a high level of integration of the blockchain layer into the FL system:

For example with *Proof of Information*, as used in ModelChain (Kuo & Ohno-Machado, 2018), the local client with that data that produces the highest error rate is chosen for the next iteration of training and validation, assuming data producing high error with the current model implicates that training with his data will add new information to the global model, “aiming at increasing efficiency and accuracy” (Kuo & Ohno-Machado, 2018, p. 6).

Other mechanisms establish a *reputation reference* to determine the committee leader in a *Proof of Authority* scenario: FLChain (Bao et al., 2019) implements a block leader selection algorithm, where the leader for a training round is selected based on the clients past reliability, while (Lu, Huang, Zhang, et al., 2020c) use a modified form of *Delegated Proof of Stake*, while the “stake” is adjusted according to each clients training performance, so that the best performing participants can specify the block leader. Because this mining process is then rewarded, good training performance is incentivised.

In the envisioned cross-silo healthcare scenario, availability is assumed according to (Kairouz et al., 2019, p. 6) so that some mechanisms can be neglected. These were often developed for IoT scenarios and are focused on selecting reliably available clients for the verification of transactions. In a healthcare scenario, the requirements for a good mechanism design boil down to:

- **Prevention of poisoning attacks**

  Poisoning attacks can be avoided by validating submitted models before they can become consensus. It can be argued that model poisoning attacks (Bagdasaryan et al., 2018) are irrelevant due to the private permissioned scenario, however, attacks on the data or model are still relevant in case a participating party gets corrupted. For this case, a robust mechanism design can prevent such attacks, as long as there are not too many attackers: Short et al. presented how the evaluation of model updates can effectively deny poisoning attacks and how this can be implemented using a smart contract approach (Short et al., 2020, p. 1178).
- **Computational efficiency**

  The choice of the block leader selection protocol has high impact on computational efficiency. For example, *Proof of Work* approaches are very energy and time consuming. Especially for the model-on-chain architecture (as hash rates are negatively influenced by transaction size) – so that in order to verify large model updates mining nodes would have to use even more power when calculating the Proof of Work. Proof of Authority solves these issues, as no computationally task has to be solved.

- **Incentivising participation**

  A very useful feature can be to incentivise training performance by rewarding model improvements by provision of services or monetary incentives. A token economy which rewards upload of model updates, or also verification (mining) by distributing digital tokens can effectively combine blockchain assisted federated learning with blockchains' more common use for cryptocurrencies: Kim et al. propose distributing a "mining reward as well as "training reward" in their BlockFL implementation (Kim et al., 2018, p. 2), Pokhrel and Choi also use this distinction in their work (Pokhrel & Choi, 2020, p. 4737). The mining reward incentivises mining in a similar way to traditional blockchain cryptocurrencies, while data reward is aimed at incentivising model contributions.

  The framework DeepChain (Weng et al., 2019) also implements an incentive mechanism and the authors reason: *"The introduction of incentive mechanism is crucial for collaborative deep learning, due to the following reasons. First, for those parties who want a deep learning model but have insufficient data to train the model on their own, incentive can motivate them to join the collaborative training with their local data. Second, with reward and penalty, incentive mechanism ensures that (1) parties are honest in local model training and gradient trading, and (2) workers are honest in processing parties' transactions."* (Weng et al., 2019, p. 5). This is even more relevant in an unpermissioned scenario, however, a certain degree of incentive for contributing training data and participating in the FL process can also be helpful in the context of healthcare. One could imagine for example, that hospitals must contribute a sufficient amount of their training data to get access to the model data and its insights itself.

5. Comparative analysis of existing blockchain solutions

This chapter will compare existing blockchain solutions and evaluate their fit to the requirements defined before. The focus is on the blockchain solutions which were identified in the literature review.

5.1. Blockchain solutions used for federated learning

Figure 3 gives an overview of which blockchain solutions were used to enable blockchain-based federated learning. As can be seen, 37% used the common protocol **Ethereum**, followed by solutions provided by the **Hyperledger** project, and only a few systems were implemented using more exotic blockchain ecosystems such as **EOS** (Martinez et al., 2019), **Libra** (Yin et al., 2020) or **Tangle** (Schmid et al., 2020). Some solutions also employed a **hybrid** approach with two distinct blockchain systems:

Fan et al. proposed using a private FISCO-BCOS blockchain with a consortium consensus mechanism for faster processing while also using the public Ethereum blockchain for making payments in the form of a better recognised cryptocurrency (Fan et al., 2020, p. 2255) .

In their PermiDAG proposal for Vehicle-to-vehicle data sharing, Lu, Huang, Zhang, et al. envisioned a global permissioned blockchain while at a local level directed acyclic graphs (DAG) were used for better performance (Lu, Huang, Zhang, et al., 2020b, p. 4303).

Finally Desai et al. proposed using Hyperledger Fabric for the implementation of a permissioned blockchain which performs the gradient aggregation and returns the aggregated model to participants through smart contracts, while a public smart contracts hosted on the public Ethereum *Ropsten* network is responsible for the tracking and verification of sent updates, also incorporating rewards and penalties (Desai et al., 2020, p. 4).

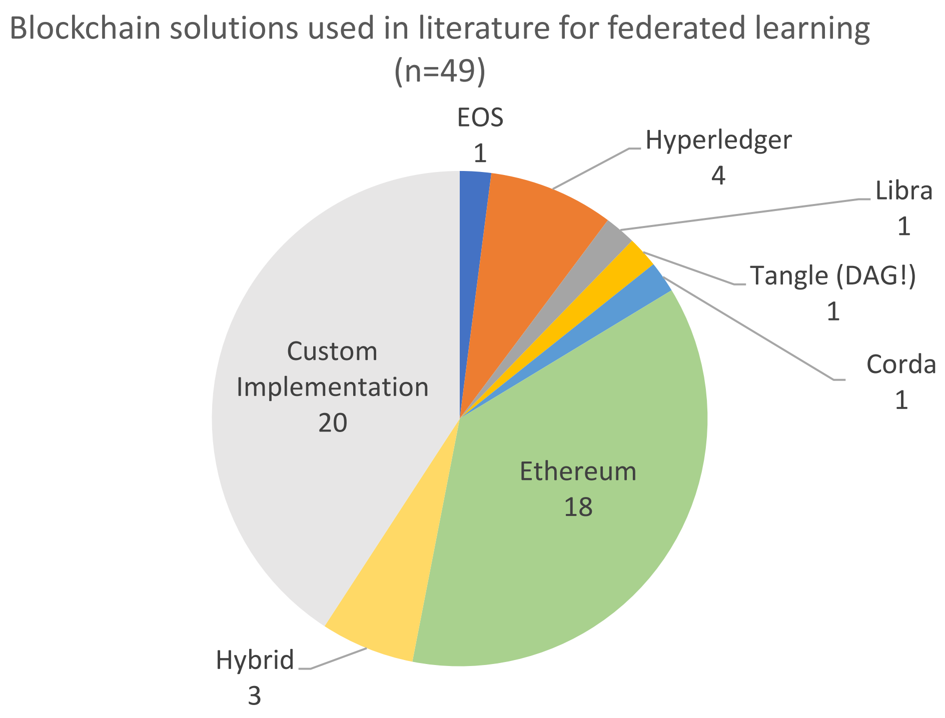


Figure 3: Blockchain solutions found in literature that were used to facilitate federated learning.

5.1.1. Ethereum

Ethereum was proposed in 2013 and was envisioned to provide a protocol for a blockchain-based ecosystem enabling the development of distributed applications. Ethereum provides Turing-complete smart contracts where rules for ownership, transaction formats and state transitions can be defined to match the desired properties (Vitalik Buterin, 2013). This enables a way more flexible system compared e.g., Bitcoin, where simple scripting exists, but loops are not allowed, and it can therefore not be Turing-complete. Smart contracts are the core component of the Ethereum network: just as normal accounts, contracts can have an address and be addressed via transactions triggered by accounts or other contracts.

The computations for contract execution and transactions are paid for with the internal cryptocurrency "Ether". Although there is a publicly available "Ethereum" Blockchain, developers can as well create their own, private, independent blockchain while leveraging the diverse functionalities of the Ethereum framework such as smart contracts.

Because of the broad use of Ethereum for distributed applications (Dapps), there exist numerous development resources, such as the *Truffle Suite*[3] which provides a development framework for Dapps with a low entry barrier. These can help when implementing a Ethereum-based solution.

Characteristics of Ethereum

As a generic blockchain platform, Ethereum provides a very flexible framework. It is governed by a community of developers and users and is constantly improved. It can be deployed in a public or private manner and provides permissionless access. The consensus model in the public Ethereum blockchain is currently built upon Proof of Work mining, however in the future the networks intends to shift towards a Proof of Stake based consensus model (Vitalik Buterin, 2013, Section "Currency and Issuance"). For private networks, the consensus model can also be modified to Proof of Authority (PoA), which saves computing resources but influences throughput (Transactions per seconds) and latency negatively (Schäffer et al., 2019, Fig. 2).

As for smart contracts, Ethereum supports the internal language Solidity[4], which was specifically designed for the implementation of smart contracts. Solidity supports numerous datatypes and enables on-contracts storage of data.

