People's -Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University Echahid Cheikh Larbi Tebessi- Tebessa

Faculty of Exact Sciences and Natural and Life Sciences
Department of Computer Science
Laboratory of Mathematics, Informatics and Systems (LAMIS)

Thesis

Presented and Publicly Defended for Obtaining the Diploma of Doctorate in Third Cycle

By:
Rofaida Khemaissia

Field: Computer Science
Specialty: Networks and Multimedia

Title

Using Blockchain to Protect Personal Data

Defended on: **03/06/2025**

In front of the jury:

Pr. Hakim Bendjenna	Echahid Cheikh Larbi Tebessi University	Chairman
Pr. Makhlouf Derdour	Larbi Ben M'Hidi University	Supervisor
Pr. Toufik Maarouk	Abbes Laghrour University	Examiner
Dr. Abelmajid Benmachiche	Chadli-Bendjedid University	Examiner
Dr. Abdellatif Sahraoui	Echahid Cheikh Larbi Tebessi University	Examiner
Dr. Mohamed Gasmi	Echahid Cheikh Larbi Tebessi University	Examiner

Academic year: 2024/2025

Dedication

This work is dedicated to all my small and large family members, first of all, to the soul of my dear father Said, to my husband Ala who believes and supports me, to my source of inspiration my Mum Fatiha and her encouragement all along, to my sweet children for their patience with me.

Abstract

Data sharing has become a sore point in information technology for decades, considering the technological advancements that have contributed to increased data generation and collection, by utilizing numerous applications and services. It has become a double-edged sword since they have taken advantage of the massive amount of individual personal data for their own profit. Consequently, several concerns have arisen related to centralized architectures security and privacy infringements.

This thesis showcases efforts made for data sharing and privacy based on the emerging technology blockchain, which backs system transparency, data confidentiality, integrity, and security. This thesis aims to make three novel contributions to preserve data-sharing privacy. In a service-oriented architecture we propose a privacy-preserving platform called "SDGChain" by employing a service-dependency graph-based permissioned blockchain to control a set of service interactions and data exchanged by granting data owner sovereignty through access permission rules. Blockchain smart contracts are deployed to allow control over confidential data exchanged between services and present a global view of system interactions. SDGchain demonstrates the robustness of the prototype using blockchain as a trusted third party, in addition to the effect of adopting off-chain storage on system scalability from the results since keeping only light data.

The research is extended to cover composite services privacy where PrsChain, a privacy preservation framework is proposed for service composition using a permissioned blockchain in the context of service-oriented architecture. The scheme is designed to solve the issue of sensitive data sharing disclosure among service providers, eliminates trust in third parties, and fosters a trustworthy blockchain for generating plan composition and managing the execution process, where the substantial features that blockchain is leveraged upon are authentication, tamper-proof, integrity, immutability, and trustless environment. We present the framework architecture and discuss different aspects of evaluating and implementing smart contracts during composition execution. In addition, the intermediate results are stored in the IPFS, which is encrypted to create a robust data access mechanism only for legal participants from the blockchain system.

The thesis develops a privacy-preserving framework for federated learning combined with blockchain technology. The proposed design, named FLBCshard, performs decentralized federated learning tasks securely using IPFS and non-fungible token-based data sharing as proof of ownership. The prototype trades off between blockchain scalability and privacy by utilizing a dynamic sharding technique and, therefore, a hierarchical architecture to alleviate overhead communication and double check reliability methods are proposed to mitigate privacy and security threats. The main objective

of the design is to maintain participant and global model data privacy with high accuracy and to prevent poisoning attacks and even model ownership theft.

Keywords: Blockchain, privacy-preserving, Hyperledger, Permissioned Blockchain, service dependency graph, service composition, Federated learning, Sharding, IPFS, NFT.

Résumé

Le partage de données est devenu un sujet sensible dans les technologies de l'information depuis des décennies, compte tenu des avancées technologiques qui ont contribué à accroître la production et la collecte de données grâce à de nombreux services et applications. Ce partage est devenu une arme à double tranchant, car ces derniers ont exploité l'énorme quantité de données personnelles à leur profit. Par conséquent, plusieurs préoccupations ont surgi concernant les architectures centralisées, la sécurité et les atteintes à la vie privée. Cette thèse présente les efforts déployés pour le partage et la confidentialité des données grâce à la technologie émergente blockchain, qui garantit la transparence des systèmes, la confidentialité, l'intégrité et la sécurité des données. Elle vise à apporter trois contributions pour préserver la confidentialité du partage des données. Dans une architecture orientée services, nous proposons une plateforme de protection de la confidentialité appelée « SDGChain ». Elle utilise une blockchain à autorisations basée sur un graphe de dépendance des services pour contrôler un ensemble d'interactions entre les services et les données échangées, en accordant la souveraineté au propriétaire des données via des règles d'autorisation d'accès. Des contrats intelligents blockchain sont déployés pour permettre le contrôle des données confidentielles échangées entre les services et offrir une vue globale des interactions du système. SDGchain démontre la robustesse du prototype utilisant la blockchain comme tiers de confiance, ainsi que l'impact de l'adoption du stockage hors chaîne sur l'évolutivité du système, d'après les résultats obtenus, compte tenu de la faible quantité de données conservées. La recherche est étendue à la confidentialité des services composites. PrsChain, une nouvelle approche de préservation de la confidentialité, est proposé pour la composition de services utilisant une blockchain dans le contexte d'une architecture orientée services. Ce schéma est conçu pour résoudre le problème de divulgation des données sensibles partagées entre les fournisseurs de services, éliminer la confiance envers les tiers et favoriser une blockchain fiable pour la génération de la composition des plans et la gestion du processus d'exécution. Les principales fonctionnalités de la blockchain sont l'authentification, l'inviolabilité, l'intégrité, l'immuabilité et l'environnement sans confiance. Nous présentons l'architecture du PrsChain et abordons les différents aspects de l'évaluation et de la mise en œuvre des contrats intelligents lors de l'exécution de la composition. De plus, les résultats intermédiaires sont stockés dans l'IPFS, chiffré afin de créer un mécanisme d'accès aux données robuste réservé aux participants légaux du système blockchain. Pour le partage indirect des données, cette thèse développe une approche de préservation de la confidentialité pour l'apprentissage fédéré combiné à la technologie blockchain. La conception proposée FLBCshard, exécute des tâches d'apprentissage

fédéré décentralisées et sécurisées grâce au système IPFS et au partage de données basé sur des jetons non fongibles comme preuve de propriété. Le prototype concilie évolutivité et confidentialité de la blockchain grâce à une technique de partitionnement dynamique. Par conséquent, une architecture hiérarchique est proposée pour alléger les communications et vérifier la fiabilité afin de limiter les menaces à la confidentialité et à la sécurité. L'objectif principal de cette conception est de préserver la confidentialité des données des participants et du modèle global avec une grande précision, et de prévenir les attaques par empoisonnement, voire le vol de propriété du modèle.

Mots clés : Chaîne de Block, la protection de la vie privée, Hyperledger, chaîne de block privé, graph de la dépendance des services, la composition des services, apprentissage fédéré, partitionnement de la chaîne de Block, système de fichiers entre les planètes, jeton non fongible.

ملخص

أصبح تقاسم البيانات موضوعا حساسا في تكنولوجيا المعلومات لعقود من الزمن نظرا للتطورات التكنولوجية التي ساهمت في زيادة توليد البيانات من جهة جمعها واستخدامها من جهة أخرى وذلك من خلال إستعمال العديد من التطبيقات والخدمات، التي صارت سيفًا ذو حدين بسبب إستفادتها من الكم الهائل من البيانات الشخصية الفردية لتحقيق أرباحها الخاصة. ونتيجة لذلك، نشأت عدة مخاوف تتعلق بالهياكل المركزية، وانتهاكات الأمن والخصوصية.

تعرض هذه الأطروحة الجهود المبذولة للحفاظ على خصوصية مشاركة البيانات القائمة على التكنولوجيا الناشئة سلسلة الكتل التي تدعم كل من شفافية النظام،سرية البيانات ،النزاهة والأمن. تقوم الأطروحة بتقديم ثلاث إضافات جديدة للحفاظ على خصوصية مشاركة البيانات. في هندسة SOA نقترح أولاً منصة للحفاظ على الخصوصية، تسمى SDGchain معتمدة على تكنولوجيا سلسلة الكتل للتحكم في تفاعلات مجموعة من الخدمات كذلك مراقبه البيانات المتبادلة من خلال منح صاحب البيانات السيادة بوضع قواعد للولوج. تقوم العقود الذكية بمراقبة البيانات السرية المتبادلة للتمكن من تقديم رؤية عامه و شامله حول تفاعلات النظام. يوضح SDGchain متانة النموذج باستخدام سلسلة الكتل كطرف ثالث جدير بالثقة كما يبين تأثير إستخدام قاعدة بيانية خارج السلسلة تحمل المعلومات المشفرة كبيانات خفيفة كخطوة نحو توسع النطاق.

تم تطوير هذا البحث من خصوصية الخدمات الفردية إلى مشكل خصوصية الخدمات المركبه في نفس الهندسة،منصة PrsChain صممت لحل مشكلة تسرب وعرض البيانات الحساسة المتشاركة بين مزودي الخدمة،كما أن هذه الدراسة تساهم في التخلص من الإرتباط بطرف ثالث و تعويضه بتقنية البلوكتشين كهيئة موثوقة لتوليد مخطط التكوين وإدارة التنفيذ ، حيث أن الميزات الجوهرية كالمصادقة، مقاومة التغيير، النزاهة للبلوكشين تجعلها تكنولوجيا مغرية للإستفادة منها كبنية قوية للمنصة،حيث نناقش في هذه الدراسة الجوانب المختلفة لتقييم وتنفيذ العقود الذكية أثناء تنفيذ خطة تكوين الخدمات. إضافة إلى كل النتاج المولدة وسط العملية، تخزن في IPFS، بعد تشفيرها لتأمينها وضمان إنشاء آلية قوية للوصول إلها فقط من طرف المشاركين القانونيين للنظام.

كما يهدف هذا البحث للحفاظ على خصوصية البيانات المتشاركة المولدة بعد التعلم الموحدمقترنة بتقنية سلسلة الكتل .من أهم المهام التي يؤديها التصميم المقترح FLBCshard تعلم موحدلامركزيا و آمنا باستخدام IPFS و إستخدام مشاركة البيانات الرمزية غير القابلة للاستبدال كدليل على الملكية. يسعى النموذج دائما للتوازن بين توسع النطاق والخصوصية من خلال استخدام تقنية التجزئة الديناميكية، وذلك بإستخدام بنية هرمية لتخفيف الاتصال العلوي وإقتراح طريقة فحص مزدوج لموثوقية البيانات المتشاركة للحد وتقليل التهديدات الأمنية.و الهدف الرئيسي من التصميم هو الحفاظ على خصوصية البيانات المتشاركة والنموذج الكلي بدقة عالية ومنع هجوم تسمم المعلومات وحتى سرقة الملكية النموذجية.

الكلمات المفتاحية : سلسلة الكتل ,الحفاظ على الخصوصية,هايبرليدجر ,سلسلة الكتل المتطلبة لتصريح المشاركة, الرسم البياني المعتمد على الخدمة, تكوين الخدمة, التعلم الموحد ,تقنية التجزئة ,رمز غير قابل للاستبدال ,نظام الملفات بين الكواكب.

Acknowledgements

First, I would like to express my sincere gratitude to Almighty Allah for granting me physical and mental strength during my Ph.D. journey.

I would like to express my heartfelt thanks to my supervisor Prof. Makhlouf Derdour, whose expertise and patience have considerably added to my graduate experience. I am grateful for his constant encouragement, which helped me throughout this journey to hold the title of the researcher.

I would like to express my gratitude to Dr. Mohammed Gasmi, Dr. Abdelatif Sahraoui, Dr. Abdelmajid Benmachiche, and Dr.Pr. Toufik Maarouk agreed to review my thesis, as did Prof. Hakim Bendjena to head my defense jury. I am deeply grateful to the members of the LAMIS Laboratory, from students to professors.

Table of contents

Table of Figures

Table of Tables

General Introduction

1.1 Threats of Data Sharing

Our contemporary world has witnessed an ever-increasing collection and use of big data in different fields, which has become a double-edged sword. In our daily lives, many applications are willing to access our data, for example, asking for our geographic location, accessing camera, and even personal contact information. This results in a massive collection of individuals' data through utilizing these applications, on top of that, the daily use of social media platforms allows for the daily collection of personal data. According to a digital global overview report in January 2024 [126], more than five billion users are creating and exchanging their data daily, which increases violation threats.

Data sharing refers to querying and storing any type of data, such as raw data, metadata, and even indirect data (AI model parameters), where sharing data contributes to offering better services and outcomes for the public and government sectors. With the advancement of information technology regarding service evolution since the inception of the Internet, individuals have become aware of their personal data destiny through the spotlight on the concept of privacy as a heavy burden that should not be overlooked. In the context of data sharing, data are conveyed from users to company, users to service, or even service to service in a variety of forms in a wide network by employing security and privacy mechanisms. In fact, these mechanisms are not sufficient to fully protect shared data, which could lead to harmful consequences for user-sensitive information (corresponding identity, address, and chronic disease). Therefore, data must be shared for specific purposes in the right place with respect to privacy and ethical data protection rules.

1.2 Privacy Preserving in Data Sharing

Privacy is an essential human right; it is necessary for autonomy and the preservation of dignity by having the right to control how personal information is collected and used for what purpose. Roger Clark [76] defines the privacy of personal data as "the act to maintain individuals' integrity where privacy of personal data claims that data about individuals themselves should not be automatically available to other individuals and organizations, and that, even where another party possesses data, the individual must be able to exercise a substantial degree of control over that data and its use."