Relevant for the federated learning application is also the transaction payload size. For Ethereum there is no hard-coded payload size, however for transactions there is a "Gas Limit". As transactions are fuelled by "gas", this gas limit also restricts the amount of data that can be in the payload of a transaction. For the public blockchain this implies that storing large data objects on the blockchain gets prohibitively expensive – according to the Ethereum technical/yellow paper (DR. GAVIN WOOD, 2021, APPENDIX G) the *SSTORE* operation storing one "word" containing 256 bits (2^5 bytes) for the first time ("changing a value from zero to non-zero" costs 20.000 gas, while changing an existing value costs only 5.000 gas. Taking the average gas price of 111 gwei (1/1.000.000 Ether) (taken from etherscan[5] on 16.02.2021) the price of storing data on the blockchain (not including the additional transaction costs) can be calculated as

$$Cost_{Gas} * Cost_{SSTORE} * N$$

where N is the number of words to store. For storing 1 MB (2^{20} bytes) of data on the public Ethereum blockchain the equivalent cost would be:

$$\frac{111\,\frac{gwei}{Gas}}{1.000.000\,\frac{gwei}{Ether}} * 20.000\ gas * \frac{2^{20}\ bytes}{2^5\ bytes} = 72.744{,}96 \text{ Ether}$$

[3] Truffle Suite website: https://www.trufflesuite.com/

[4] Documentation for Solidity is found at https://docs.soliditylang.org

[5] Information on current gas costs taken on 16.02.2021 from https://etherscan.io/gastracker

which equals more than 100 million € with the current price of 1.482 €/Ether, while the retrieval of the data does not cost any gas.

As can be seen, data storage directly on the public blockchain is prohibitively expensive and limited by the block gas limit. Private solutions where gas and transaction costs do not play a big role have to be considered as they are more fitting to the federated learning scenario.

For the scalability in private scenarios there are many parameters that can influence the blockchain performance (Schäffer et al., 2019, Section 3): Block frequency, Block size, Workload type and quantity, Node configuration, Network parameters, Blockchain client and amount of participating nodes.

Evaluation for federated learning

Advantages - The public Ethereum ledger can be of use, as the underlying cryptocurrency Ether has the second highest market capitalisation[6] after bitcoin. As a trusted, globally recognised currency with cash-equivalent Ether can be useful for making payments e.g., for reward pay-outs. For the process of federated learning however, a private blockchain is to be preferred. The Ethereum framework offers building blocks to build a private blockchain, and there are many resources helping developers to fit their Ethereum solution to their needs. This flexibility and it being the most well-known Blockchain 2.0 design led to Ethereum being used the most for federated learning implementations. For realising Proof of Authority, there are extensions to the Ethereum protocol called clique (Péter Szilágyi, 2017). Mechanism design etc. can be realised using the smart contracts.

Disadvantages - Because Ethereum was originally designed for a public permissionless scenario, there are faster solutions for scalable private networks, Schäffer et al. note that “scaling is only possible to a limited extent due to the current design of Ethereum” (Schäffer et al., 2019, p. 116).

5.1.2. Hyperledger (Fabric)

Hyperledger[7] is an umbrella project hosted by the Linux Foundation providing a collection of open-source blockchain implementations. In the identified literature, the modular framework *Hyperledger Fabric* was the Hyperledger project used most often for implementing FL in a private permissioned scenario (Baranwal Somy et al., 2019; Desai et al., 2020; Kang et al., 2020), so only the subproject *Hyperledger Fabric* will be evaluated.

Characteristics of Hyperledger Fabric

Hyperledger Fabric is an enterprise-grade DLT platform developed by IBM amongst others. It is designed as a permissioned, modular platform e.g., the consensus model is implemented as an interchangeable module, so that developer can decide on which consensus model to use. In its core, Fabric is an

[6] Source: https://coinmarketcap.com/
[7] Hyperledger Website: https://www.hyperledger.org/

"operating system for permissioned blockchains that executes distributed applications written in general-purpose programming languages (e.g., Go, Java, Node.js)" (Androulaki et al., 2018). The smart contracts written in generic languages are called "chaincode". Data stored on the blockchain can be queried by using keys or JSON queries (hyperledger.org, 2020). The network consists of *Clients*, *Peers*, and an *Ordering Service*, where the *Clients* can submit their transactions and help coordinating. The *Peers* are validating and executing the transactions, and they maintain the ledger which keeps an immutable record of all transactions (the blockchain). The *Ordering Service* nodes establish the total order of all transactions, however they do not participate in executions or validation of transactions (Androulaki et al., 2018, p. 5). Fabric also brings the possibility to create multiple parallel blockchains called "*Channels*". With different *Channels*, the Blockchain can be viewed as a network of networks with isolated and private transactions in each channel which are still verifiable via hashes on the public chain (hyperledger.org, 2020).

The maximal transaction payload size on Hyperledger Fabric is currently not configurable for each blockchain, but hard coded into the transport layer protocol (gRPC). The current value of 100 MB is however very high compared to other blockchains e.g., Ethereum.

Evaluation for federated learning

Advantages

As Hyperledger Fabric was – in contrast to many other Blockchain designs – from the start designed for a private permissioned scenario, it has, at the time, advantages when used in a permissioned scenario. *Private Key Authentication* for participating nodes is supported by default, which enhances security. The highly modular design also satisfies the requirements made on the flexibility. Furthermore, the possibility to create separate channels with a shared ordering service can be beneficial in a federated learning scenario, for example Majeed and Hong proposed creating a separate channel for each global learning model, with the model details stored in each genesis block (Majeed & Hong, 2019, p. 2). This allows for the creation of multiple models without having to setup a completely new blockchain network for each one. Furthermore, the possibility of large payloads can enable Fabric to store more model information on the blockchain. In a small private scenario with low latency large blocks also pose no problem to the consensus process.

Disadvantages

Fabric is a relatively recent addition to the field of blockchain solutions (first release was in January 2018, first long-term-support version announced in 2019), so less practical application and evaluation has been performed so far, which could lead to unsolved issues or bugs when using it in a production environment. However, as more time passes, this disadvantage will resolve itself.

5.1.3. Other Blockchain solutions

EOS

EOS is based on a 2018 white paper (block.one, 2018) and is realised by the private company *block.one*. EOS aims to solve scaling issues of other blockchain solutions and, similar to Hyperledger Fabric, provides an "operating system-like construct upon which applications can be built." (block.one, 2018)
EOS uses Delegated Proof of Stake (DPOS) where all holders of tokens can vote for *block producers* which then verify the transactions of all users. Per round there are 21 selected block producers which work together to reach consensus (block.one, 2018). Block producers that miss blocks (meaning they do not deliver the block for a timeslot assigned to them) can get removed from the list of block producer candidates, so that reliability is ensured. Top block producers each amass around 3% of all votes.[8] These design choices result in an extremely scalable, performant blockchain while eliminating transaction fees. To facilitate Federated Learning, Martinez et al. intend using the EOS Blockchain in conjunction with IPFS to record uploaded models and also reward users according to the training work they did (Martinez et al., 2019). For their testing however, they resorted to Hyperledger Fabric for a private implementation, while "Future plans include a larger implementation of this system with EOS blockchain," (Martinez et al., 2019, p. 55). As this example shows, EOS strengths may lie at scaling performance and no transaction fees, which is not necessary for the envisioned scenario of federated learning in healthcare.

Corda

Another potential candidate for supporting federated learning is the blockchain system Corda (Brown et al., 2016; Hearn & Brown, 2019). Developed from R3[9], Corda is aimed at financial transactions and tries to unify financial processes to save transaction costs that arise when banks each keep their own ledgers by creating a valid global ledger. Kang et al. use the Corda blockchain with the consensus model Practical Byzantine Fault Tolerance (PBFT) to implement a reputation blockchain. From its design, Corda seems however not suited as underlying infrastructure of the FL process, may however work fine as a reputation blockchain.

Other permissioned blockchain solutions will not be further examined due to lack of scholarly sources. Although they are not that widespread and often used for similar tasks, those could theoretically also be viable in a federated learning setting: Symbiont/Assembly[10], Kadena[11], Quorum[12], HydraChain[13] or Exonum[14], to name a few.

[8] Source: Eos Authority - https://eosauthority.com/
[9] The company website: https://www.r3.com/
[10] Website: https://www.symbiont.io/technology
[11] Website: https://www.kadena.io/
[12] Website: https://consensys.net/quorum/
[13] Website: https://github.com/HydraChain/hydrachain
[14] Website: https://exonum.com/

5.2. Selection of most promising blockchain solutions

As can be seen in the last subchapters, the blockchain solutions most often used for FL in literature, as well as the solutions providing the best fit to the before defined requirements are 1) *Ethereum* and 2) *Hyperledger Fabric*. Accordingly, those two will be evaluated using practical experiments.

6. Practical experiments

6.1. Environment

To evaluate the blockchain solutions in a controlled and reproducible setting, virtual machines (VM) deployed by the Infrastructure-as-a-Service Provider bwCloud[15] were deployed.. For the experiments VM's with 2 GB of RAM and 1 virtual CPU (variant *m1.small*) were used, as the *m1.nano* variant with only 1 GB of RAM turned out to be insufficient for executing transactions on a running blockchain. As for the OS, both Debian 10 and Ubuntu 18.04 were used, there seem to be more online resources for troubleshooting when setting up blockchain solutions with Ubuntu, however, obviously both variants do work eventually. The VM's all operate on one local network ensuring low latencies. The software versions used: Ubuntu 18.04, nodejs (for Fabric) v10.20.0, nodejs (for Ethereum) v15.11.0, Hyperledger Fabric 2.3.1, go-ethereum 1.10.1., Truffle v5.2.3, Solidity (solc-js) v0.5.16.

6.1.1. Ethereum Deployment and Testing

All used code for the Ethereum tests can be found on GitHub[16]. To evaluate the Ethereum blockchain, the framework Truffle[17] was used. Truffle runs on NodeJS and can compile, deploy, and manage smart contracts for an Ethereum blockchain. Truffle also comes with a testing framework, which can be used to run tests written in JavaScript (for testing applications with contract interaction) or Solidity (for testing smart contracts). As for the Ethereum blockchain, ganache-cli[18] is another tool from the truffle toolbox that can simulate a private Ethereum blockchain. The ganache-cli client was ran inside a *tmux*[19] session to let the blockchain run in the background and the *truffle* was used for evaluation.

For testing the latency and runtime, a simple contract was designed which could store simple data objects and enables simple get and set functionality. This simulated a contract providing the global model to all participants and enabling them to update the model. Then, multiple participants (accounts on the blockchain) each retrieved the data from the contract, calculated the hash of the data to simulate a training round, and then sent the hashed data back to the contract. The next account did the same, until all accounts did this action once, which will be referred to as one "round". Then the runtime was recorded for

[15] Information about bwCloud: https://www.bw-cloud.org/.
[16] The code is found on GitHub at: https://github.com/vinzenzzinecker98/blockchain-tests.
[17] Truffle Suite website: https://www.trufflesuite.com/truffle.
[18] ganache-cli code available at: https://github.com/trufflesuite/ganache-cli.
[19] The tmux description: https://github.com/tmux/tmux/wiki.

multiple combinations of rounds and participants. Because per round and participant the data is updated once, which produces a transaction, the number of transactions is defined as $\text{participants} * \text{rounds}$, as expected from the setup there was no measurable difference between different participants.

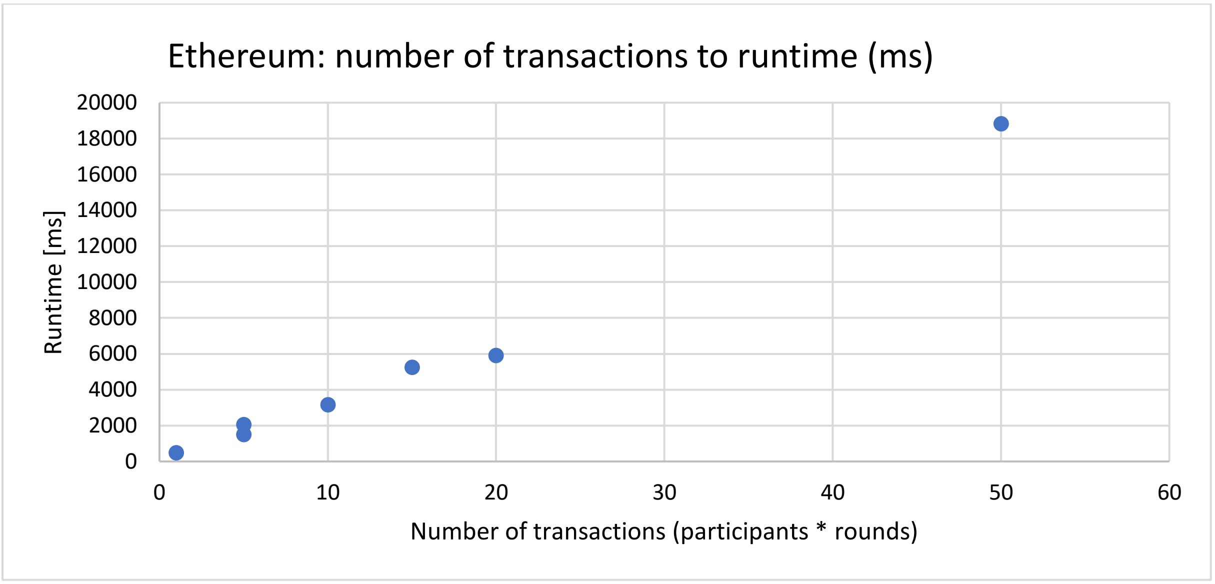


Figure 4: Ethereum time performance using ganache-cli

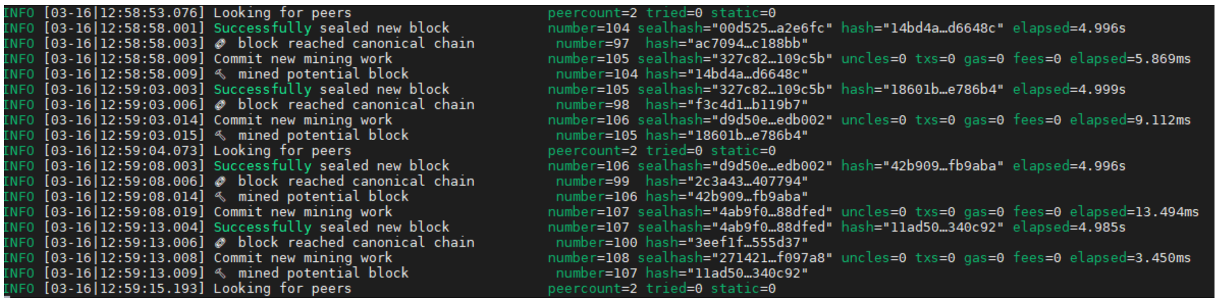
Linear regression showed that the runtime could be calculated as $y = 374{,}29 * x - 364{,}65; R^2 = 0{,}9904$, so the initialization took about 365 ms and each cycle of retrieving the data, hashing it to simulate training and reuploading it to the blockchain took about 374 ms per participant. The strong correlation implies no scaling effects, as was to be expected from the setup. This data does however not reflect the behaviour in a real setup, but it showed that the contract and testing code was valid.

To better simulate a real distributed setting, another setup was deployed, using *go-ethereum/geth*[20], the official Go Implementation of Ethereum. As opposed to *ganache-cli*, which is designed to test smart contracts, this is a real blockchain setup and could be deployed with multiple nodes on physically separate machines, for testing purposes however all nodes were ran on the same machine to capture the aggregated resource usage.

Geth also supports permissioned operation with *Proof of Authority*, where only permitted nodes can sign transactions and therefore mine blocks.

After deploying the network, having one signing node and two additional nodes, remote procedure calls were enabled to node 1 over HTTP to facilitate the testing using the *truffle* framework and the same testing contract as before. As opposed to ganache-cli, the mining process is now being done continuously and not "on demand", as the console output seen in Figure 5 confirms.

[20] Go-ethereum Website: https://geth.ethereum.org/



```
INFO [03-16|12:58:53.076] Looking for peers                      peercount=2 tried=0 static=0
INFO [03-16|12:58:58.001] Successfully sealed new block          number=104 sealhash="00d525…a2e6fc" hash="14bd4a…d6648c" elapsed=4.996s
INFO [03-16|12:58:58.003] 🔗 block reached canonical chain          number=97  hash="ac7094…c188bb"
INFO [03-16|12:58:58.009] Commit new mining work                 number=105 sealhash="327c82…109c5b" uncles=0 txs=0 gas=0 fees=0 elapsed=5.869ms
INFO [03-16|12:58:58.009] 🔨 mined potential block                  number=104 hash="14bd4a…d6648c"
INFO [03-16|12:59:03.003] Successfully sealed new block          number=105 sealhash="327c82…109c5b" hash="18601b…e786b4" elapsed=4.999s
INFO [03-16|12:59:03.006] 🔗 block reached canonical chain          number=98  hash="f3c4d1…b119b7"
INFO [03-16|12:59:03.014] Commit new mining work                 number=106 sealhash="d9d50e…edb002" uncles=0 txs=0 gas=0 fees=0 elapsed=9.112ms
INFO [03-16|12:59:03.015] 🔨 mined potential block                  number=105 hash="18601b…e786b4"
INFO [03-16|12:59:04.073] Looking for peers                      peercount=2 tried=0 static=0
INFO [03-16|12:59:08.003] Successfully sealed new block          number=106 sealhash="d9d50e…edb002" hash="42b909…fb9aba" elapsed=4.996s
INFO [03-16|12:59:08.006] 🔗 block reached canonical chain          number=99  hash="2c3a43…407794"
INFO [03-16|12:59:08.014] 🔨 mined potential block                  number=106 hash="42b909…fb9aba"
INFO [03-16|12:59:08.019] Commit new mining work                 number=107 sealhash="4ab9f0…88dfed" uncles=0 txs=0 gas=0 fees=0 elapsed=13.494ms
INFO [03-16|12:59:13.004] Successfully sealed new block          number=107 sealhash="4ab9f0…88dfed" hash="11ad50…340c92" elapsed=4.985s
INFO [03-16|12:59:13.006] 🔗 block reached canonical chain          number=100 hash="3eef1f…555d37"
INFO [03-16|12:59:13.008] Commit new mining work                 number=108 sealhash="271421…f097a8" uncles=0 txs=0 gas=0 fees=0 elapsed=3.450ms
INFO [03-16|12:59:13.009] 🔨 mined potential block                  number=107 hash="11ad50…340c92"
INFO [03-16|12:59:15.193] Looking for peers                      peercount=2 tried=0 static=0
```

Figure 5: Console output from the singning node indicating the minig process

After setting up the blockchain network with *Proof of Authority*, the same simulation as before was ran to guarantee comparability, while observing the output of the signing node, where the transactions can be seen in real time. As the *Proof of Authority* based block leader selection is computationally more effective than Proof of Work, the mining of blocks not restrained by computing resources, but by the **`BLOCK_PERIOD`** variable as described in the *Clique* specification (Péter Szilágyi, 2017). This means the difference of the timestamps of two following blocks must be greater than five seconds by default, so a new block is mined/signed every five seconds. This can also be observed in the timestamps of the mining/signing process output seen in Figure 5. As expected, the time needed for a given number transactions in the simulation were linear with a factor of ~ 5000 ms (R^2 =0,9977), with only small discrepancies between runs.