Data sharing is increasingly the mainstay of many technologies and services to acquire colossal economic and scientific production, which increases the risk of protecting these data against manipulation, theft, and loss that could include users ‘critical and personal data. Several domestic and foreign privacy laws emphasize what, how, and where to protect personal data ,in Algeria The Algerian Data Protection Authority (ANPDP) came into force in August 2023 after law No 18-07 was launched on June 10, 2018, to protect individuals' data in the processing, which requires operators to adhere to the requirements established in the law that was inspired by the European General Data Protection Regulation (GDPR) and others such as HIPAA and COPPA regulations, where all of them utterly seek to maintain a high level of data security to whom personal data regarding their compliance are processed, by boosting the transparency as long as using personal data with the right to access, modify, erase and even transfer with significant penalties. The essential challenge is to guarantee secure and private data sharing without revealing the user's identity or even inferring information about sensitive data. The data can be shared with third parties without the data owner's consent, which may lead to inferring some facts about personal life, whereby this information is utilized by malicious services or even scammer persons to ask for a ransom.

1.3 General Background

For decades, researchers have declared that guaranteeing data sharing privacy is a heavy burden that calls for deeply seeking adequate solutions for handling multifaceted issues that enforce data sharing privacy from disclosure through encrypted, anonymized, or perturbed original data (plain data). The encryption mechanism is intended to protect plain data from disclosure using cryptographic ecosystems. This scheme is mostly employed to maintain the privacy of theexchanged messages by adopting symmetric or asymmetric encryption. However, encryption-based privacy preservation does not adhere to full privacy preservation owing to the steady increase in the computation complexity, and it is deficient in some essential security objectives such as authentication, non-repudiation, and anonymity, that is, in case the secret key is stolen both the concerned parties lose trust ineach other, which requires managing the access control to obtain effective data sharing. Moreover, there is the potential to decrypt the shared data owing to the gaps in the mathematical algorithms formulas [1]. Anonymization is renowned for the concept of k-anonymity, which was introduced in [75] as a process of allowing data modification before it is given to data analytics [76]. The main idea is to combine sets of data with similar attributes in which each record is indistinguishable from at least k-1 other records. Thus, de-identification is probably impossible, as it overcomes the linking attack, which refers to mapping external data with anonymized data. K-anonymity is prone to two well-known attacks: homogeneity attack and background knowledge

attack, where taking over the dataset is the target. A homogeneity attack paves the way to L-diversity [3], which is intended to address such vulnerability by defining L as a well-represented value for sensitive attributes in the equivalence class (a set of records that have the same values). The application of L-diversity proved to be efficient. However, adopting such a technique over all types of data is not always possible, and implementing L-diversity is an NP-hard computing problem [77]. L-diversity is threatened by not skewness and similarity attacks. If the critical attribute is numerical and the values within a group are L-diverse but extremely similar, the adversary in a similarity attack can estimate the critical value by using a narrow value interval. As a result, ignoring the semantic closeness of these values. Another improvement of L-diversity is T-closeness as per in [78] it is "An equivalence class is said to have t-closeness if the distance between the distribution of a sensitive attribute in this class and the distribution of the attribute in the whole table is no more than a threshold t. A table is said to have t-closeness if all equivalence classes have t- t-closeness". Hence, this method is based on the Earth Mover Distance (EMD) [4], which is a metric of the distance between the distributions of sensitive attributes in each equivalence class. Therefore, the main drawback of t-closeness intends to protect against attribute disclosure instead of identity disclosure.

Data perturbation is intended to perturb the data or add some noise before the sharing process, the most common example is differential privacy, which refers to a mathematical framework standard for protecting an individual's privacy by allowing data analysis without revealing an individual's sensitive information in a dataset. The concept of differential privacy was first introduced by Dwork et al. [79], a reference work that intends to ensure individuals' personal data privacy for the sake of building a statistical database in which the mechanism's outcome is not affected by the existence or the absence of any individual record in the dataset, thus the probability to detect the difference is almost negligible, every algorithm realizes this constraint it is called differentially private, thus provides a formal guarantee that no information leakage about individual personal data. Achieving personal data privacy through differential privacy is realized by several mechanisms, such as the injection of calibrated noise while maintaining the data output utility for masking the contribution of individuals in a given dataset, compared to the entire accuracy of the analysis. The Laplace Mechanism is ε- differentially private, it is the best-known technique for achieving differential privacy in numeric queries, which pursues to add noise to the output of a function after computing this latter, consequently to the function sensitivity. The sensitivity of a function that preserves privacy is founded on determining how much its output must be perturbed. In other words, it is the greatest potential change that will occur in the output if a person is added or removed from an input dataset, which leads to determining the accurate amount of noise. The randomized response is

another mechanism of differential privacy that works with plausible deniability to perturb the individual data by asking them to answer ‘‘’ questions in a random way, in which some probability of a given honest answers and certain probability of giving a randomized one in such a way protecting their privacy whilst being able to collect and analyze their data. As such, RAPPOR, or Privacy-Preserving Aggregable Randomized Response, is an example of using a randomized response by Google [80] to protect users’ privacy, whose data is collected constantly by providing a robust privacy guarantee.

Although differential privacy has been supported by many research studies, the application in the real world is inefficient in terms of the diversity of the shared data. addition,Besides, it is hard to trade-off between the data utility and preserve privacy to get comprehensive protection. For example, if the calculated noise is too large, it is evident that the data becomes useless, and vice versa, and privacy cannot be reached.

1.4 Centralized vs. Decentralized Data Sharing Privacy Preserving

In centralized architecture, user data are collected and stacked with enormous amounts, which paves the way for internal and external attacks in that the data are maintained in one place. In a centralized environment and the absence of trust authority, the data are likely to be misused, and the entity privacy is abused as a real example of abusing the sensitive data privacy concerning individual blood relatives [75], let alone the awful consequences that may happen. In fact, personal data are always at risk, especially when these data are sold to third parties to make a material profit. These data are generated, shared, and even stored by third parties (stakeholders/service providers) without determining the access rights or specifying the time limit for the process, where the paramount challenges are privacy and security vulnerabilities, which undoubtedly affect harmful consequences on individual privacy.

The centralized systems failed in guaranteeing full privacy protection, which researchers declared that depending on one control point that contributes to creating a vulnerable environment, and the programming bugs are targeted to several types of attacks, the famous one is the risk of a single point of failure (SoF) as an example the denial of service attacks, that entails to an effective solution to fill the gaps relevant to security and privacy concerns, that holds decentralized, efficient, and trust properties to handle such problems. Blockchain technology as a distributed database provides these features and was further investigated by both academia and industry [6] as an effective, trustworthy solution for data sharing and privacy preservation. The distributed network was born with the public platform Bitcoin using peer-to-peer networks in order to eliminate relying on third parties and tackle

the double spending transaction issue. The emergent and complicated design, owing to relying on other technologies such as cryptographic primitives, smart contracts, and consensus protocols, is helping to create a secure and trusting environment to store the data in the form of timestamped chronologically chained blocks. This emergent solution has been used as a mechanism to guarantee privacy and security, where the transactions are accurately nonfinancial and include querying, storing, and even sharing the data in distributed and decentralized networks [81].

1.5 Blockchain and Data Sharing Privacy Preservation

Blockchain provides the solution of ensuring confidentiality and tracking the data logs in a legitimate manner, which handles several shortcomings of distributed and centralized platforms-based data sharing by referring to the owner to get the consent of sharing with stakeholders. To secure data sharing, Blockchain is combined with off-chain storage, where the origin data are secured and stored off-chain (ex-Mongo DB, Couch DB), and their metadata (the correspondent hash) is kept on-chain, which effectively ensures the privacy requirement such as authentication and auditability. However, the increasing number of participants will affect Blockchain response and performance regarding the huge number of transactions transmitted in order to create new blocks and synchronize the ledger, let alone block size. Thus, blockchain scalability remains a hot concern that must be addressed.

In an indirect data-sharing situation, the owners do not share their raw data, and they give only a set of vectors that represent the shared information. For example, sharing the model learned by machine learning methods is the best example of indirect data sharing. In contrast to the centralized learning paradigm, where the models are learned by central servers using the user's raw data, which leads to privacy issues, federated learning approaches aggregate local models learned by data owners, which preserve and protect data privacy. Blockchain technology has been merged with federated learning to support its trust, enhance its privacy, and add a security layer. The main concern of this merging is the scalability issue, where there is an increasing number of participants, which leads to a vast number of transactions and communication being overheard. Therefore, every blockchain-based federated learning must take into account the scalability concern, which is one of our main thesis objectives.

1.6 Motivation

Blockchain is a credible entity that ensures the security and privacy of a system that has been witnessed by many proposed solutions. Owing to the transparency, immutability, decentralization

and auditability handle relevant threats that make it an appropriate solution in many domains and even other technologies.

For raw data sharing situations, such as in service provider-centric mode, users' personal data are accessible, housed, and non-confidential by other parties without referring to the owner's permission, which leads to serious risks in terms of security and privacy that need to control, detect, and prevent unauthorized access to the system. However, integrating blockchain in a service-oriented architecture enables the user-centric mode to define access policies and data control by implementing smart contracts. New challenges in terms of privacy have arisen, although using cryptographic and non-cryptographic methods to protect identity and transaction data along with the user's permission to process the data by a specific service, where the question is how to ensure that the data privacy is still withstood and the identity will not disclose. The data protection legislation states that the data owner, maybe an organization, a service, or even an individual, should have control over personal data by pinpointing the requirements and the obligation on how to manage and process these data. The legislation imposes acts on how a service provider deals with a personal data owner even after processing, without considering that affecting service provider privacy may also affect user privacy. Thus, it is necessary to have a keen privacy inter-service.

In model data-sharing situations, such as federated learning, the adoption of blockchain as a secure database is widely used, which gives rise to scalability issues owing to communication overhead concerns such as the system's throughput and latency. These two metrics are affected and have not been proposed for many proposed works.

In the next section, we refer to any data, such as raw data or a learned model from the word data. We elaborate on some of the main research questions as follows:

- How to preserve privacy inter-service providers?
- How to preserve privacy in data service composition?
- How can the scalability of Blockchain-based federated learning be ensured?

This thesis proposes the integration of blockchain technology with privacy-preserving techniques to answer these questions. Starting with these research questions, this thesis discusses specific problems

1.6.1 How to preserve privacy inter-service provider

In the service-provider-centric mode, services share sensitive personal attributes to collaborate and achieve a satisfactory result after processing, and malicious services utilize these data to define data

owner identity and predict its behaviors . To this end, many solutions have been proposed to restrict service provider jurisdiction by integrating blockchain as a trusted authority and even as a secure database compliant with data protection legislation and enabling privacy-preserving schemes by adding cryptographic and non-cryptographic techniques to increase system efficiency. However, how do we solve the issues of data sharing and privacy preservation between services in a decentralized environment? How can we provide full control over service interactions? How do we track the data transformation? Furthermore, several studies have been conducted without integrating the blockchain, or when adopting blockchain, this scoop is marginalized.

1.6.2 How to Preserve Privacy in Data Service Composition

To achieve a user's query, multiple service providers are composed to act as one service to fulfill a desired task. As per the composition process, the data are exchanged among services, which depend on a third party to undertake the data transfer while employing cryptographic or non-cryptographic secure mechanisms. In this regard, trust in third parties can be considered a lack, regardless of the degree of reliability on one side and the availability of data. To this end, many existing solutions have opted to adopt a mediator with a low degree of confidence based on centralized architectures and non--scalable and non-practical security methods. Therefore, how can we eliminate trust in third parties in sharing data service composition? How can we protect the plan composition from tampering? How do service providers authenticate each other?

1.6.3 How to Ensure the Scalability of Blockchain-Based Federated Learning

Blockchain is renowned for its robust consensus that stands on agreeing on the same valid transactions, which are later replicated in all blockchain network nodes, creating overtime throughput and latency issues with regard to the chain size. Blockchain scalability has become a hot topic in applications that adopt transparency, security, and decentralization features. To this end, many solutions have been proposed to enhance blockchain scalability and preserve privacy in federated learning via a sharding technique, which requires high computational overhead in terms of the number of participants or energy resources. However, how can we ensure that dynamic shards are less likely to join malicious nodes? How do we guarantee model ownership and protect intellectual property from digital theft?

1.7 Research Objectives

Although the existing research on how the Blockchain succeeds in preserving raw data sharing privacy in different fields and cooperates with various technologies, where the majority of them concentrate on user privacy and marginalizes the service side in a service-oriented architecture. Because the service's trustworthiness and goodness are relative, affecting service privacy may lead to user privacy disclosure. Accordingly, filling this gap is the focus of this thesis. It is necessary to build a framework that enables the management and control of service interactions.

Moreover, in a service composition environment, it is evident that there is a need for a trusted mediator to authenticate the participating composition and preserve data privacy. Consequently, integrating blockchain, which acts as a decentralized server for managing data sharing queries in service composition, takes part in handling the Single point of failure that ensures data confidentiality and the integrity associated with each participating service.

Furthermore, this research examines blockchain performance in preserving the privacy of AI model sharing, especially in collaborative learning, also called federated learning, in which these two paradigms jointly aim to maintain user privacy. However, the large number of network communications, in addition to the number of blockchain peers, creates a challenge between privacy and scalability. To achieve a trade-off between model privacy preservation and blockchain scalability, the architecture employs a partitioning technique to alleviate the overhead costs and accelerate transaction validation. The following specific objectives were set:

- To define an access control mechanism based on blockchain to protect data from illegal access and manipulation and to ensure its integrity.
- To design a privacy-preserving framework that is compliant with data regulations and to grants the data owner full control over its data.
- To achieve the objective of secure and private data sharing in service composition, the blockchain should act as a mediator to coordinate services. This underlying role gives the proposed solution the ability to maintain data integrity and traceability for further audit and control.
- Ensuring the scalability of blockchain-based federated learning and boosting its security to provide efficient data privacy while maintaining scalability. A sharding technique was adopted to alleviate communication overhead.