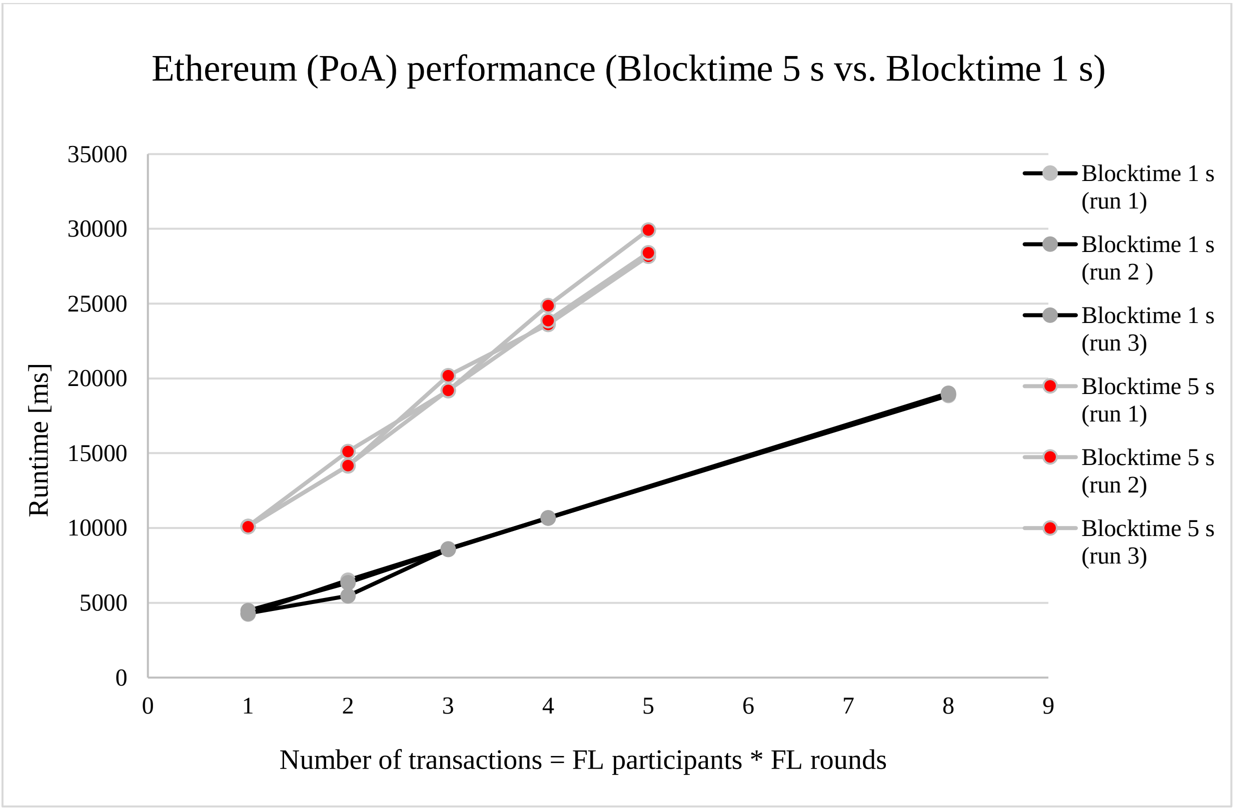


Figure 6: Performance of Ethereum with Proof of Authority (with different settings)

To remove this restriction, the genesis block containing the **`BLOCK_PERIOD`** value was manually changed and the blockchain was ran with **`BLOCK_PERIOD`** set to 1 second. This is not recommended, as it creates instability for a distributed system. The block period is higher, so that all nodes participating can synchronise with the consensus state before a new block can be proposed. However, for this test setup all nodes are hosted on the same network (as all VMs share a network), so for the simulation setup this worked fine.

Although the block time was set to only 1000 milliseconds (so a new block was signed every second), transactions took slightly more than 2000 milliseconds to process, here an apparent performance cap of the Ethereum platform can be observed, still this performance should be good enough to not slow down the FL process.

All in all, the Ethereum experiments were successful, and by using an external storage infrastructure such as IPFS, the test data used for test purposes could be replaced with e.g., the IPFS hash of the model weights. With additional aggregation mechanics implemented in the smart contract and nodes that perform the aggregation process, Ethereum can be used for FL without many adjustments.
With the envisioned FL scenario, especially the permissioned Ethereum *clique* protocol with *Proof of Authorit*y is interesting, so the use of *Clique* for *Proof of Authority* support in an Ethereum-based Federated learning setup is recommended.

6.1.2. Hyperledger Fabric Deployment

The code for the Hyperledger Fabric experiments can also be inspected on GitHub[21]. Hyperledger Fabric was deployed similar virtual machines using the docker images provided by the project. For the testing purposes a test network consisting of two peer nodes and one ordering node was set up. The network was setup with Certificate Authorities (CA) enabled, which enabled Transport Layer Security (TLS) and secure peer management to better simulate a real-world scenario.

As the Truffle framework which was used before does not yet support all Hyperledger Fabric implementations, testing had to be set up manually. The network could be queried using the “peer” Command Line Interface (CLI) however using the CLI for every transaction must be done by hand or using bash scripts. To better automate the testing, the `fabric-network`[22] package was used, which enables NodeJS applications to interact with a running fabric network. For the chaincode (smart contract) a contract from the Hyperledger repository[23], which is written in Go and originally intended for simple asset management, was reused. It also suited the application of simply storing a value on the blockchain for later retrieval. To guarantee comparability, the same simulation pattern as employed for the tests for

[21] Code available on GitHub: https://github.com/vinzenzzinecker98/fabric_tests.
[22] The module is described at: https://www.npmjs.com/package/fabric-network.
[23] The Script`s code is located at: https://github.com/hyperledger/fabric-samples/tree/main/chaincode/sacc.

Ethereum (repeated queries and updates to the ledger state for measure performance) was used. The results were slightly better to Ethereum, an average transaction time of 2116 ms was observed.

6.2. Comparison of test results

Transaction time

As can be seen in Figure 7, the original Ethereum setup had the worst transaction rate, what can obviously be attributed to the default block time being set to five seconds. However, even after changing the block time to one second (what could destabilise the consensus mechanism in production networks due to new blocks not dissipating fast enough in the network), the performance of Ethereum still lags behind Hyperledger Fabric, but not by much.

The results of the ganache simulation cannot be compared to the performance of actual blockchain; however, this result shows how tools like ganache-cli can help in a development environment for fast testing. Having tools like that increase the attractiveness of using Ethereum, where fast prototyping can be less time consuming due to sandbox tools.

All in all, Hyperledger Fabric presented a slightly faster transaction rate (2116 ms per cycle compared to 2271 ms – 7,3% faster) in the experiments, when this experiment was transferred to the real-world FL-application, the faster state updates could theoretically decrease the training delay and improve overall runtime performance. However, because the difference is small, other factors would outweigh this small difference.

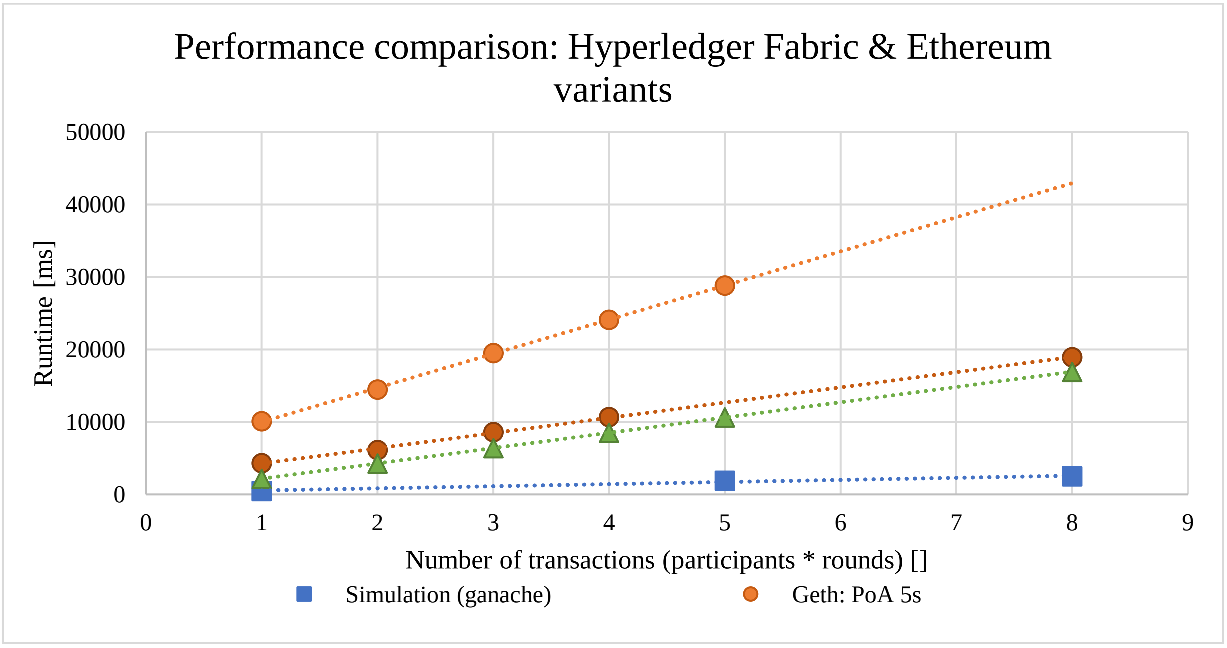


Figure 7: Time performance results of conducted experiments

Security

Regarding the security aspect, Hyperledger Fabric seems to be superior to Ethereum, as certificates issued by defined Certificate Authorities (CA) can be used to verify the identity of nodes. The CA feature is integrated in a modular way, so the blockchain can also be ran without using certificates. For the test runs, the certificate option was enabled.

For Ethereum, no native support for custom CA's was available, so the keys were not signed by an external CA, however the authentication of nodes is done via private key authentication and signatures are used for signing transactions. With go-ethereum, the *netrestrict* flag ensured to only approved nodes from the specified network could participate in the blockchain. Also, the funds available to the nodes had to be specified in the genesis block, which means newly joined nodes do initially not have access to transactions as they have no stake – alternatively the joining of new nodes can be restricted.