1.8 Contributions

To answer these questions and resolve the aforementioned problems and issues, this thesis is divided into three main contributions that will be discussed in this section. To achieve the objectives listed above, our main propositions are as follows:

- **Contribution 1 (SDGChain):** In the first contribution, called SDGChain, which treats the privacy and security in sharing raw data attributes inter-services, a service dependency graph that contains a set of shared attributes between the services is created and saved on blockchain to maintain its privacy. Therefore, accessing this SDG is granted using only blockchain access control and data owner policies. In this situation, every service that needs given data must ask for the corresponding attribute from its owner, and in case of acceptance, the data will be transferred directly to the requester. Every service can put and define its access policies on its data, where it can define attributes as insensitive and sensitive. Concerning the non-sensitive attribute the access has no constraint, however, for the second, the consent of the data owner settles whether yes or no. All SDGChain operations are saved on-chain in a log registry for further audit and verification in the case of illegal data access. We used Hyperledger Fabric as a permissioned blockchain where every service must follow authentication and an authorization process to access the blockchain network and execute its smart contracts.
- **Contribution 2 (PrSChain):** The second contribution is called PrSChain, which addresses the problem of sharing raw data inter-services in order to answer a user's query that must incorporate multiple data sources from several service providers. In this situation, in contrast to previous works that rely on a central mediator, blockchain is adopted as a mediator to ensure decentralization and maintain the data privacy of all participating service providers. Similar to the first contribution, we used Hyperledger Fabric, and therefore, every participant was assigned a certificate to invoke smart contracts. Blockchain peers take responsibility for generating the composition plan and its query, where the latter are saved on-chain whereas the data used to execute the subqueries are saved off-chain. To ensure the scalability of PrSChain, the IPFS is incorporated as off-chain storage to save the intermediate results of the sub-queries. In this contribution, the participants are not aware of each other; therefore, they cannot guess who owns a given data result.
- **Contribution 3 (FLBCShard):** Apart from the aforesaid contributions which have been used in the context of direct raw data sharing, this contribution, called FLBCShard, relies on sharing a model learned from raw data in a federated learning paradigm. The latter allows to aggregate several shared models to obtain a final trained global model while keeping the data private for

the participants. One of the main concerns of blockchain cooperation with federated learning is scalability. Therefore, FLBCShard adopts blockchain dynamic sharding to resolve this issue. Dynamic sharding can reduce the number of transactions by assigning participants to a set of shards rather than to one main blockchain. Every shard must produce a model that is aggregated from the local models sent by the shard predicants. At the end of every iteration, a global model is generated by aggregating all shard models, and subsequently, a new shard formation is performed by generating new ones. The main goal of the last strategy is to eliminate malicious nodes that have been detected in previous iterations. Another benefit is redistributing participants according to their reputations to boost the accuracy of shards with low accuracy. In addition, our contribution uses proxy nodes as a gateway between the participants and blockchain to alleviate communication overhead and maintain privacy. In addition, to protect global model ownership, FLBCShard uses non-fungible token (NFT) model sharing. Therefore, every model sharing or ownership transfer must be performed using NFT. Similar to the two previous contributions, Hyperledger Fabric is used where two types of chains are defined one of the global models (main chain) and the other one for every shard (shard chain). IPFS has also been adopted as off-chain storage to store global models and their shared models and improve FLBCShard scalability and availability.

1.9 Publications' relation to our Contributions

In this section, we discuss the relationship between our publications and the contributions listed above.

- The publication Sdgchain: When the service dependency graph meets the blockchain to enhance privacy, corresponds to contribution 1 in Section 1.4.
- Prschain: A blockchain-based privacy preserving approach for data service composition, corresponds to Contribution 2 of Section 1.4.
- The publication Blockchain Sharding-based Federated Learning Empowered with NFT-based Privacy-Preserving Model Sharing corresponds to Contribution 3 in Section 1.4.

1.10 Thesis structure

This section presents the structure and organization of our thesis, where we provide a brief description of each chapter's content.

- **Chapter 1** provides a general introduction to the topic of our work by splitting the main problem into sub-ideas and proposing challenges that may face data sharing. Furthermore, our motivation introduces the purpose of this thesis, that is, the importance of this research, along with the research challenge questions. Moreover, the objectives of this thesis and its main contributions are presented. Finally, it presents the structure of the thesis, and a summary of each chapter.
- **Chapter 2** discusses the background and literature review relevant to this dissertation. The general background presents the main preliminaries and definitions of the concepts and technologies related to the thesis objectives. Moreover, the relevant literature review in the blockchain domain preserved personal data privacy. Furthermore, this chapter presents how blockchain technology preserves privacy by proposing the security and privacy properties and the taxonomy of the recent solutions to cryptographic and non-cryptographic techniques in privacy-preserving based blockchains. This section also describes the integration of artificial intelligence and blockchain as promising solutions for protecting data.
- **Chapter 3** introduces the worksrelated to this thesis which is taxonomized into three categories corresponding to our three contributions privacy-preserving data sharing based blockchain, privacy-preserving data service composition, and federated learning privacy preserving based blockchain. Furthermore, each category is discussed by mentioning the challenges and relationships with our contributions and the similarities and differences in features.
- **Chapter 4** presents a novel privacy-preserving data-sharing inter-atomic service model named SDGchain, with the integration of blockchain technology as a secure and trustworthy authentication and access control mechanism. This model adopts the service dependency graph for the sake of having a full control over service interaction, this scheme can assist in reducing the malicious behaviors via tracking and auditing the system ‘entity interactions. This work employs a permissioned blockchain viz Hyperledger Fabric to define the access permissions to the data ledger and off-chain viz CouchDB to maintain the encrypted data.
- **Chapter 5** details an efficient privacy-preserving mechanism for data service composition named PrSchain, which employs the developed smart contracts for effective users query execution using service composition. This system is built on blockchain technology, which acts as a trust mediator. This system employs a permissioned Hyperledger Fabric to generate and execute a composition plan. In contrast, IPFS, as decentralized data storage, is integrated to store the intermediary results after ciphering for secure, scalable and efficient data sharing.

- **Chapter 6** introduces a new scheme for the revolution of artificial intelligence named federated learning by maintaining privacy based on scalable blockchain using the sharding technique, which helps to partition the large network into subnetworks by clustering blockchain peers. This scheme, named FLBCShard, addresses the trade-off between model accuracy and system scalability. A permissioned blockchain is employed to grant decentralized, federated learning. This study proposes the use of non-fungible tokens that guarantee the owner's intellectual property.
- **Chapter 7** provides a summary of the conclusions and analysis of the thesis by highlighting future research directions to extend the existing thesis work.

Background and Literature Review

2.1 Chapter overview

In this section, we discuss the paramount concepts and the relevant literature review in the blockchain domain, which preserves personal data privacy. In the first step, section 2.2 gives the background of this thesis, including the main security and privacy preliminaries, the concepts and definitions of blockchain technology, and other related technologies. Section 2.3 discusses how blockchain technology preserves privacy by showcasing the security properties and then a taxonomy of the existing solutions to cryptographic and non-cryptographic privacy-preserving methods. This section also describes the integration of artificial intelligence and blockchains as promising solutions for protecting data. section 2.4 discusses some issues and challenges that may face privacy preservation based on the blockchain. We summarize the conclusions of the chapter in Section 2.5.

2.2 Background

This section will cover the definition of the key concepts and technologies that have been put forward in this thesis to give the reader a better understanding of the security and privacy mechanisms used to protect personal data by adopting blockchain.

2.2.1 Cryptographic methods

Paramount cryptographic primitives have been used and supported by blockchain technology (i.e., creating blocks) to promote the protection of sensitive data from disclosure by reaching security properties.

2.2.1.1 Symmetric key cryptographic

Symmetric key cryptography is best known as secret-key cryptography, it employs the same key for encrypting and decrypting a message, where the sender and the receiver share a single key known as a private key. The process in the symmetric key encryption starts when sender A encrypts a given plaintext m by using the secret key to get the ciphertext $m'$and send the result to the receiver, where the receiver applies an inverse operation using the same secret key K to get the plaintext m.

In symmetric key cryptography there are two types of encryption algorithms: *stream algorithm* and *block algorithm*. In the in-stream algorithm, data bits are encrypted every individual bit at a time

using a stream key without the need to maintain the data in the memory of the system., RC4, SALSA, and PANAMA are famous stream algorithms. Stream algorithms are typically faster and more efficient. However, they are considered less secure. Otherwise, the block algorithm allows the dissection of the data into chunks of fixed size and encrypts the entire block, which varies from 64 to 256 bits, using the same key for each block, as AES, DES, and *Blowfish* algorithms. Although block algorithms are slower than stream algorithms, they have many advantages that enable their application to many day-to-day activities on the Iinternet.

Despite the merits and facility of using symmetric key algorithms in various applications, which grant a considerable level of security and decrease cryptography complexity with better performance and fewer computations, some drawbacks related to them cannot be ignored.

- *Key distribution***:** Symmetric key encryption depends on the secure transportation of the key between the sender and receiver over a secure channel or by using a key exchange protocol. Kerberos is a key distribution protocol that allows parties to communicate over insecure networks and confirm their identities with one another in a secure manner.
- *Key large-scale management***:** It is practical and feasible to use only a few secret keys. When the number of users increases, it becomes challenging to manage and distribute secret keys, which leads to large-scale problems.
- *Digital signature***:** In a symmetric key, the secret key is shared only between the two parties. Both the sender and the receiver have the same privileges. Evidently, they authenticate each other. However, they cannot prove their identity to a third party, which means they cannot be authenticated. In cases where the key is stolen, this issue is handled with asymmetric cryptography, especially with digital signatures.

2.2.1.2 Asymmetric Key Cryptographic

It is known as public key cryptography, in which the key pairs (public key and private key) undertake the encryption/decryption process where the keys are related to each other. The public key is exposed to public. In contrast, the private key is kept hidden and saved in secure software called a wallet, where its primary role is to decrypt the ciphertext to plaintext. Because the encryption and decryption keys are not identical, asymmetric key cryptography solves this key management issue. Blockchain implementations employ asymmetric key cryptography for many reasons, authentication and integrity are the paramount incentives that enable the establishment of trust relationships among users who do not know or even trust one another. Although these cryptosystems have numerous

advantages, some drawbacks cannot be ignored: a slow process that takes much longer than symmetric cryptography, which uses a single secret key for the encryption and decrypting process. Asymmetric cryptography employs longer keys to provide higher security than symmetric key cryptography. RSA and ECC are well-known asymmetric algorithms.

2.2.1.3 Hash Function

The hash function is a mathematical procedure that transforms a random dataset into a fixed-length character series regardless of the input data size. A hash or digest is a unique outcome of applying a cryptographic hash function to any type of data (called a message), which could be a file, text, or image. Although the data are differentiated in terms of format and size, the hash function determines an output digest of the same size. For example, the hash function SHA-256 of the phrase "Hello world" is converted to a set of bits in the 25-bit string.$SHA3-256(Hello\ World) = 369183d3786773cef4e56c7b849e7ef5f742867510b676d6b38f8e38a222d8a2$

The main security properties of applying a cryptographic hash function can be summarized as follows:

- The hash function is deterministic, which means that for a given x as an input, the output result will always be a hash(x).
- The hash function is preimage resistant, which implies that it is difficult to retrieve an input x from a given output hash(x)=digest.
- Regardless of the amount of data input or format (text, image, or video), the hash function has the same size as the output data.
- Because the hash function is collision-resistant, it is computationally impossible to find the same output for different inputs.

The hash function is sensitive, which implies that even minor changes to the data inputs result in a different digest.

2.2.1.4 Digital signature

A digital signature allows one to associate a message with an entity. In other words, it is a digital signature as an electronic binding that associates the signer's identity with the origin of the message. It provides the authentication, integrity, and non-repudiation security objectives, and for example, signing a document digitally requires an asymmetric cryptography ecosystem using the private key sk to sign a message, then to verify the signature validity, the receiver employs the

corresponding public key pk. To obtain a valid digital signature, two operations were required. $sign(m)$ and $verify(m)$ which grants the following properties:

- $sign(-)$The input function is the message digest m which was computed using the hash function (i.e., SHA-256), and then the private key of the sender was used to produce the signature x .
- In the counterpart, the receiver employs the inverse function $verify(x)$ which takes as input the signature x and the sender's public key pk in order to return a true Boolean result if the signature is valid or false.

2.2.1.5 Merkle tree

A Merkle tree [82] is a data structure also known as a hash tree, which has been constructed based on a one-way cryptographic hash function to create an over hash pair until it maintains that only one hash is the hash root, or it is known as a Merkle root. The construction of the tree starts at the bottom, which represents the data items, and the leaves tree represent their hash. However, further nodes illustrate the concatenation of the two hash child data pairs until one hash left refers to the tree root. A merge tree is defined as a complete binary hash tree if each node has two children. In practice, the Merkle tree is designed to authenticate nodes. It is considered as a proof of existence to demonstrate that given data truly exists inside the tree only from the root, in which the tree is an efficient scheme to check data integrity, and can be broken into tiny pieces for remote verification, processing, or even transmitting data across the network.