Setup difficulty and support materials

For setting up the blockchain, both Ethereum and Fabric have good documentation, while for *go-ethereum,* there are more external materials available, while *Fabric* has less external documentation, but the materials provided by Hyperledger are very useful and beginner-friendly.[24]

For the test setup, Ethereum required much more manual setup (writing genesis block, choosing which ports to use, etc.), while Fabric was ready-to-use with the docker containers provided. However, accessing the Fabric network turned out to be more challenging, as the automated testing suite used for Ethereum (*truffle*) was not available for Fabric.

6.2.1. Resource usage

In order to not only record the transaction rate of the blockchain solutions, but also the effect on the resources of the virtual machine hosting the blockchain, a script[25] to monitor for CPU (Central Processing Unit / Processor) and RAM (Random Access Memory / main memory) usage was used. The script recorded 1) the main memory reported as "used", 2) the "Load Average" as reported by the Linux module "/proc/loadavg", which relates to the number of processes in the run queue (so processes waiting for CPU time) averaged over one minute, as well as 3) the percentual CPU usage of the "Idle" process as reported by the "mpstat" module for more accurate CPU measurements, with the total used CPU time calculated as $total\ CPU = 100\% - percentage\ of\ CPU\ time\ used\ by\ "Idle"$.

In Figure 8, the resource usage of the Ethereum Blockchain can be seen, while Figure 9 depicts the resource values of the Hyperledger Fabric blockchain. To be comparable, Ethereum was run with three nodes, of which one was the signing node, while the Fabric network ran with two peers and one ordering service. In both cases the nodes were all hosted on one virtual machine to capture the whole load. The graphs are organised in the phases according to the status of the blockchain: 1) starting up the blockchain network, 2) idle blockchain running, 3) blockchain processing transactions continuously (simulation active) and 4) shutting down the network.

[24] Hyperledger Fabric Documentation is found at https://hyperledger-fabric.readthedocs.io.
[25] The code is available at: https://github.com/vinzenzzinecker98/fabric_tests/blob/master/memorylog.sh.

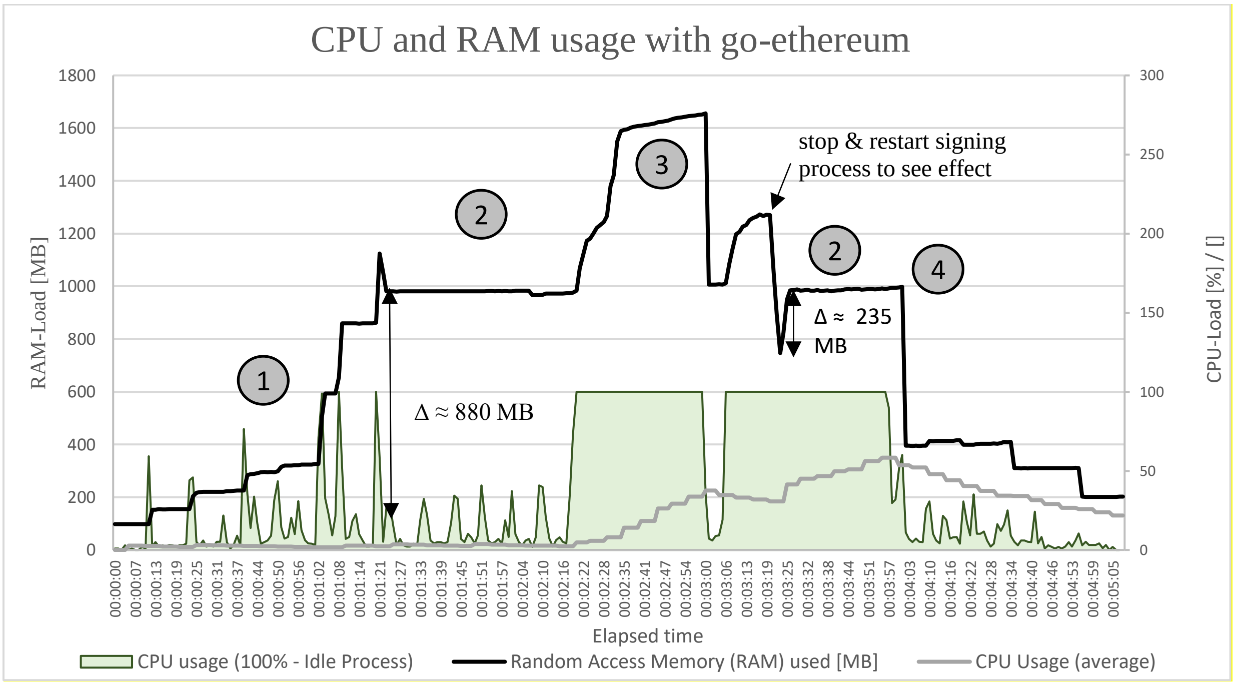


Figure 8: Resource usage while starting up and using an Ethereum blockchain

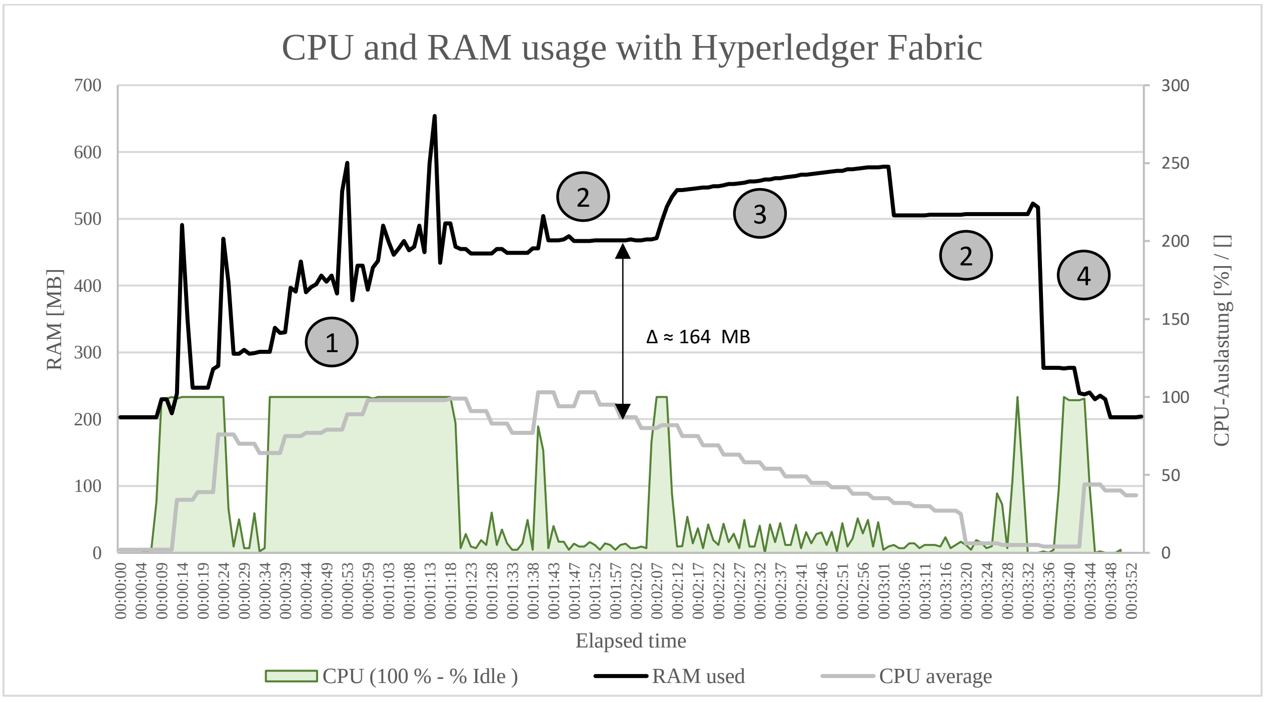


Figure 9: Resource usage while starting up and using the Hyperledger Fabric blockchain

The measurements showed that in an idle state, the Ethereum Blockchain consumed around 880 MB of main memory, of which around 235 MB are used by the block-signing process, while the Fabric network only demanded around 164 MB in total. In the setup phase, the script setting up the fabric network produced spikes indicating high RAM and CPU usage, which could be explaining by the setup of the docker containers running the network and cryptographic processes for setting up the certificate authorities. While the simulation was running the difference in RAM usage reached about 1 GB (550 MB with Fabric, 1600 MB with Ethereum). The most interesting part of this is how for Hyperledger Fabric, even when processing transactions in phase 3), the CPU is not under full load as opposed to when using Ethereum, where the CPU is under full load from the start of the test run on. Summarising, the resource consumption using Hyperledger Fabric was notably lower when compared to the Ethereum blockchain.

6.3. Interpretation and summary of experiments

The experiments conducted had the objective to 1) evaluate the setup difficulty for the tested blockchain solutions and 2) run a simulation of a federated learning process to measure a) performance and b) resource consumption.

As for the setup part, there were no large differences between Hyperledger Fabric and Ethereum. While the Ethereum blockchain was manually setup in terminals, the Fabric network was deployed inside Docker[26] containers using the *test-network* scripts provided by Hyperledger.

For the security aspect, Ethereum locks the transaction with a password, to mitigate typing in a password for each transaction, the nodes were “unlocked” for accessing them via RPC, which would obviously not hold up in a real scenario, however this shows how the read & write access is in addition to the identification of the nodes only protected by password identification with the setup used. For Hyperledger Fabric, Certificate Authorities can be enabled to ensure only peers with a valid certificate can participate in the ledger. This had also not to be mitigated for testing purposes, instead the correct certificate files have been used for authorising transactions.