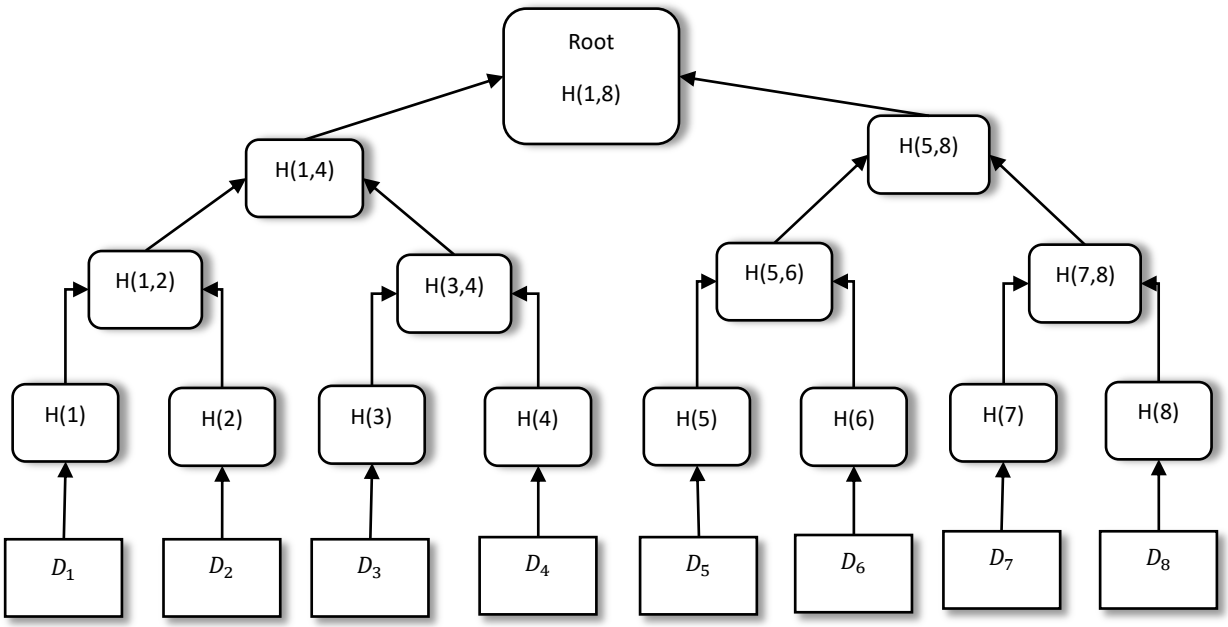

Figure 2.1 Merkle hash tree [82]

Figure 2.1 illustrates an example of a Merkle tree construction built from a set of data items. $D = \{D_1, D_2, \dots, D_8\}$The eight leaves correspond to the hash of each data item $H(i) = H(D_i)(i = 1 \dots 8)$. The value of the inner nodes refers to the hash of the child nodes. In the verification process within the Merkle tree, the root hash $H(1,8)$ allows to determine the data integrity and whether the data belong to a hash tree. To verify that D_1corresponding to the Merkle tree, we initially recompute the root node h from D_1 which requires computing the hash value of $H_1, H_{1,2} = H(H_1|H_2), H_{1,4} = H(H_{1,2}|H_{3,4}), H_{1,8} = H(H_{1,4}|H_{5,8})$, if the computed root hash is equal to the existing Merkle root $H_{1,8}$ then D_1 is part of the tree.

Recently, Blockchain cryptocurrencies have integrated and supported the Merkle tree for many advantages that allow transaction data integrity, thereby ensuring block integrity without modification or alteration.

2.2.2 Blockchain Technology

Blockchain is an emergent technology in the field of information technology with the aim of storing data in a distributed database to maintain data integrity, immutability, and traceability and considering a trusted entity to eliminate dealing with a third party. Blockchain first came with Satoshi Nakamoto, Bitcoin [83] as the first cryptocurrency. A peer-to-peer system with no trust deals with a group of nodes called miners, who work under a specific consensus according to the blockchain type. This database consists of an indeterminate number of blocks, where each block is considered as a container that includes all the validated transactions, timestamps, a Merkle tree, and the answer to the mathematical puzzle, in addition to a random number called the nonce. All of these data were considered as the body of the block. Otherwise, the header contains the hash of all the data saved in the current block along with the hash of the previous block (see Fig 2.3). Hence, Blockchain has prevailed in many fields, where the data are not restricted to monetary transfers. It accepts all forms of data.

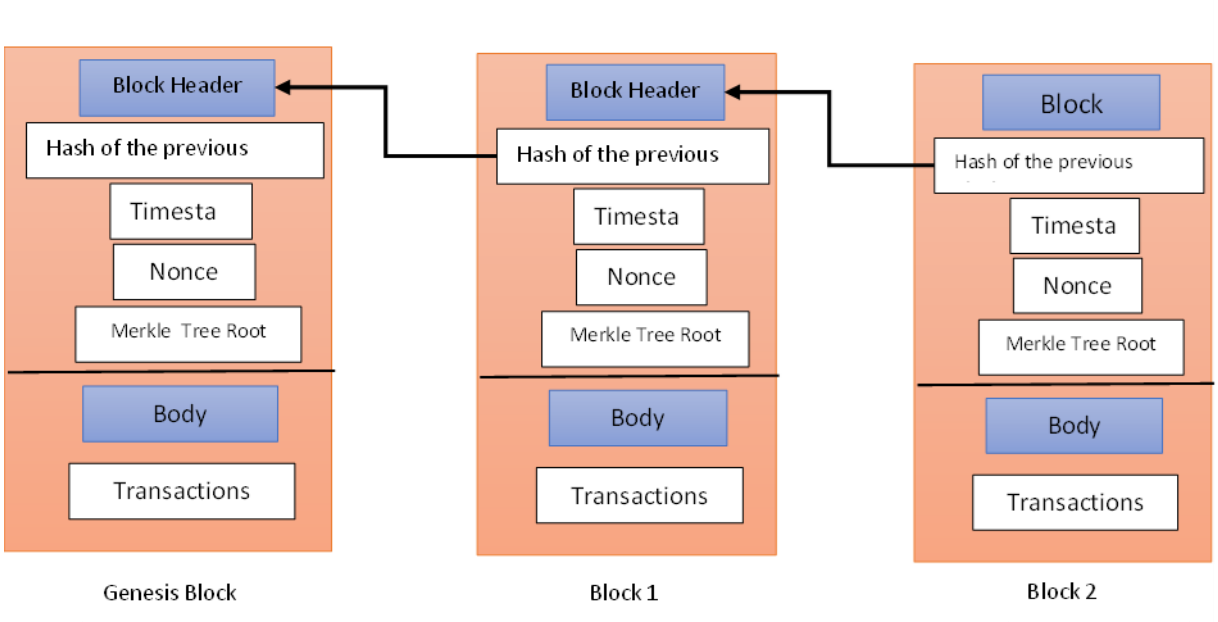

Figure 2.2 Blockchain structure [83]

2.2.2.1 Blockchain characteristics

The recent solution that handles several issues considers secure data sharing in a network of users' communication, data processing, and data storage, without requiring the involvement of third-party blockchain that has distinguished from other technologies regarding its characteristics, which led to building a subtle, unchangeable and robust ledger where we highlight the paramount characteristics of blockchain besides Zheng et al. [84] characteristics were adopted to point out the effectiveness of the blockchain which has made it unique and resistant against threats.

- **Decentralization**: duplicating the same information and spreading it among nodes in the blockchain network after validating the information without authenticating or interfering with a central authority conducted between two peers (P2P).
- **Persistency**: It is difficult to tamper with block content because each transaction is validated, confirmed, and spread out in the network before being recorded in the block. Therefore, the approval of inadequate transactions is more likely to be impossible.
- **Anonymity:** To interact with the blockchain network, users must generate one or more identities with their address, for example, the hash that hides the real identity.
- **Auditability:** Because the transaction was validated and recorded in a block with a timestamp, users can verify and trail traces of the previous records.

- **Immutability:** Blockchain is an unalterable ledger, which means that because the transaction was recorded in a block, it is permanent, and it is likely impossible to delete or tamper with it, which promotes security and trust.
- **Transparency:** Because of blockchain's suitability, users are allowed to access and track the history of their processed data, thus guaranteeing their provenance.

2.2.2.2 Blockchain overview

A Blockchain network can be an open distributed network that includes a set of users who are considered to be participant nodes; however, Yaga et al. [77] pointed out blockchain's overview where and we will go over the definitions of the key elements and outline their primary functions in structuring a blockchain network.

***Cryptographic Hash Functions*:** The hash function is considered the core component that ensures blockchain robustness and solidity against whatever threats. This section presents the definition and properties of cryptographic hash functions that thrust blockchain implementation to take advantage of the quickness of the generated hash output for security requirements such as authenticity and integrity. Blockchain employs several hash functions, including SHA-256, RIPEMD-160, and Keccak; however, SHA-256, which stands for the Secure Hash Algorithm, remains the fastest to compute and the best function supported by hardware. A cryptographic hash function is used in blockchain implementation for various tasks. According to [77], the hash is used as a unique identifier, address derivation, secure block data, and eventually secure the block header.

***Transactions*:** In the literature, a transaction refers to an agreement, contact, or exchange of assets between two or more parties. However, in a Blockchain network, a transaction is defined as the data exchanged across network nodes, which can be tangible or intangible assets. A transaction is a method of recording operations that occur in digital or physical assets. Fig 2.4 depicts an example of a Bitcoin transaction. Within Blockchain, each block has zero or more transactions, and the structure of transactions (data input /output) differs from one another. However, the basic concept remains the same: each transaction can be identified using a set of data inputs to obtain data outputs. A new transaction is issued by the user, broadcast to all nodes in the peer-to-peer network, and recorded in a new block.

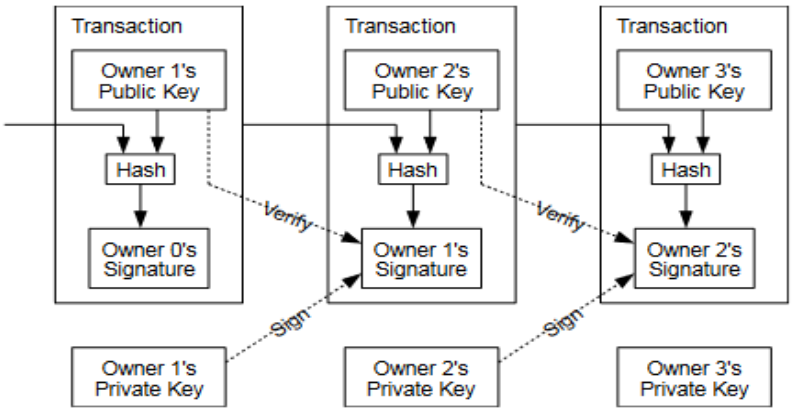

Figure 2.3 Example of Bitcoin transaction [83]

In most cryptocurrency transactions, the inputs are a list of digital assets to be transferred. Otherwise, transactions outputs are usually the accounts that will obtain the digital assets, as well as the number of digital assets they will receive.

***Asymmetric key Cryptographic*:** This section explains asymmetric key cryptography in detail. Asymmetric key cryptography tackled the blockchain security claims that each issued transaction is digitally signed using a private key that enables the recipient to check and verify the issuer's identity by decrypting the ciphertext with the corresponding public key. RSA and ECDSA are well-known asymmetric key cryptographies used in blockchain implementation. Blockchain implementations benefit from asymmetric key cryptographic properties, in which each key has its own role. For instance, in a blockchain network, the address is derived from the public key, which is used to verify the digital signature that is already signed by the corresponding private key.

***Addresses and Address Derivation*:** On a blockchain network, an address that serves as a unique identifier is utilized to send, receive, and even store digital assets, it is a string of alphanumeric characters that is used to verify ownership of a specific digital wallet. It is considered an essential component in blockchain implementations, thereby avoiding the interference of a third party (bank in cryptocurrencies) by facilitating secure and transparent transactions in a decentralized environment. Hash functions are used to generate blockchain addresses, which take input from either the public or the private key and result in a unique digest that serves as a unique identifier. Broadly, the process of generating a blockchain address takes four steps: generating the private key, generating the corresponding public key, extracting a unique address as a unique string of alphanumeric characters by hashing the public key, and finally, adding a checksum to the address to ensure that the address is valid and avoid transaction issues.

Ledger: The word ledger refers to a record of all transactions, and recently, blockchain technology has become a synonym for the distributed ledger, which has been stored digitally in a distributed network. Distributed ledger technology DLT [72] "refers to a novel and fast-evolving approach to recording and sharing data across multiple data stores. This technology allows transactions and data to be recorded, shared, and synchronized across a distributed network of different network participants" To maintain the transparency and verifiability of the record of transactions, all data are stored in blockchain network nodes (hold multiple copies). Therefore, any changes in the ledger imply a synchronous change of all ledger holders, where the data is added in an immutable and unchangeable manner during the ledger's life after a particular consensus between the participant nodes which is not required for intermediaries.

***Blocks and Chaining Blocks*:** Blockchain consists of countless ordered containers known as blocks. Each block contains a set of transactions, that are not confined to cryptocurrencies, which means that the block structure is identical in every blockchain implementation. However, this varied with the stored data. The block header includes the precedent block hash, timestamp, and a number that is changed by the mining node in blockchain networks that use mining to solve the computational hash puzzle in the Bitcoin Blockchain, the so-called nonce, as well as the Merkle tree root which refers to the hash of all transaction for maintaining block data integrity. Otherwise, the block body comprises a list of transactions and other data. Blockchain represents a set of chaining blocks where the chronological hash order promotes the security and trustworthiness of the blockchain, which facilitates the detection and rejection of altered blocks. To become immutable, robust, and tamper resistant

Consensus: Consensus, in general, means agreeing on the same information among a cluster of participants by abiding by a specific rule. In Blockchain technology, a consensus refers to the agreement that the ledger reaches among distributed network nodes. As long as consensus is achieved, trust in a third party is eliminated, which enables blockchain nodes to validate and record transactions. The block is created by particular nodes, where the paramount utility of using a consensus strategy is to organize and regulate the process of creation, even appending a new block, in addition to mitigating malicious attacks, thus increasing trustworthiness. Myriad consensus algorithms have been proposed to adapt blockchain implementations in various applications for a variety of domains, and we solely focus on the ones that have a relation to this thesis.

- ***Proof of Work*:** This is a consensus that is supported by Bitcoin Blockchain where the publishing nodes (so-called miners) attempt to solve a challenging computation puzzle by

founding a Nonce, a random number that satisfies a certain condition to create a new block; once a miner finds the random number, it is the winner node that will be rewarded via network tokens and transaction fees thus publishing a block. The entire process is known as the mining process. Even though adapting PoW enabled the Bitcoin blockchain to be resistant to double spending attacks, several performance bottlenecks and sustainability problems have appeared [9], such as, unsustainable energy consumption, low transaction capacity, poor scalability, and long-term security concerns as mining rewards diminish.

- ***Proof of Stack*:** For the sake of shrinking the high computational requirements triggered by PoW, Peercoin [10] is the first proof of stack (PoS), which is another consensus algorithm where the participants seek to solve the mathematical puzzle in an energy-saving manner without the need for huge computational resources; in the mining process the participant who has a higher chance of being selected has a higher coin age which is so-called leader based on his stake that he is holding. The stacks are digital tokens, for example, coins in cryptocurrencies, where the leader solves a difficult problem with his own resources and difficulties, thus adding a new block, which can diminish coin age consumption. PoS is distinguished by PoW in terms of faster transaction confirmation time.
- ***Proof of elapsed time*:** Similar to PoS, proof of elapsed time (PoET) was developed by Intel Corporation in 2016 using Software Guard Extension (SGX) technology and implemented in the open-source Hyperledger Sawtooth blockchain framework. This method is based on the PoW strategy. However, it is proposed to prevent high computational resources and waste of PoW energy. The core idea of the consensus is that each user must wait for some time generated by the code running inside a "trusted execution environment" (TEE) before it is allowed to create a block. The waiting time must follow a probability distribution. F which was determined by the scheme [85]. Intel SGX technology was adopted to securely generate a certificate for the public key and deploy it to the system.
- ***Practical Byzantine Fault Tolerance:*** To solve Byzantine general problems, PBFT was introduced in the late 90s by Miguel Castro and Barbara Liskov [11]. It aims to tackle the issue of guaranteeing the consistency and correctness of the final decision despite the presence of malicious nodes throughout the network, where PBFT guarantees liveliness and safety if at the most numerator, n minus one end numerator, over three out of a total of n replicas are faulty nodes. Faulty nodes do not affect the entire network's ability to reach a consensus, and include two types of nodes, that is, a primary node and a backup node. The client node requests the primary node by issuing network transactions. The primary node

then selects the request's execution order and spreads it all over the backup nodes. After receiving the request, the backup nodes verify their authenticity, decide whether to execute it and respond to the clients [12]. It is also possible to divide the nodes into Byzantine error and normal nodes. In PBFT, there is a time stage mark called view. Each view used a unique primary node. If a primary node messes up, the backup node achieves consensus and moves on to the next view [86]. As the number of nodes increased, the network became more secure. To execute a transaction, the majority, in other words, at least 51%, of the network nodes must approve the transaction [13].

- ***Raft consensus:*** The Raft consensus was proposed by [87], building upon the idea of fault-tolerance, a leader-based log replication protocol, where the peers are categorized as leaders, followers, or candidates. The leader is selected through democratic elections to create blockchain blocs, where the network nodes enter into competition to become the leader node when they receive the majority of the nodes' votes. A follower becomes a candidate when he does not hear from the leader for a certain period of time, and the candidate asks for votes from other nodes in order to become a leader. A follower maintains their vote and does not grant it to a candidate when it receives a heartbeat within the minimum election timeout of hearing from the current leader. It does not grant its vote to the candidate, which helps maximize the duration of a leader's work while avoiding frequent interruptions from some unsolicited nodes for which they have not received a heartbeat. In a Blockchain network, the leader is the node that receives all transactions, and each node is added to the node's ledger as an entry. In particular, the leader is in charge of the ledger replication process, which involves replicating all new transactions received by followers. Raft consensus has been adopted by many approaches based on blockchain technology owing to its low complexity and rapidity which can improve the scalability of the distributed ledger.

2.1.1.2 Blockchain taxonomy

Blockchain was originally known as a public ledger to solve a double issue of trust on a third party, then according to business needs, which in the development and appear several projects that can be characterized as building upon the level of permission that can be categorized as public, private or consortium blockchain considering the access rights to the network and the level of centralization.

- ***Public Blockchain*:** The best-known permissionless blockchain is that anyone can participate and join the network as a node(miner) without restriction by creating, validating, and appending the transactions requested in blocks of data for cryptocurrencies by solving a

cryptographic puzzle, where the public blockchain is completely decentralized such as Bitcoin, Ethereum, and Litecoin.

- ***Private Blockchain*:** also known as managed blockchain, whereas access to the network is controlled and restricted by a central authority and allows only the legal nodes to join the network through grant permission which defines that private blockchains are partially decentralized. Examples of open-source private Blockchains are Hyperledger and Ripple.
- ***Consortium Blockchain*:** It is a mash-up of public and private blockchains, it is renowned for federated blockchain as well, which is permission and governed not only by one central authority, which leads to a fully decentralized environment. Typically, nodes are pre-selected, equally privileged, and considered semi-private. Hyperledger, Corda, and Quorum are examples of consortium blockchains.

2.1.1.3 Blockchain's comparison

This section presents a detailed comparison between public, private, and consortium blockchains, providing brief definitions and a subsequent comparison of the famous and open-source platforms, Ethereum, Hyperledger, and Corda.

- ***Ethereum:*** A public, open-source, and decentralized platform that supports smart contract features to build and develop applications launched in 2015 when Vitalik Buterin [14] declared various shortcomings of the scripting language of Bitcoin, which had a weak version of the concept of smart contracts. In the Ethereum network, participants had identical roles and tasks. The ether is used as a cryptocurrency in Ethereu. Recently, in September 2022, it switched from proof of work consensus to proof of stack, where validators created new blocks and worked together to verify the information they contained.
- ***Hyperledger Fabric*:** A distributed foundation for developing applications or solutions with a modular architecture, is an open source that was established under the Linux Foundation and designed for use in enterprise contexts. In Hyperledger fabric nodes take different roles and tasks to reach consensus; they could be clients, peers, or order.
- ***Corda*:** The Corda platform has been distributed and open-source software since 2016, and is designed for recording and processing financial agreements, specifically for use with regulated financial institutions. In Corda, a digital document that records the existence, content, and current state of an agreement between two or more parties is called a state object. When nodes in Corda reach a consensus with transaction validity and transaction uniqueness, this platform supports several consensuses [15].

Table 2.1 : Comparison of Ethereum, Hyperledger Fabric, and Corda [16]

Characteristic	Ethereum	Hyperledger Fabric	R3 Corda
Description of platform	Generic blockchain platform	Modular blockchain Platform	Specialized distributed ledger platform for financial industry
Governance	Ethereum developers	Linux Foundation	R3
Mode of operation	Permissionless, public or private	Permissioned,private	Permissioned,private
Consensus	Mining based on proof-of-work (PoW)/Ledger level	Broad understanding of consensus that allows multiple approaches/Transaction level	Specific understand-ing of consensus (i.e., notary nodes)/ Transaction level
Smart contracts	Smart contract code (e.g., Solidity)	Smart contract code (e.g., Go, Java)	Smart contract code (e.g., Kotlin, Java) Smart legal contract (legal prose)
Currency	• Ether • Tokens via smart contract	• None • Currency and tokens via chaincode	None

Table 2.1 [16] illustrates a summary of the three aforementioned Blockchains unless Bitcoin, its public nature, contributes to the exploitation of potential vulnerabilities. In addition, the PoW consensus that requires colossal energy consumption pushes us to opt for Hyperledger Fabric Blockchain in our study for numerous reasons. The top one is the permissioned environment that allows users and clients to interact privately, which leads to the determination of only legitimate members of the company or the organization. Moreover, the modularity feature makes the desired implementation pluggable for the sake of increasing confidentiality, resiliency, flexibility, and scalability, as well as the merit of a swapped consensus mechanism that enables designers to choose their appropriate consensus. Furthermore, many studies and performance analyses have shown that Hyperledger Fabric outperforms Ethereum in terms of average latency and throughput. Moreover, Hyperledger Fabric consumes less hardware resources than Ethereum[88]. On the same scale, Corda is a sheer private and permissioned blockchain that would prevent privacy applications for financial services only. In contrast, Hyperledger Fabric is designed for many industrial use cases that motivate researchers to integrate Corda into Hyperledger Fabric, which is seen as a complement, not a competitor [16]. Recently, in 2020, Hyperledger Fabric proposed a new term for private data

collection, with the aim of increasing data privacy among a group of organizations on a channel that needs to keep data from other organizations on that channel.

2.2.3 Interplanetary file system (IPFS)

IPFS [132] is a peer-to-peer data-sharing distributed protocol that allows storing, accessing, and sharing files, directories, and websites in a decentralized manner to render the web more rapid and secure. IPFS supports a decentralized landscape by interconnecting a set of computers called nodes that employ distributed hash tables. In IPFS, the data is addressed using its contents instead of its location (IP address), which indicates that once the data are uploaded to the network, IPFS returns the corresponding hash that will be used for requesting these data.

The main components of IPFS are the following:

- ***Distributed Hash Table (DHT):*** This is a distributed data structure that maps keys to their values. DHT is the core component of an IPFS that empowers nodes to store and retrieve content from other nodes in a decentralized network. Similar to a hash table, every node in the network can request a value corresponding to a hash key, by mapping the data requester to the peer that stores the matching content.
- ***Block Exchange:*** The exchanges are performed by counting on a peer-to-peer data exchange BitSwap protocol in order to exchange data between nodes. The ptotocol serves to reward nodes that partake to each other, in contrast it punishes those who only request resources.
- ***Merkle DAG:*** This is a merge between the Merkle tree and directed acyclic graph, which tracks data file alteration and damage in a distributed manner, using acryptographic hash function to organize data blocks.

2.2.4 Sharding technique

The sharding technique partitions a massive database to gain rapid micro-databases known by the shards to be more manageable and smoother [89] for the sake of keeping them inside isolated servers to alleviate the pressure of relying on one server, which helps to enhance the performance and increase the storage capacity of the entire database [17]. This technique is employed in blockchain to divide the entire network into smaller subsets known as shards or committees, each of which contains a defined number of network nodes.

In every sharding epoch, only one shard treats a set of transactions, which prevents double spending, does not hinder the parallel processing of transactions on distinct blocks, and maintains ledgers. Scalability is achieved when the system throughput and latency are combined. For that purpose, a

greater number of shard nodes increases the throughput that the blockchain system augments by constructing a computational environment among shards nodes, thereby decreasing the communication costs by achieving a consensus among shards and, as a consequence, diminishing the replication of the entire network. Moreover, reducing the enormous number of recorded transactions results in earning of data storage. Many solutions have been proposed to handle the challenge of scalability by adopting a sharding-based consensus.Some of the best known permissionless blockchain solutions are Elastico [90], Omniledger [91], RapidChain [92], Monoxide [93], Zilliqa [18], and Harmony [19]. Permissioned Blockchains have also used the sharding technique, for instance, Cosmos [127], RSCoin [94], AHL[95], Blockplane [96], and Sharper [97].In general, blockchain sharding has the following essential properties:

- ***Shard formation*:** To partition the blockchain into shards, each participant is designated to a specific shard, by generating an identity that requires for example the participant public key, its IP address, and a solution for the PoW puzzle to defeat the Sybil Attack [98]. This step is customized only for the public blockchain because the permissioned network participant identities and numbers were previously defined.
- ***Intra-shard consensus*:** In each epoch, the nodes within the same shard run a predefined consensus protocol to agree on the same block that includes valid transactions. PBFT is famous for its efficiency in dealing with Byzantine nodes.
- ***Cross-shard mechanism*:** A transaction consists of multiple inputs that drive the cooperation of multiple shards to be valid. This process is called cross-shard processing, in which the transaction is divided into sub-transactions, where each shard is in charge of confirming whether the input is valid by ensuring that the output shard does not deal with the transaction is valid unless all related input shards are altogether committed. Eventually, the transaction is considered valid in the entire system if the sum of the sent inputs equals the sum of the outputs [20].
- ***Shard reconfiguration*:** Participating nodes must be updated periodically to ensure shared security and liveliness. Fostering a random selection strategy before launching the next shard epoch, helps eliminate the case where an adversary compromises a certain node by replacing the old nodes with new ones, to ensure that the rate of malicious nodes will not pass the pre-defined safety threshold (e.g. 1/3 in PBFT).

2.2.5 Non-Fungible Token

Non-fungible tokens (NFT) are regarded as a type of cryptocurrency derived from Ethereum smart contracts that aim to define intellectual property, such as digital assets, using distinguished tokens where NFTs are similar to cryptocurrency only in the programmed design. Blockchain assets can be represented as fungible or non-fungible tokens. According to the NFT definition, tokens are indivisible and unsubstituted by other tokens of the same type, rendering them unique and unable to be exchanged like-for-like, which empowers the identification of something or someone in a unique way [128]. This is a proof of ownership. Fungible tokens can distinguish tokens that consist of a unique ID and the corresponding metadata to distinguish each asset from the others. The first NFT was created in 2017 as the first use case of an Ethereum Blockchain game, which was a virtual online game named CryptoKitties that allows players to trade with virtual cats. This departure contributed to the large proliferation of NFTs to prevail in all domains; another well-known example is the National Basketball Association Top Shot for the purpose of buying and selling video clips of basketball moments. NFT prevails in real-world marketplaces by tokenizing tangible and intangible assets. For example, the largest NFT marketplaces are OpenSea, Foundation, and Rarible.

2.3 Blockchain preserves privacy

2.3.1 Blockchain Security and Privacy Properties

Blockchain is designed to boost the security and privacy of personal data using asymmetric key cryptography (the public and, private keys) to issue a corresponding transaction without disclosing the user's real identity. Therefore, Blockchain guarantees the security and privacy that relies on cryptographic primitives and the design's nature, which contribute to making it unique and emergent in dealing with online transactions. This section discusses a set of security and privacy properties that enable blockchain trustworthiness.

- ***Ledger 'consistency*:** The consistency of Blockchain as a distributed ledger stands for all network nodes replicating the same information of the ledger at each splot of time. In particular, the consistency of the first blockchain implementation of Bitcoin is argumentative. Therefore, research opinions are bifurcated into strong consistency [73] and weak consistency, known as eventual consistency [99]. A problem is found when responding to the request of the reader with stale data before updating the ledger. The eventual consistency model brings together the availability and consistency, which is proposed for distributed computing, which ensures broadcasting the data updates all over network nodes

in a lazy way and provides the reader with the last value, which diminishes the risk of obtaining stale data. Blockchain technology addresses this challenge by recording a new transaction request into a new block and starting a new computational puzzle proof of work. Then, the miner in charge propagates the block to all peers along with its proof and looks forward to getting back to the acknowledgements. The other miners, after verifying the transaction's validity, start to add the new block and generate its hash from the precedent block hash.