For the simulation, it showed that setting up a private network with *Proof of Authority* in Ethereum was doable using the *Clique* (Péter Szilágyi, 2017) protocol. However, this comes by default with a block time of five seconds, which was a bottleneck for the transaction processing. For faster transactions, this restriction was manually removed, although this is not recommended in a real-world application for stability reasons. Even after this adaption, the Fabric network provided a slightly higher transaction speed (and therefore overall better simulation performance, as can be seen in Figure 7).

While the tests ran, the memory and CPU usage were monitored which showed how the Fabric network used up way less resources in the process.

All in all, based on the experiments conducted, Hyperledger Fabric seems to be better suited for the task at hand than Ethereum.

[26] Docker provides containerisation software. Website: https://www.docker.com/

7. Discussion

7.1. Principal findings

The aim of this thesis was to present how a blockchain solution can be integrated into a federated learning setting in healthcare, what assumptions are made on the blockchain layer and what current blockchain solutions best fit to the scenario. The systematic literature review in the first part of this work examined what different ways of implementation there are for blockchain-assisted FL (chapter 2.3) and showed how especially in the last years, many research papers have been produced that propose the use of a blockchain layer to facilitate federated learning in a more secure and less centralised way. However, papers differ highly in the way of implementation and the blockchain ecosystem used. A quantitative analysis showed how some works propose the creation of a designated blockchain infrastructure, while others use existing blockchain frameworks and protocols, with most papers proposing Ethereum as the preferred framework to use (see Figure 3). The latter is often realised in conjunction with external distributed storage solutions, to minimise the amount of data stored on the blockchain.
The requirements for the blockchain layer of such a system, depending on the implementation type, were highlighted in Chapter 4. The analysis of the requirements for the blockchain layer in a healthcare federated learning scenario pointed towards a private permissioned blockchain with focus on performance, reliability and security.
Looking at the different blockchain solutions employed in the identified literature in chapter 5.1, the most promising candidates, namely Ethereum and Hyperledger Fabric, were then further analysed and examined, using practical experiment setups conducted on virtual machines.
The simulation suggested slightly superior time performance with much lower resource consumption for the simulation using Hyperledger Fabric. Furthermore, specific security features, such as certificates, are already integrated into Fabric.
Considering the simulation test results, the requirements defined for the scenario as well as my personal experience while setting up the testing for both Ethereum and Hyperledger Fabric, it can be concluded that, while both tested solutions can perform the task at hand perfectly fine, Hyperledger Fabric can be better suited to facilitate federated learning than Ethereum. This is due to enhanced security, better modularity and superior resource efficiency. This may be explained by the fact that Fabric was designed for a private permissioned scenario from the beginning, while Ethereum was adapted and originally designed for public deployment.

7.2. Implications for Research and Practice

The literature review on the topic and the categorisation of approaches can help future researchers as guidance for getting a quick overview over the topic. Furthermore, future works that intend to set up a federated learning system with a blockchain-based infrastructure can use this work to inform themselves about different modes of integrating the blockchain and use the results of this thesis to make a more educated decision with regards to which blockchain framework to use. As the state of blockchain technology my be changing rapidly as time passes, the developed requirements on the blockchain can still help with the identification of a suited blockchain solution even as new candidates are emerging.

7.3. Limitations and Future Research

One limitation of this work is the strong actuality of the topic, as the literature review analysed the current state of research and there will be many more papers on this topic in the future. Also, software solutions used in the experiments can be updated and optimised, so that future evaluations could lead to different results. Future works could pick up the results and compare them to novel solutions and proposals. Examining the fit of more upcoming blockchain solutions can also be beneficial to research.
Furthermore, the practical experiments conducted have a quite narrow scope, as basically only transaction speed was actively monitored for, and the setup with VM's on a single network are by no means a perfect model of a real-world scenario. Future works could create more realistic testing conditions, such as a truly distributed setting and evaluate a real federated learning process instead of a simulation.
The inspected federated learning scenario was also kept very broad, specialising on a specific learning task or data type could allow more fine-grained analysis of the requirements on the blockchain layer.

8. Conclusion

This thesis examines how blockchain solutions can be used to enhance federated learning in a healthcare scenario. A systematic literature review on the use of blockchain systems with federated learning was conducted and different modes of integrating the blockchain layer into a federated learning system were identified: Using the blockchain layer to propagate the model data; Using the blockchain in composition with an external storage infrastructure; Performing centrally coordinated federated learning with additional blockchain infrastructure to record actions or participants' reputation.
After motivating the use of blockchain in this setup, the main requirements on the blockchain layer were defined and the fit of existing blockchain solutions to these requirements was discussed. The best fitting candidates, namely Ethereum and Hyperledger Fabric were then further analysed. Practical experiments showed marginally better performance for Hyperledger Fabric with lower resource consumption and higher security, so considering all researched aspects, Hyperledger Fabric is recommended over Ethereum for the integration into a federated learning system into a healthcare setting.

Appendix

A. Appendix A

Table 2: Citations for the data presented in Figure 2

Block-chain Mode	**Designated blockchain (permissioned)**	**Smart contracts on existing blockchain solutions**	**Hybrid solutions**
Model on chain	15 papers: (Bao et al., 2019; Chai et al., 2020; Doku & Rawat, 2020; Kim et al., 2020; Kuo & Ohno-Machado, 2018; Lu, Huang, Zhang, et al., 2020b, 2020c; Ma et al., 2020; Majeed & Hong, 2019; Pokhrel & Choi, 2020; Y. Qu, Gao, et al., 2020; Y. Qu, Pokhrel, et al., 2020; Shayan et al., 2018; Wang et al., 2020; Weng et al., 2019)	8 papers: (Cui et al., 2020; Yi Liu et al., 2020; Ramanan & Nakayama, 2019; Rathore et al., 2019; Schmid et al., 2020; Toyoda & Zhang, 2019; ur Rehman et al., 2020; Y. Wu et al., 2020)	4 papers: (Desai et al., 2020; Fan et al., 2020; Lu, Huang, Zhang, et al., 2020a; Lugan et al., 2019)
Key to Model	1 paper: (Zhao et al., 2020)	11 papers: (Arachchige et al., 2020; Awan et al., 2019; Cai et al., 2020; Hu et al., 2020; Kumar et al., 2020; L. Liu et al.; Martinez et al., 2019; Mugunthan et al., 2020; Passerat-Palmbach et al., 2019; Ratadiya et al., 2020; X. Wu et al., 2020)	
Audit trail / Reputa-tion Block-chain	2 papers: (Kang, Xiong, Niyato, Xie, & Zhang, 2019; Yuan Liu et al., 2020)	4 papers: (Kang et al., 2020, 2019; Yin et al., 2020; Zhang et al., 2020)	
Other	2 papers: (Lu, Huang, Dai, et al., 2020; X. Qu et al., 2019)	2 papers: (Baranwal Somy et al., 2019; Ouyang et al., 2020)	

References

Al-Rubaie, M., & Chang, J. M. (2019). Privacy-Preserving Machine Learning: Threats and Solutions. *IEEE Security & Privacy*, *17*(2), 49–58. https://doi.org/10.1109/MSEC.2018.2888775

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., Caro, A. de, Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., . . . Yellick, J. (2018). Hyperledger fabric. In R. Oliveira, P. Felber, & Y. C. Hu (Eds.), *ACM Conferences, Proceedings of the Thirteenth EuroSys Conference* (pp. 1–15). ACM. https://doi.org/10.1145/3190508.3190538

Ann Cavoukian. (2011). *Privacy by Design* [Information & Privacy Commissioner].

Arachchige, P. C. M., Bertok, P., Khalil, I., Liu, D., Camtepe, S., & Atiquzzaman, M. (2020). A Trustworthy Privacy Preserving Framework for Machine Learning in Industrial IoT Systems. *IEEE Transactions on Industrial Informatics*, *16*(9), 6092–6102. https://doi.org/10.1109/TII.2020.2974555

Awan, S., Li, F., Luo, B., & Liu, M. (2019). Poster: A Reliable and Accountable Privacy-Preserving Federated Learning Framework using the Blockchain. In L. Cavallaro, J. Kinder, X. Wang, & J. Katz (Eds.), *CCS'19: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, United Kingdom, November 11-15, 2019* (pp. 2561–2563). Association for Computing Machinery. https://doi.org/10.1145/3319535.3363256

Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., & Shmatikov, V. (2018, July 2). *How To Backdoor Federated Learning*. https://arxiv.org/pdf/1807.00459

Bao, X., Su, C., Xiong, Y., Huang, W., & Hu, Y [Yifei] (2019, August). FLChain: A Blockchain for Auditable Federated Learning with Trust and Incentive. In *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)* (pp. 151–159). IEEE. https://doi.org/10.1109/BIGCOM.2019.00030

Baranwal Somy, N., Kannan, K., Arya, V., Hans, S., Singh, A., Lohia, P., & Mehta, S. (2019, July). Ownership Preserving AI Market Places Using Blockchain. In *2019 IEEE International Conference on Blockchain (Blockchain)* (pp. 156–165). IEEE. https://doi.org/10.1109/Blockchain.2019.00029

Beam, A. L., & Kohane, I. S. (2018). Big Data and Machine Learning in Health Care. *JAMA*, *319*(13), 1317–1318. https://doi.org/10.1001/jama.2017.18391

Benet, J. (2014, July 14). *IPFS - Content Addressed, Versioned, P2P File System*. https://arxiv.org/pdf/1407.3561

block.one. (2018). *EOS.IO Technical White Paper v2*. https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md

Brisimi, T. S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I. C., & Shi, W. (2018). Federated learning of predictive models from federated Electronic Health Records. *International Journal of Medical Informatics*, *112*, 59–67. https://doi.org/10.1016/j.ijmedinf.2018.01.007

Brown, R. G., Carlyle, J., Grigg, I., & Hearn, M. (2016). *Corda: An Introduction.* https://doi.org/10.13140/RG.2.2.30487.37284

Cai, H., Rueckert, D., & Passerat-Palmbach, J. (2020). 2CP: Decentralized Protocols to Transparently Evaluate Contributivity in Blockchain Federated Learning Environments. *IEEE 2nd International Workshop on Advances in Artificial Intelligence for Blockchain (AIChain*. https://arxiv.org/pdf/2011.07516

Chai, H., Leng, S., Chen, Y., & Zhang, K. (2020). A Hierarchical Blockchain-Enabled Federated Learning Algorithm for Knowledge Sharing in Internet of Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 1–12. https://doi.org/10.1109/TITS.2020.3002712

Chang, K., Balachandar, N., Lam, C., Yi, D., Brown, J., Beers, A., Rosen, B., Rubin, D. L., & Kalpathy-Cramer, J. (2018). Distributed deep learning networks among institutions for medical imaging.