- ***Tamper-resistance 'ledger:*** Tamper-resistance refers to the ability to resist before any attempt to tamper with the data whether inwardly (users) or outwardly (adversaries). Blockchain is renowned as a tamper-resistant ledger in that transaction data are not tampered with during block generation or post-generation. In Bitcoin, the miner is in charge of generating the block; in such cases, it can modify the transaction signature or even tamper with the data content. Bitcoin treats such problems by signing the transaction with the sender's private key using ECSDA and generating the transaction hash value using the hash function SHA-256, on top of that, no transaction is added to the ledger before checking the validity of the transaction by the other miners. The exterior threat occurs when an adversary attempts to modify the block's content, and bitcoin tackles this issue by chaining the blocks cryptographically using the hash value. As a result, any data change will be remarkable and impossible, owing to the fact that each node of the network has a replica of the ledger.
- ***User Pseudo-anonymity:*** Pseudonymity is a technique that extracts a hidden identity from necessary information to identify an entity. Blockchain deals with pseudonymity to identify anetwork user's address in the form of ahash value for the corresponding public key to protect the user's real identity. Blockchain users have the right to generate more than identity to increase their confidentiality and privacy.
- ***Distributed denial of service attack resistance (DDOS):*** Distributed denial of service attack is a famous cyberattack, where the idea behind it is flooding the system with fake messages in order to hinder a particular set of functions performed to render the server unavailable and out of service that are organized from multiple disparate sources that are distributed across the Internet, which results in cutting the communication between users and the system. The challenge that may be faced by the Blockchain is whether the DDOS attack may affect its availability when compromising a considerable number of network nodes. In fact, blockchain is designed to be resistant to DDOS because its functionality and processing for

mining and creating blocks are carried out even though the majority of nodes are dropped out owing to its decentralization and p2p protocol of communication.

- ***Resistant Consensus majority attack:*** This refers to 51% of blockchain nodes cheating in the consensus. In practice, one single node takes over the control of more than 51% of the computational power of the network. Among the serious risks of compromising more than half of the network node DDOS attacks, the double spending attack refers to spending the same coins more than once, which leads to damage to the trust and reliability of the blockchain, tampering with transaction information and blockchain history, and preventing the rewards of validators. In fact, performing such a task is non-trivial and requires high computational energy resources and time. One of the effective solutions is switching to another consensus protocol to decrease the risk of control.

2.3.2 Privacy-Preserving Techniques Used in Blockchain

Blockchain technology has attracted the attention of the world through its decentralization, immutability, transparency, and trustworthiness, without the need for third authorization. With regard to the peer-to-peer protocol that makes the distributed ledger tamper-proof, these features assist many business applications in embracing this technology to improve the security and privacy of end-users by recording and auditing their data in a decentralized environment. Bitcoin and Ethereum, the public ledgers that log cryptocurrency transactions in the clear, have deluded the world by maintaining the privacy of the user without requiring main cryptographic methods for building blocks that are used to support privacy and settle for the anonymized identity of the user by generating as much one, despite the fact that there is no need for real identities. Many studies have confirmed how evidently linking the random generating addresses to their real users [21,100,101], which results from raising some burden concerning the privacy and confidentiality of the data stored. Blockchain ensures security and privacy protection, which counts on the fostered consensus level, not the data privacy itself [102], which implies a call for cryptographic and non-cryptographic privacy-preserving techniques to strengthen privacy. Research has pinpointed the following fundamental types of privacy-preservation techniques based on blockchains [61,62]. The classification of blockchain privacy-preserving mechanisms is described in this section and illustrated in Figure 2.5.

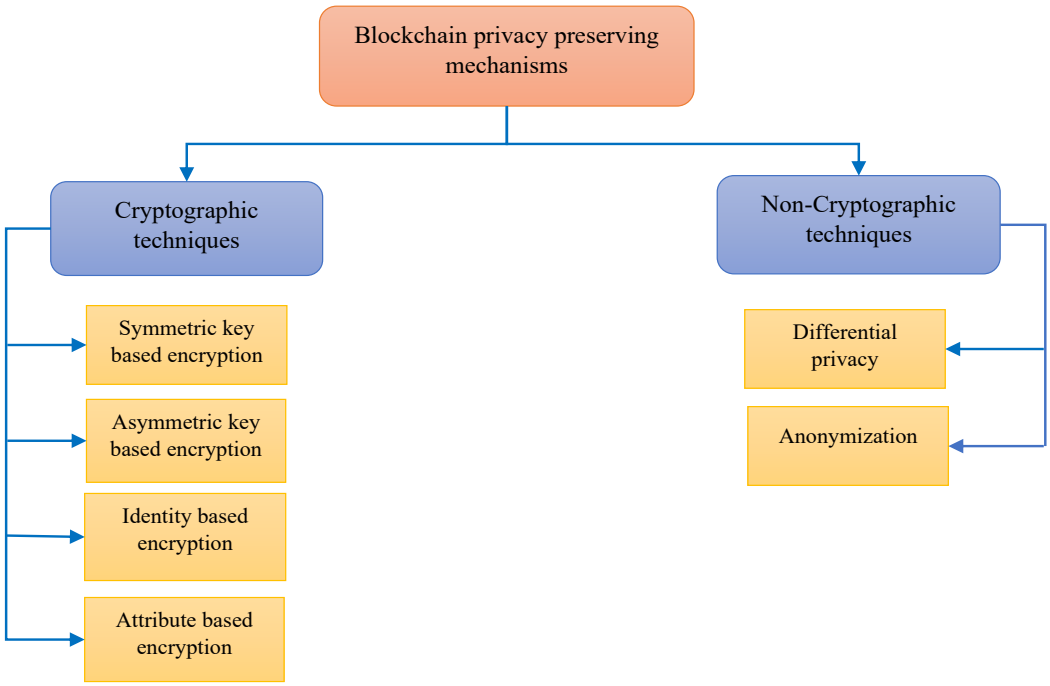

Figure 2.4: Blockchain privacy-preserving methods Taxonomy

2.3.2.1 Blockchain privacy preserving based cryptographic approaches

The concept of cryptographic methods refers to symmetric or asymmetric cryptographic ecosystems, which are approved methods in blockchain to ensure confidentiality for both identity and transaction data and to protect the privacy of users. Encryption techniques are integrated into the blockchain for the sake of security requirements and data ownership verification [24]. Symmetric key cryptography is integrated into Blockchain to ensure confidentiality and non-repudiation by allowing only authorized users to encrypt and decrypt the data using the same key that has been transferred to a secure channel for communication, whilst asymmetric key cryptography refers to employing key-pair cryptography to encrypt and decrypt the data, and the public key can be shared, otherwise amongst the private key functionality is the digital signatures in order that proves the right sender or even to guarantee that only a desire receiver can read it. Several methods, such as attribute-based encryption and identity-based encryption, have been integrated into blockchain to guarantee identity data privacy and manage the access control process. In this section, we provide a detailed survey of vital proposed works based on cryptographic approaches that ensure the security and privacy of personal data in different fields.

- ***Symmetric key encryption-based approaches:*** When using the same secret key between the sender and the receiver, a symmetric key is used on a larger scale in blockchain technology, which alleviates cryptography complexity with better performance and fewer computations; the main algorithms used are data encryption standard (DES), advanced encryption standard (AES), blowfish and stream ciphers such as RC4 and A5/1. In the following section, the main symmetric encryption-based approaches are presented.

  Panwar [103] proposed BlAnC, a distributed credit network-based blockchain that ensures the unlikability between user identity and transaction data to ensure anonymity, where credit transfer between a sender and -receiver pair occurs on demand using the symmetric algorithm AES-256 by performing path-based transactions. The proposed approach based on blockchain publicly certifies transactions and identifies malicious parties in the form of transaction records. The BlAnC design was fully decentralized. A secure and private sharing system for email communication based on blockchain technology for managing access control proposed by [25], where the design proposed a shared identity between users and services along with the associated permission for privacy requirements and are kept on-chain, the Blowfish cryptographic encryption is used to encrypt the email information transaction in -off-chain storage, and recuperate the pointer that will be used later for reading the email from users or service.

- ***Asymmetric key encryption-based approaches:*** Asymmetric key encryption, renowned for public key cryptography, is widely applied to many application-based blockchain systems to realize data sharing security and manage the access control process [23]. In the following sections, some approaches based on public key encryption are described.

  In [104], ProvChain is proposed as a platform for data provenance verification at the cloud storage level based on permissioned Blockchain, where the architecture tackled the problems of gathering and verifying data provenance by the services provider via auditing and recording the hash of the operations that occurred on data as a transaction in the Merkle tree. The proposed design applies public key encryption for data sharing and blockchain as a secure database to record any provenance data entry. The work in [26] presented a blockchain-based solution to enhance the security and privacy of virtual circuit-based devices, which was implemented in an Ethereum public blockchain and evaluated using an IoT-based application in a virtual vehicle monitoring system. To retain the privacy of the stored data, this approach utilizes public key encryption pentazole elliptic curve cryptography and the secure hash algorithm SHA for integrity. Blockchain's stark nature

assists in generating an intruder alert in any instance of unauthorized access. To protect sensitive data from intruders, MediBchain [105] proposed a platform based on blockchain for adata management framework and to ensure data integrity. A user-centric model was proposed that allows only the user to control their data. Public key cryptography is used to ensure patients' pseudonymity by assigning each patient an identity (ID) that refers to his or her pseudonym. Using this ID, smart contracts search for and retrieve the ID of the corresponding blocks.

- ***Identity-based encryption approaches:*** In 1984, Adi Shamir proposed the idea of identity-based cryptography (IBE) [106], a type of public key cryptography and an alternative to using digital certificates. It settled for generating identities solely using the user's attributes such as email address, cellphone number, IP address, etc. Encrypt and decrypt the messages. This idea is based on bilinear nondegenerate maps, which are mathematical functions and pairing elements from one cyclic group to another of the same prime order, in which the discrete log problem is difficult in the first group [129]. IBE requires the involvement of trust in a third party to generate a private key, called the Private Key Generator. In [107] Sash, a data-sharing framework is proposed that brings together the blockchain technology and IOT platform, in which Hyperledger Fabric Blockchain was employed to safeguard access control policies and make access control decisions jointly with identity-based cryptography and audit all operations. In the same endeavor and seeking to secure the IOT architecture, [27] proposed a healthcare system-based blockchain that enables the protection and preservation of the privacy of collected medical files using an IBE algorithm empowered by smart contracts that define the authentication policy in a simple manner and define the roles of each actor.
- ***Attribute based-encryption approaches:*** Attributed-based encryption (ABE) is a public key encryption technique derived from IBE proposed by Sahai and Waters [108] as a fussy IBE that defines the basis of encryption and decryption of the ciphertext based only on the user attributes that satisfy the access rules pinpointed by the system. The attribute-based encryption scheme was successfully coupled with encryption and access controls. Generating a secret key from an identity requires a set of descriptive attributes in addition to the access structure that carries out access control. The paramount types of ABE are based on the place where the access structure is attached [109]; they are categorized as key policy attribute-based encryption (KP-ABE) and ciphertext-based attribute-based encryption (CP-ABE). In the KP-ABE scheme, the user key is associated with the access policy, where each

key is labeled by an access structure that defines the type of ciphertext that can decrypt. In the CP-ABE scheme, the data owner can define a policy of who can decrypt the cyphertext. In other words, encrypted data are accessed only by the user whose attributes satisfy the desired policy.

Integrating ABE with Blockchain to secure cloud data sharing in order to eliminate reliance on third authority, the work of [28] proposed BCAS, a secure scheme that protects data owner rights using CP-ABE along with blockchain for managing access control and guaranteeing user authentication and privacy. The authors took advantage of integrity and traceability to maintain each hash of the key, the hash of the decrypted data, and the hash of the ciphertext. With regard to cloud computing, prone to several attacks and securing digital documents [29] proposed a solution for protecting the privacy and confidentiality of the shared documents by restricting user access policies by data owners using CP-ABE and Ethereum Blockchain; for scalability, the interplanetary file system is used to store the encrypted data to alleviate block size and eliminate centralized storage as a single point of failure.

2.3.2.2 Blockchain privacy preserving based non-cryptographic approaches

Non-cryptographic techniques renowned as well by masking, several methods were proposed to deal with privacy leakage without adopting the encryption and decryption process for securing sensitive data, in this section, we give a detailed survey of the significant research approaches based on differential privacy and anonymity.

- ***Differential privacy approaches:*** Regarding differential privacy, a mathematical technique is applied to preserve users' sensitive data in large statistical databases by adding a specific amount of noise to the data before evaluating the query after calculation. In such a way that the absence or presence of a specific user is indistinguishable, the ubiquitous use of the concept of differential privacy even in real-time applications. Blockchain is a decentralized and distributed data storage that is required to protect user and transaction data from disclosure owing to the use of pseudonymity, which cannot lead to a fully private environment. Integrating differential privacy data perturbation as a strategy to overcome certain blockchain privacy issues by perturbing the blockchain node's identity by adding noise [30], wherein the data perturbation mechanism, an error rate is calculated, and then a noise is calculated using the error rate to ensure privacy requirements. Subsequently, noise was added to a specific value to ensure privacy protection. The recorded value is differentially private, and consequently, an adversary cannot distinguish the exact value,

absence, or presence of any user that belongs to a decentralized database. In the following section, some differential privacy-based approaches are described.

In smart homes, data aggregation creates privacy-preserving challenges regarding how smart applications generate vast quantities of personal data for transfer to different service providers. In [31], a secure privacy-preserving architecture was proposed based on private Ethereum Blockchain to retain security among smart home participants and access control using smart contracts to authenticate access to IoT smart home devices. The authors declared that privacy can be preserved by employing a differential privacy machine learning algorithms to send private data to cloud computing. Merging the three paradigms Blockchain, edge computing, and Industrial Internet of Things creates kind of rapid growth in [32] proposed Blockchain-based Internet-of-Edge model for data sharing as a privacy-preserving, scalable, and controllable task allocation model while hindering adversaries from obtaining sensitive data access differential privacy is employed as a protection mechanism to protect edge node identities, the experiment efficiency of BIoE is implemented in Ethereum Blockchain. In [110], architecture-based blockchain data sharing using discrete M-band wavelet transform with differential privacy Laplace-Sigmoid noise was used in order to protect both patient identities and secure sensitive electronic health records EHRs; according to their design, the medical centers were allowed to store and share patient data as well as other EHRs with each other in order to train machine learning models while keeping the privacy of the patient identity protected without using smart contracts where blockchain is used as a secure database for storing the encrypted data.