Journal of the American Medical Informatics Association : JAMIA, *25*(8), 945–954. https://doi.org/10.1093/jamia/ocy017

Cui, L., Su, X., Ming, Z., Chen, Z., Yang, S., Zhou, Y., & Xiao, W. (2020). CREAT: Blockchain-assisted Compression Algorithm of Federated Learning for Content Caching in Edge Computing. *IEEE Internet of Things Journal*, 1. https://doi.org/10.1109/JIOT.2020.3014370

Desai, H. B., Ozdayi, M. S., & Kantarcioglu, M. (2020, October 15). *BlockFLA: Accountable Federated Learning via Hybrid Blockchain Architecture*. https://arxiv.org/pdf/2010.07427

Doku, R., & Rawat, D. B. (2020, May). IFLBC: On the Edge Intelligence Using Federated Learning Blockchain Network. In *2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)* (pp. 221–226). IEEE. https://doi.org/10.1109/BigDataSecurity-HPSC-IDS49724.2020.00047

DR. GAVIN WOOD. (2021). *ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER: PETERSBURG VERSION 41c1837 – 2021-02-14*. https://ethereum.github.io/yellowpaper/paper.pdf

Dwork, C. (2006). Differential Privacy. In *Lecture Notes in Computer Science, 33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006)* (33rd ed., pp. 1–12). Springer Verlag. https://www.microsoft.com/en-us/research/publication/differential-privacy/

Fan, S., Zhang, H., Zeng, Y., & Cai, W. (2020). Hybrid Blockchain-Based Resource Trading System for Federated Learning in Edge Computing. *IEEE Internet of Things Journal*, 1. https://doi.org/10.1109/JIOT.2020.3028101

Fredrikson, M., Jha, S., & Ristenpart, T. (2015). Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In I. Ray, N. Li, & C. Kruegel (Eds.), *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (pp. 1322–1333). ACM. https://doi.org/10.1145/2810103.2813677

Fung, C., Yoon, C. J. M., & Beschastnikh, I. (2018, August 14). *Mitigating Sybils in Federated Learning Poisoning*. http://arxiv.org/pdf/1808.04866v5

Hearn, M., & Brown, R. G. (2019). *Corda: A distributed ledger*. https://www.r3.com/wp-content/uploads/2019/08/corda-technical-whitepaper-August-29-2019.pdf

Hu, Y [Yifan], Xia, W., Xiao, J., & Wu, C. (2020, October 21). *GFL: A Decentralized Federated Learning Framework Based On Blockchain*. https://arxiv.org/pdf/2010.10996

hyperledger.org. (2020). *Hyperledger Fabric White Paper*. https://www.hyperledger.org/wp-content/uploads/2020/03/hyperledger_fabric_whitepaper.pdf

K. Wüst, & A. Gervais (2018). Do you Need a Blockchain? In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., . . . Zhao, S. (2019, December 10). *Advances and Open Problems in Federated Learning*. http://arxiv.org/pdf/1912.04977v1

Kaissis, G. A., Makowski, M. R., Rückert, D., & Braren, R. F. (2020). Secure, privacy-preserving and federated machine learning in medical imaging. *Nature Machine Intelligence*, *2*(6), 305–311. https://doi.org/10.1038/s42256-020-0186-1

Kang, J., Xiong, Z., Niyato, D., Xie, S., & Zhang, J [Junshan] (2019). Incentive Mechanism for Reliable Federated Learning: A Joint Optimization Approach to Combining Reputation and Contract Theory. *IEEE Internet of Things Journal*, *6*(6), 10700–10714. https://doi.org/10.1109/JIOT.2019.2940820

Kang, J., Xiong, Z., Niyato, D., Zou, Y., Zhang, Y [Yang], & Guizani, M. (2019, October 14). *Reliable Federated Learning for Mobile Networks*. https://arxiv.org/pdf/1910.06837

Kang, J., Xiong, Z., Niyato, D., Zou, Y., Zhang, Y [Yang], & Guizani, M. (2020). Reliable Federated Learning for Mobile Networks. *IEEE Wireless Communications*, *27*(2), 72–80. https://doi.org/10.1109/MWC.001.1900119

Kannengießer, N., Lins, S., Dehling, T., & Sunyaev, A. (2020). Trade-offs between Distributed Ledger Technology Characteristics. *ACM Computing Surveys, 53*(2), 1–37. https://doi.org/10.1145/3379463 (ACM Computing Surveys, 53(2), 1-37).

Kim, H., Park, J., Bennis, M., & Kim, S.-L. (2018, August 12). *Blockchained On-Device Federated Learning*. https://arxiv.org/pdf/1808.03949

Kim, H., Park, J., Bennis, M., & Kim, S.-L. (2020). Blockchained On-Device Federated Learning. *IEEE Communications Letters*, *24*(6), 1279–1283. https://doi.org/10.1109/LCOMM.2019.2921755

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, *114*(13), 3521–3526. https://doi.org/10.1073/pnas.1611835114

Kumar, S., Dutta, S., Chatturvedi, S., & Bhatia, M. P. (2020, September). Strategies for Enhancing Training and Privacy in Blockchain Enabled Federated Learning. In *2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM)* (pp. 333–340). IEEE. https://doi.org/10.1109/BigMM50055.2020.00058

Kuo, T.-T., & Ohno-Machado, L. (2018, February 6). *ModelChain: Decentralized Privacy-Preserving Healthcare Predictive Modeling Framework on Private Blockchain Networks*. https://arxiv.org/pdf/1802.01746

Li, Y., Chen, C., Liu, N., Huang, H., Zheng, Z., & Yan, Q. (2020, April 2). *A Blockchain-based Decentralized Federated Learning Framework with Committee Consensus*. http://arxiv.org/pdf/2004.00773v1

Liu, L., Hu, Y [Yifan], Yu, J., Zhang, F., Huang, G., Xiao, J., & Wu, C. Training Encrypted Models with Privacy-preserved Data on Blockchain. In *ICVISP 2019 2019* (pp. 1–6). https://doi.org/10.1145/3387168.3387211

Liu, Y [Yi], Peng, J., Kang, J., Iliyasu, A. M., Niyato, D., & El-Latif, A. A. A. (2020). A Secure Federated Learning Framework for 5G Networks. *IEEE Wireless Communications*, *27*(4), 24–31. https://doi.org/10.1109/MWC.01.1900525

Liu, Y [Yuan], Sun, S., Ai, Z., Zhang, S., Liu, Z., & Yu, H. (2020, February 26). *FedCoin: A Peer-to-Peer Payment System for Federated Learning*. https://arxiv.org/pdf/2002.11711

Lu, Y., Huang, X., Dai, Y., Maharjan, S., & Zhang, Y [Yan] (2020). Blockchain and Federated Learning for Privacy-Preserved Data Sharing in Industrial IoT. *IEEE Transactions on Industrial Informatics*, *16*(6), 4177–4186. https://doi.org/10.1109/TII.2019.2942190

Lu, Y., Huang, X., Zhang, K., Maharjan, S., & Zhang, Y [Yan] (2020a). Blockchain Empowered Asynchronous Federated Learning for Secure Data Sharing in Internet of Vehicles. *IEEE Transactions on Vehicular Technology*, *69*(4), 4298–4311. https://doi.org/10.1109/TVT.2020.2973651

Lu, Y., Huang, X., Zhang, K., Maharjan, S., & Zhang, Y [Yan] (2020b). Communication-efficient Federated Learning and Permissioned Blockchain for Digital Twin Edge Networks. *IEEE Internet of Things Journal*, 1. https://doi.org/10.1109/JIOT.2020.3015772

Lu, Y., Huang, X., Zhang, K., Maharjan, S., & Zhang, Y [Yan] (2020c). Low-latency Federated Learning and Blockchain for Edge Association in Digital Twin empowered 6G Networks. *IEEE Transactions on Industrial Informatics*, 1. https://doi.org/10.1109/TII.2020.3017668

Lugan, S., Desbordes, P., Brion, E., Ramos Tormo, L. X., Legay, A., & Macq, B. (2019). Secure Architectures Implementing Trusted Coalitions for Blockchained Distributed Learning (TCLearn). *IEEE Access*, *7*, 181789–181799. https://doi.org/10.1109/ACCESS.2019.2959220

Ma, C., Li, J., Ding, M., Shi, L., Wang, T., Han, Z., & Poor, H. V. (2020, September 20). *When Federated Learning Meets Blockchain: A New Distributed Learning Paradigm*. https://arxiv.org/pdf/2009.09338

Majeed, U., & Hong, C. S. (2019, September). FLchain: Federated Learning via MEC-enabled Blockchain Network. In *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)* (pp. 1–4). IEEE. https://doi.org/10.23919/APNOMS.2019.8892848

Martinez, I., Francis, S., & Hafid, A. S. (2019, October). Record and Reward Federated Learning Contributions with Blockchain. In *2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)* (pp. 50–57). IEEE. https://doi.org/10.1109/CyberC.2019.00018