- ***The anonymization approaches:*** Anonymization meets blockchain technology requirements in different fields such as IoT, healthcare, financial platforms, and vehicle networks. Anonymization started since blockchain inception via pseudonymization to fight the linkability between user identity and transaction data, and many solutions have been proposed in the literature for increasing anonymization by adopting K-anonymity that preserves sensitive data privacy by keeping the user's identity hidden in a given dataset, where l-diversity is an extension of K- anonymity by increasing the diversity of users' sensitive data in a given dataset, in order to greatly reach privacy. However, t-closeness enforces a large distance between the sensitive attributes in a dataset. In the followingsection, some of the anonymization approaches based on blockchain are described.

  A Blockchain-based location privacy-preserving mechanism was proposed in [33] where the use of the k-anonymity model for the sake of anonymizing users' actual location before

transmitting it to the network without requiring a server, another user 'privacy-preserving via leveraging multiple private blockchains through maintaining the service's quality. In order to overcome the issue of information management and validation in higher education, the work of [34] adopted the Hyperledger Fabric Blockchain along with heuristic K-anonymity for data generalizations where both confidentiality and privacy of the data enclosed in the nodes of the channel are achieved. In this research the NP problem is handled using a stochastic diffusion search optimization algorithm, which gives a better result for various parameters.

2.3.3 Integrating Blockchain in Artificial Intelligence to Preserve Privacy

Artificial intelligence has contributed for a couple of decades to making human lives easier and faster by generating machine model patterns that assist real applications for classification, prediction, optimization, or even making decisions. AI applications (machine or deep learning) require a massive amount of training data to obtain an accurately learned model, which requires data from several organizations. However, data availability is still the main challenge that acts as a barrier, where AI applications build on the last up-to-date viz the evolution of cancer, thereby calling to store and share data from the available, accessible, secure and decentralized network.

One of the decent solutions combining the two paradigms Blockchain and AI has witnessed an effective solution for securing data sharing among healthcare organizations. MedRec [130] is an MIT healthcare project that adopted Blockchain in regards to its decentralization and for secure management of medical data, in which the patients practice their ownership by providing full control over their health data. The role of AI was embodied in granting precious predictions according to patient data, which assists healthcare providers in enhancing their care services. Industry 4.0 stands on smart manufacturing in many fields, such as the food, agriculture, and textile industries, and when it meets blockchain transparency and traceability in smart agriculture [35], the market has prevailed and turned from Industry 4.0 to Industry 5.0, which collaborates between human efforts and robots working to test and create a self-driving vehicle for transportation. To promote trust between customers and producers, face recognition is employed to identify those who send goods. Blockchain properties captivating, such as decentralization and security, have also been investigated [36] to protect the ecosystem from pollution by tracking plastic waste for recycling activities to obtain a green supply chain where consumers have the right to use and access ledger information to make purchasing decisions. Machine learning and Auto-Regressive Integrated Moving Average have been used to automate the process and simplify the calculations. In the SIEMF [37] platform, an efficient combination of blockchain and deep learning to predict traffic incidents and assist the

evidence management for forensic analysis within the IoV network can be used at any time, in addition to the use of ABE to manage the access control of the data stored in blockchain that renders resistance to privacy and spoofing attacks.

In 2016, AI knew a new breed of machine learning that aims to train AI models collaboratively, locally, and privately, called federated learning, which was proposed by Google according to HIPAA and GDPR legislation standards that prevent sharing personal data and even predictive machine learning with third parties. The federated learning approach is relevant in addressing the need to share raw data to train a global machine learning model that can ensure data privacy and train a collaborative model from several data providers [42]. Federated Learning is advantageous in terms of privacy preservation, lower latency, alleviation of power consumption, and smarter models. However, it requires a trusted authority to avoid trainers dropping out of the training or sending poisoned messages to their partners [111]. Regarding the distribution as a common feature between federated learning and blockchain that gives rise to a new paradigm FLchain that provides a trust inter-participant, which enables them to train their datasets along with incentives mechanizems, similar to traditional federated learning; however, the aggregation process is conducted in a decentralized manner which preserves the privacy of data providers in different fields. In FLchain, there are two types of nodes: clients/participants and miners. Participants could have been devices that were selected according to their reputation behavior or energy consumption. Miners are Blockchain workers in charge of the mining process and append new blocks after reaching a specific consensus. The Flchain process can be described as follows:

- A computing task is published, its requirement is determined, and a committee of nodes is selected as a participant (e.g., Devices in IoT field). A new round was started after downloading the reference model (initial model) ,in which the participants updated the model using their local datasets.
- The participants send local updates as a transaction to blockchain miners to generate the global computation.
- Blockchain miners check the validity of the issued transactions; to this end they gather into a block to be mined after applying a specific consensus (e.g., resolve the mathematical computation puzzle (PoW) to decide the next appended block in the chain). The faster miner will be rewarded with tokens such as Bitcoin and, ether, and it propagates the new block throughout the peer-to-peer network.

- Using the data held by the new block, an aggregation process was launched to create a global modal update. This step can be done inwardly or outwardly through the chain, and the result is maintained in the blockchain.
- Finally, the participants download the new global updates to start a new round

In [38], FLchain was proposed as a crowdsourcing task to enhance its service quality by considering customers' opinions about home appliance manufacturers. The federated learning system was designed to train customers' data locally, which will help manufacturers predict customers' consumption behaviors and requirements in the future. After collecting the data from different home appliances, the customers train the reference model based on their local private data, and the outcome parameters are sent to the decentralized aggregator server, blockchain, after adding a formal privacy guarantee that perturbs the gradient using the differential privacy technique. Most EHRs recorded in the cloud server suffer from inward and outward attacks. The work of [39] introduced a secure and scalable privacy-preserving federated learning in smart healthcare integrated with a consortium blockchain that enabled an IoT cloud structure for secure storage. The main idea behind this is to train machine learning models in IoT devices at different locations without transferring the generated medical record to the cloud server unless the parameters of the trained models are used. Blockchain substituted the central aggregator in many FLchain approaches, as [40] proposed the use of a consortium blockchain as an underlying framework in the IoV system. After local training, the model gradient is improved using multi-Krum technology and encrypted with homomorphic encryption, which enables the verifiability of local models to achieve privacy preservation. The proposed architecture is simulated, and it demonstrates efficient and secure federated Learning in the IoV while preserving vehicle data privacy compared to typical centralized learning.

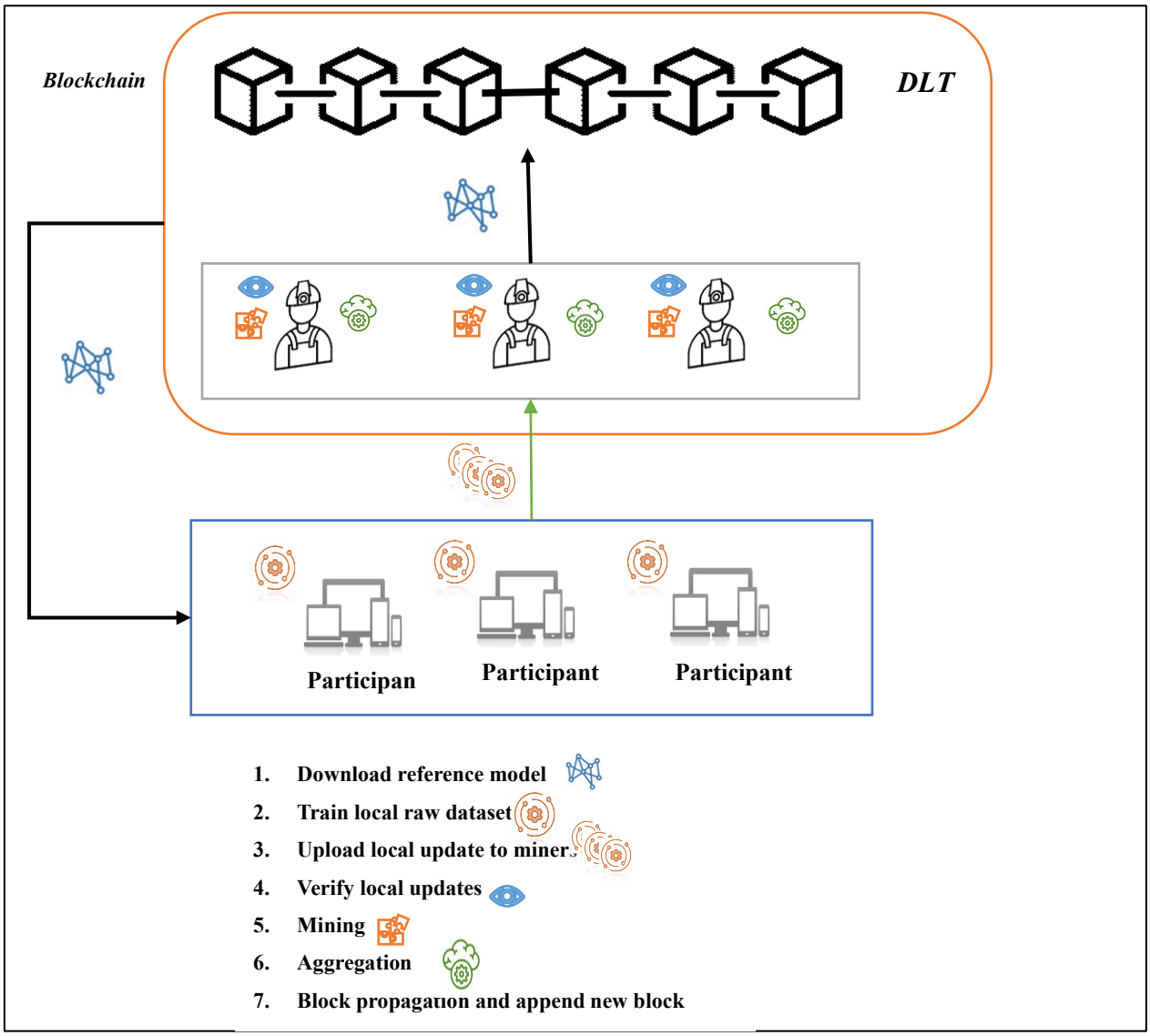

Figure 2.5 : Blockchain-Based Federated Learning Architecture FlChain

2.4 Discussion

From the comparison indexed in Table 2.2 of the blockchain privacy-preserving approaches based on cryptographic and non-cryptographic techniques, the survey discusses how the security and privacy of data sharing can be achieved and applied in different domains and technologies (cloud computing, IoT, IIoT). In comparison, several factors are taken into account to characterize the surveyed methods, including whether transaction privacy and identity privacy are adhered to, blockchain implementation, security and privacy requirements, and the field of application or technologies adopted. From the discussion, it is evident that the stated methods do not adhere to full data sharing privacy-preserving requirements after processing that marginalize the data service privacy among other services, which makes them affected by internal and external attacks on one hand and does not treat blockchain scalability.

Table 2.2 : Summary of privacy-preserving approaches based on Blockchain

Privacy-preserving Model		Ref	Transaction privacy	Data privacy	Blockchain platform	Security and privacy requirements	Technology/ themes
Cryptographic methods	Symmetric key encryption	[103]	Yes	Yes	Bitcoin	Integrity, Confidentiality	Finance
		[25]	Yes	Yes	Bitcoin	Non-repudiation	-
	Asymmetric key encryption	[104]	Yes	Yes	Permissioned	authentication, integrity	Cloud computing
		[26]	Yes	Yes	Public Ethereum	authentication, access control, integrity,	Virtual circuit
		[105]	No	Yes	Hyperledger fabric Permissioned	authentication, integrity, confidentiality, access control	Healthcare
	Identity-based encryption	[107]	Yes	Yes	Hyperledger Fabric permissioned	access control, integrity	IoT
		[27]	Yes	Yes	Ethereum	Authentication, integrity, access control	Internet of Medical Things IoMT
	Attribute-based encryption	[28]	Yes	Yes	-	Authentication, Integrity, confidentiality, non-repudiation, access control	Cloud computing
		[29]	Yes	Yes	Ethereum	Integrity, access control	Cloud computing
Non-cryptographic methods	Differential privacy	[31]	Yes	Yes	Private Ethereum	Authentication, access control, integrity	Smart home
		[32]	Yes	Yes	Ethereum	Confidentiality, access control	IOT
		[110]	Yes	Yes	Hyperledger Fabric permissioned	Control access, integrity	Healthcare
	Anonymization	[33]	Yes	Yes	Remix Ethereum	Integrity, confidentiality	Location-based service
		[34]	Yes	Yes	Hyperledger Fabric permissioned	Confidentiality	University management record
	Federated learning	[38]	Yes	Yes	Consortium	Integrity, confidentiality	Crowdsourcing IOT
		[39]	Yes	Yes	Consortium	Authenticity, confidentiality	Internet of Health Thing
		[40]	Yes	Yes	Consortium	Integrity	Internet of vehicle

To overcome such challenges and existing limitations, a system that manages the data services interactions and obtains full access control of the data owner is introduced, where the blockchain is used as an underlying entity to support access permission and ensure both authenticity and trust among service providers, along with deploying smart contracts that are written in specific code to verify owner identity, access permissions and data ‘integrity that satisfy security and privacy requirements along with respecting blockchain network scalability.

2.5 Chapter Summary

This chapter discusses the necessary preliminaries related to the security objectives and privacy techniques in general, the main concepts relevant to digging into blockchain technology, and the background that helps the reader understand our dissertation in addition to the security and privacy properties that are provided by blockchain technology. We present a blockchain privacy-preserving taxonomy based on cryptographic and non-cryptographic methods by surveying some of the proposed existing research solutions, and we discuss the main shortcomings in terms of privacy preservation.

Related Works

3.1 Chapter overview

This chapter discusses several research solutions that have been performed in three categories, namely privacy preserving data sharing based blockchain, privacy-preserving service composition, and federated learning privacy-preservation-based blockchain, along with their challenges and relationships with our thesis by highlighting their connections and differences with our contributions by putting forward a summary for each category and discussion.

3.2 Privacy-preserving data sharing based on Blockchain

Many research solutions based on the emerging technology blockchain have been applied in different fields to preserve privacy, which literally means the fusion of three concepts: security, confidentiality, and integrity. Preserving the privacy of an individual protects both the real identity and generated data until its destruction. To achieve this objective, numerous cryptographic methods and security approaches have been proposed, starting with Zyskind et al. [81], who were the first to propose a decentralized blockchain system that empowers mobile users to manage and control access to their shared data with a service provider, which cooperates with off-chain storage to eliminate data leakage.