Mathur, P. (2019). Overview of Machine Learning in Healthcare. In Mathur, P. Mathur, & S. John (Eds.), *Machine Learning Applications Using Python* (1st ed., pp. 1–11). Apress. https://doi.org/10.1007/978-1-4842-3787-8_1

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., & Arcas, B. A. y. (2016). Communication-Efficient Learning of Deep Networks from Decentralized Data. http://arxiv.org/pdf/1602.05629v3

Muguthan, V., Rahman, R., & Kagal, L. (2020, July 8). *BlockFLow: An Accountable and Privacy-Preserving Solution for Federated Learning*. https://arxiv.org/pdf/2007.03856

Ouyang, L., Yuan, Y., & Wang, F.-Y. (2020). Learning Markets: An AI Collaboration Framework Based on Blockchain and Smart Contracts. *IEEE Internet of Things Journal*, 1. https://doi.org/10.1109/JIOT.2020.3032706

Pandl, K. D., Thiebes, S., Schmidt-Kraepelin, M., & Sunyaev, A. (2020). On the Convergence of Artificial Intelligence and Distributed Ledger Technology: A Scoping Review and Future Research Agenda. *IEEE Access*, *8*, 57075–57095. https://doi.org/10.1109/ACCESS.2020.2981447

Passerat-Palmbach, J., Farnan, T., Miller, R., Gross, M. S., Flannery, H. L., & Gleim, B. (2019, October 12). *A blockchain-orchestrated Federated Learning architecture for healthcare consortia*. http://arxiv.org/pdf/1910.12603v1

Péter Szilágyi. (2017). *EIP-225: Clique proof-of-authority consensus protocol*. https://eips.ethereum.org/EIPS/eip-225

Pokhrel, S. R., & Choi, J. (2020). Federated Learning With Blockchain for Autonomous Vehicles: Analysis and Design Challenges. *IEEE Transactions on Communications*, *68*(8), 4734–4746. https://doi.org/10.1109/TCOMM.2020.2990686

Qu, X., Wang, S., Hu, Q., & Cheng, X. (2019, December 26). *Proof of Federated Learning: A Novel Energy-recycling Consensus Algorithm*. https://arxiv.org/pdf/1912.11745

Qu, Y., Gao, L., Luan, T. H., Xiang, Y., Yu, S., Li, B., & Zheng, G. (2020). Decentralized Privacy Using Blockchain-Enabled Federated Learning in Fog Computing. *IEEE Internet of Things Journal*, *7*(6), 5171–5183. https://doi.org/10.1109/JIOT.2020.2977383

Qu, Y., Pokhrel, S. R., Garg, S., Gao, L., & Xiang, Y. (2020). A Blockchained Federated Learning Framework for Cognitive Computing in Industry 4.0 Networks. *IEEE Transactions on Industrial Informatics*, 1. https://doi.org/10.1109/TII.2020.3007817

Raisaro, J. L., Gwangbae, C., Pradervand, S., Colsenet, R., Jacquemont, N., Rosat, N., Mooser, V., & Hubaux, J.-P. (2018). Protecting Privacy and Security of Genomic Data in i2b2 with Homomorphic Encryption and Differential Privacy. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *15*(5), 1413–1426. https://doi.org/10.1109/TCBB.2018.2854782

Ramanan, P., & Nakayama, K. (2019, September 16). *BAFFLE : Blockchain Based Aggregator Free Federated Learning*. http://arxiv.org/pdf/1909.07452v2

Ratadiya, P., Asawa, K., & Nikhal, O. (2020, November 22). *A decentralized aggregation mechanism for training deep learning models using smart contract system for bank loan prediction*. https://arxiv.org/pdf/2011.10981

Rathore, S., Pan, Y., & Park, J. H. (2019). BlockDeepNet: A Blockchain-Based Secure Deep Learning for IoT Network. *Sustainability, 11*(14), 3974. https://doi.org/10.3390/su11143974

Saleh, F. (2018). *Blockchain Without Waste: Proof-of-Stake.* https://doi.org/10.2139/ssrn.3183935

Satoshi Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf

Schäffer, M., Di Angelo, M., & Salzer, G. (2019). Performance and Scalability of Private Ethereum Blockchains. In C. Di Ciccio, R. Gabryelczyk, L. García-Bañuelos, T. Hernaus, R. Hull, M. Indihar Štemberger, A. Kő, & M. Staples (Eds.), *Business Process Management: Blockchain and Central and Eastern Europe Forum* (pp. 103–118). Springer International Publishing.

Schmid, R., Pfitzner, B., Beilharz, J., Arnrich, B., & Polze, A. (2020, May). Tangle Ledger for Decentralized Learning. In *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (pp. 852–859). IEEE. https://doi.org/10.1109/IPDPSW50202.2020.00144

Shayan, M., Fung, C., Yoon, C. J. M., & Beschastnikh, I. (2018, November 24). *Biscotti: A Ledger for Private and Secure Peer-to-Peer Machine Learning*. https://arxiv.org/pdf/1811.09904

Sheller, M. J., Reina, G. A., Edwards, B., Martin, J., & Bakas, S. (2019). Multi-Institutional Deep Learning Modeling Without Sharing Patient Data: A Feasibility Study on Brain Tumor Segmentation. *Brainlesion : Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries. BrainLes (Workshop), 11383,* 92–104. https://doi.org/10.1007/978-3-030-11723-8_9

Short, A. R., Leligou, H. C., Papoutsidakis, M., & Theocharis, E. (2020, July). Using Blockchain Technologies to Improve Security in Federated Learning Systems. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)* (pp. 1183–1188). IEEE. https://doi.org/10.1109/COMPSAC48688.2020.00-96

Snyder, H. (2019). Literature review as a research methodology: An overview and guidelines. *Journal of Business Research*, *104*, 333–339. https://doi.org/10.1016/j.jbusres.2019.07.039

Sunyaev, A. (2020). *Internet Computing: Principles of Distributed Systems and Emerging Internet-Based Technologies* (1. ed. 2020). Springer International Publishing; Imprint: Springer.

Toyoda, K., & Zhang, A. N. (2019, December). Mechanism Design for An Incentive-aware Blockchain-enabled Federated Learning Platform. In *2019 IEEE International Conference on Big Data (Big Data)* (pp. 395–403). IEEE. https://doi.org/10.1109/BigData47090.2019.9006344

ur Rehman, M. H., Salah, K., Damiani, E., & Svetinovic, D. (2020, July). Towards Blockchain-Based Reputation-Aware Federated Learning. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (pp. 183–188). IEEE. https://doi.org/10.1109/INFOCOMWKSHPS50562.2020.9163027

Vitalik Buterin. (2013). *Ethereum Whitepaper*. https://ethereum.org/en/whitepaper/

Vries, A. de (2018). Bitcoin's Growing Energy Problem. *Joule, 2*(5), 801–805. https://doi.org/10.1016/j.joule.2018.04.016

Wang, Y., Su, Z., Zhang, N., & Benslimane, A. (2020). Learning in the Air: Secure Federated Learning for UAV-Assisted Crowdsensing. *IEEE Transactions on Network Science and Engineering,* 1. https://doi.org/10.1109/TNSE.2020.3014385

Weng, J [Jiasi], Weng, J [Jian], Zhang, J [Jilian], Li, M., Zhang, Y [Yue], & Luo, W. (2019). DeepChain: Auditable and Privacy-Preserving Deep Learning with Blockchain-based Incentive. *IEEE Transactions on Dependable and Secure Computing,* 1. https://doi.org/10.1109/TDSC.2019.2952332

Wu, X., Wang, Z., Zhao, J [Jian], Zhang, Y [Yan], & Wu, Y [Yu] (2020, June). FedBC: Blockchain-based Decentralized Federated Learning. In *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)* (pp. 217–221). IEEE. https://doi.org/10.1109/ICAICA50127.2020.9182705

Wu, Y [Yifu], Mendis, G. J., & Wei, J. (2020). DDLPF: A Practical Decentralized Deep Learning Paradigm for Internet of Things Applications. *IEEE Internet of Things Journal*, 1. https://doi.org/10.1109/JIOT.2020.3033482

Xiao, Y., Zhang, N., Lou, W., & Hou, Y. T. (2020). A Survey of Distributed Consensus Protocols for Blockchain Networks. *IEEE Communications Surveys & Tutorials*, *22*(2), 1432–1465. https://doi.org/10.1109/COMST.2020.2969706

Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D., & Beaufays, F. (2018, December 7). *Applied Federated Learning: Improving Google Keyboard Query Suggestions*. http://arxiv.org/pdf/1812.02903v1

Yin, B., Yin, H., Wu, Y [Yulei], & Jiang, Z. (2020). FDC: A Secure Federated Deep Learning Mechanism for Data Collaborations in the Internet of Things. *IEEE Internet of Things Journal*, *7*(7), 6348–6359. https://doi.org/10.1109/JIOT.2020.2966778

Zhang, W., Lu, Q., Yu, Q., Li, Z [Zhaotong], Liu, Y [Yue], Lo, S. K., Chen, S., Xu, X., & Zhu, L. (2020). Blockchain-based Federated Learning for Device Failure Detection in Industrial IoT. *IEEE Internet of Things Journal*, 1. https://doi.org/10.1109/JIOT.2020.3032544

Zhao, Y., Zhao, J [Jun], Jiang, L., Tan, R., Niyato, D., Li, Z [Zengxiang], Lyu, L., & Liu, Y [Yingbo] (2020). Privacy-Preserving Blockchain-Based Federated Learning for IoT Devices. *IEEE Internet of Things Journal*, 1. https://doi.org/10.1109/JIOT.2020.3017377

Declaration about the Thesis

Ich versichere wahrheitsgemäß, die Arbeit selbstständig verfasst, alle benutzten Quellen und Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

[illegible]

Karlsruhe, den 15. April 2021 Vinzenz Zinecker