In an analog direction, Truong et al. [41] put forward a platform based on permissioned BC for the management of personal data by checking and controlling access, authentication, and authorization, which must be GDPR compliant. This restricts data access for only the data subject and data controller, which allows control and log access to a resource server via smart contracts and blockchain.

In [112], an identity management prototype-based BC was proposed. The design integrates off-chain storage to save patient data at work. The system consisted of five components. The fundamental one is the authorization and authentication server, which plays the role of mediator between the application server and the BC. Furthermore, the server allows identity checking from the database. Liang et al. [42] proposed DronChain, a distributed solution that secures collection and even communication via drones and saves data in conventional cloud technology. However, to ensure security and integrity, the generated data hash is stored within the blockchain to safely access data.

Al Omar et al. proposed MediBchain [105], a platform based on BC for data management and ensuring data integrity in the healthcare system. A user-centric model that allows only the user to control his or her data was proposed. Each patient has an identity (ID) that refers to his or her pseudonym, and by using this ID, smart contracts obtain the ID of the corresponding block.

In [107] Sash, a data-sharing framework is proposed that brings together the blockchain technology and IOT platform, in which Hyperledger Fabric Blockchain was employed to safeguard access control policies and make access control decisions jointly with identity-based cryptography and audit all operations.

In [31], a secure privacy-preserving architecture was proposed based on private Ethereum Blockchain to retain security among smart home participants and access control using smart contracts to authenticate access to IoT smart home devices. The authors declared that privacy can be preserved by employing differential privacy machine-learning algorithms to send private data to cloud computing.

Table 3.1 : Summary of privacy-preserving data-sharing approaches based on Blockchain

Method	**Privacy-preserving technique**	**Blockchain openness**	**Privacy**	**Access control**	**Off-chain storage**
[81]	Asymmetric key cryptography	Public Bitcoin	User	Policy	Yes
[41]	Asymmetric key cryptography	Private Hyperledger	User	Policy	Yes
[42]	Hash function	Private Hyperledger	Service	Hash+policy	Yes
[105]	Asymmetric key cryptography	Private Hyperledger	Use	ID	No
[107]	identity-based cryptography	Private Hyperledger	User	Policy+IDE	Yes
[31]	differential privacy	Private Ethereum	User	Policy	Yes

All the aforementioned solutions are fostering blockchain technology in order to protect privacy in different areas using various privacy-preserving techniques, which is an undeniable fact that are remarkable and significant solutions that adhere to user privacy in Table 3.1. Several comparison factors are taken into account to characterize the surveyed methods that include privacy-preserving techniques, blockchain implementation, privacy of which entity (user or service), access method adopted, dependency based on only permission or other criteria and finally the generated data are stored on-chain or off-chain.

However, some of the proposed approaches marginalize controlling service interaction, which is a crucial part of managing and controlling data sharing; therefore, taking over the process of accessing data and pinpointing malicious behaviors. As a result, we can exactly target the problem when a service shares data with other services without permission from the owner, the service owner's privacy may be threatened and even breached, this problem started off owing to a lack of a global view of service control. Our solution addresses this issue by introducing a system-based service dependency graph for managing, controlling, and even logging service interactions. Blockchain technology can be adopted as a substantial access control mechanism to authenticate system services and maintain data integrity, along with the deployment of smart contracts that fit system requirements and stated concerns.

3.3 Privacy-Preserving Data Sharing in Service Composition

This category is bifurcated into two underlying branches: blockchain integration and privacy preservation in service composition. In the following section, we discuss each of these in detail.

3.3.1 Blockchain Integration in Decentralized Service

One of the benefits of applying blockchain in service composition is decentralization, which is a fascinating property that has led several academic researchers to investigate. In [43,44], there is evidence of cost saving and time-saving beyond the use of blockchain to tackle centralization concerns, not only the security, scalability, and transparency properties.

In [43], the selection of an optimal service composition was proposed for a decentralized environment through particle swarm optimization consensus to handle sophisticated cloud manufacturing tasks. Similarly, in [44], a decentralized platform-based blockchain was proposed for cloud manufacturing service composition. The mining process has incentivized miners by rewarding them when they proposed an optimal service composition solution, and a consensus proof of optimality has been put forward for the sake of building transactions and blocks.

In [113], blockchain was adopted as a decentralized cloud solution for Internet of Things services, which met software-defined networks (SDNs) and fog computing to compose complex services without requiring any intermediary. The composition process uses a reinforcement-learning technique to build secure and reliable paths. Moreover, Blockchain guarantees trust among the parties under the same framework. Furthermore, in [45], an automated business process management system was proposed based on blockchain to realize both processes, the selection and composition of services in an open business environment, and the work has proved that blockchain technology is cost saving in multiple aspects.

3.3.2 Privacy Preserving in Service Composition

Numerous typical and recent studies have attempted to address the concern of user privacy in service composition, emphasizing that the user's sensitive information was not disclosed to an unauthorized third party. In [46], a practical multi-source data integration approach was proposed, which tackles privacy concerns that appear when a composition is executed to prevent leakage among service providers or toinfer data about users. They applied K-protection to protect critical data between service providers. As the mediator is an untrusting entity, the authors used OPES to encrypt the exchanged numerical data.

Similarly, K-protection was used by [47] to guarantee a privacy-preserving in service composition that counts on a mediator. However, it is considered untrustworthy. Service providers authenticate each other using a secret sharing string between the parent and child services that belong to the same plan. This solution suggests that the main functionalities of the mediator are creating and sharing the plan with all service providers that have partaken. A mapping table was proposed and used by the mediator in the selection process. Despite the query being executed by the mediator, which relies on hashed critical data as input, the mediator is blind. The authors did not state how the mediator dealt with the final results. Furthermore, in [48], a privacy model that enables a service to define a privacy policy that specifies the set of privacy practices for any collected data and a set of privacy requirements that specify the set of privacy conditions that a third party must follow to consume the service's data. Moreover, the composition plan is defined under a negotiation mechanism to achieve compatibility with the services concerned. The mediator is aware of the exchanged data between the service consumer and producer.

Table 3.2 : Summary of privacy-preserving in service composition

Approach	Blockchain based	Decentralization	Privacy between service provider	Data integrity	Privacy preserving technique	Service composition plan	Degree of mediator' trust
[46]	No	Centralized	Yes	No	K-anonymity	shared	Low
[47]	No	Centralized	Yes	Yes	Hash+k-anonymity	shared	Low
[48]	No	Centralized	No	No	None	shared	Very high

From Table 3.2, we examined multiple vital factors, including decentralization, keeping the confidentiality between service providers denoted by "Yes" or "No," the data integrity is realized or not, which method employed for preserving privacy, the visibility of the service composition plan, whether shared or hidden and finally we evaluated the trust of the third party named mediator.

From the discussion, existing solutions suffer from numerous shortcomings that do not guarantee fully secure and private protection. The majority fostered a centralized environment that was effective and accurate in terms of performance. However, they are prone to a single-point-of-failure attack. The main limitation is that a mediator is required to generate the service composition and manage the query execution. Despite the mediator's trustworthiness, there is a potential probability of leverage from this liability and tampering with the composition plan quality by selecting specific service providers. Moreover, in most of the studies, the plan is shared in which the participating service providers have learned about the generated one, which enforces putting restrictions for authentication and maintaining privacy by using K-anonymity, which could have a negative impact in terms of scalability by returning additional unnecessary values by the query and it is insufficient to provide full data privacy protection in thigh datasets owing to lack of data diversity. Most stated solutions exchange subquery results through a mediator. In this case, if the data were not secured, it would be easy to breach theparticipant data. Otherwise, if the exchanged data are secure and protected, such as byusing K-protection, the mediator can deal with the child service provider by sending raw data (without using K-protection). Concerning the works that have integrated blockchain for decentralization and transparency, they have leveraged the security requirement of blockchain as a trust-distributed database. However, they ignored the privacy requirement, which

leads to harmful abuse and consequences for user and service personal data. Another limitation is that data subquery results are exchanged through a mediator. In such an example, if the data are not secured using an efficient method, it is trivial to breach the participant data. In another case, when the data are completely secure and anonymized using K-protection, the mediator can deal with malicious service provider participants by sending the original data (without using K-protection).

Our solution overcomes all the aforementioned limitations in the existing solutions, starting with decentralization by integrating trust authority with blockchain technology as an underlying entity that plays the role of trust mediator and fosters service composition plan generation and even authenticates legitimate participating and unauthenticated intruders by managing data access control. To promote both the security and privacy requirements, our design is mixed with asymmetric key cryptography and uses a hash function. In addition, we use IPFS secure and decentralized off-chain storage to overcome the scalability concerns in the existing approaches.

3.4 Enhancing Federated Learning Privacy and Scalability Based on Blockchain

Federated learning is a new type of machine learning that has been proposed to deal with Google client privacy by locally training their data. However, a centralized federated learning system suffers from several issues and challenges in terms of client data sharing and privacy preservation, which require a decentralized and trustworthy authority. Many studies have been put forward for the incorporation of blockchain that empowered federated learning in data sharing, and the relevant existing studies are bifurcated into two branches: blockchain meets federated learning for privacy-preserving and federated learning-based blockchain sharding for privacy-preserving data sharing.

3.4.1 Blockchain Meets Federated Learning for Data Sharing Privacy Preserving

The Flchain paradigm has prevailed in several domains regarding the security and privacy provided in the healthcare field [49], and proposed a privacy-preserving solution to handle the poisoning attack by checking the validity of the local machine learning model from the client side using an encrypted inference method and then eliminating the poisoned model. To ensure secure aggregation, the process is performed on-chain to upload only a secure local model after applying the secure multiparty computation method to protect the privacy of the model parameters.

In the same endeavor, the work of [50] proposed a traffic flow prediction privacy preservation based on consortium blockchain, and this approach achieved two objectives. The first is to retain the ledger by pre-defining a limited number of reliable miners, whereas the second is to enhance the federated learning performance by eliminating poisoning attacks. The crucial idea revolved around each iteration, each miner adopted a predefined filtering strategy for purifying the network from malicious entities that caused the failures or the poisoning attack using the dBFT consensus. Another merit of diminishing internal attacks is that the user is able to pick up miners to keep the aggregation process under control. To protect local updates of vehicle data from privacy attacks, a local differential privacy method is employed.

Furthermore, in the Internet of Medical Things, the work in [51] maintains privacy and ascalable federated learning model based on asecure and private blockchain architecture in IoT cloud storage that allows both data sharing and model training to be secure. Moreover, the work of [52] proposed a privacy-preserving framework in IoT, where acustomized user context for predicting the reliability of the blockchain system based on federated learning, where the proposed study goal is to hide user information and jointly collaborate with the central server for training a global model using Federated Learning Neural Collaborative Filtering, for the sake of personalizing blockchain reliability prediction, as a consequence select the more reliable Blockchain peers to deal with.

3.4.2 Federated learning-based Blockchain sharding for data sharing privacy-preserving.

The ever-increasing amount of big data generated in different domains causes an increase in communication overhead security and privacy disclosures. Combining Blockchain and federated learning presents huge challenges in terms of learning convergence and scalability issues with regard to the increased number of federated learning clients or blockchain peers. The solution to such a challenge is to deal with blockchain scalability and adopt one of the scalability techniques, such as the sharding of the distributed network into a subnetwork to hinder malicious participants and decrease the high communication overheads.

For this purpose, ChainFL [53] was proposed as an asynchronous federated learning combined with a two-layer blockchain, a mainchain and a sub-chain, using consortium Hyperledger Fabric as a private network, in addition to adopting the Raft consensus inter-shards. The proposed solution scales the blockchain network by customizing the sub-chain layer for IoT network committees(shards), which limits thecommunication overheads. However, the mainchain layer

employs a Direct Acyclic Graph to achieve parallel and asynchronous cross-shard validation to thwart the stale model problem.

Similarly, in ScaleSFL [114], a sharding technique partitions the blockchain network in a federated learning solution, in which the proposed framework supports interoperability by dispersing the off-chain federated learning process, to check the model update validity rather than controlling the entire federated learning. The proposed solution is supported by a mainchain for collecting and coordinating the received aggregated models from the shards. However, the sub-chain maintains individual shared shards data.

Furthermore, in the smart grid, the work of [115] proposed federated learning-based Blockchain sharding for decentralized voltage stability utilizing smart contracts, along with two layers of blockchain in order to improve the system's scalability, in addition to a fixed number of computing resources. The proposed prototype is tested on the Hyperledger Fabric platform, referring to the evaluation results, and the system performance increases steadily with the shard 'number.

From Table 3.3, the two paradigms, federated learning and blockchain, are combined to form a Flchain architecture. However, it bifurcated into two branches. The first illustrates how to preserve privacy using federated learning in different fields and various privacy techniques to protect local update transfers. However, the second branch focuses on tackling the issue of blockchain scalability and alleviating communication costs using scalability techniques. Based on the summary, we examined various important factors that include the distributed ledger, whether blockchain data structure or directed acyclic graph data structure, and then we pinpointed the type of ledger synchronization relevant to the data structure and characterized each approach regarding the openness of the distributed ledger along with the consensus adopted. Model update privacy denotes the privacy method employed to protect local updates before aggregation. The security factor discusses the attacks that were defeated by each approach.

Table 3.3: Summary of federated learning privacy preserving based Blockchain solutions

Approaches	**Data structure**	**Data Asynchrony**	**Openness**	**Consensus**	**scalability technique**	**Model update privacy**	**Security**
[49]	BC	synchronous	Private	Proof of work	-	Secure multiparty computation	Poisoning attack
[50]	BC	synchronous	Private	Delegated Byzantine	-	differential privacy	Poisoning attack

				Fault Tolerance			
[51]	BC	Synchronous	-	-	Poisson distribution	secure multiparty computation	Poisoning attack
[53]	BC+DAG	Asynchronous	Private	Raft/PBFT	sharding	Not mention	Poisoning, Backdoor attack
[114]	BC	Synchronous	Private	Raft	sharding	differential privacy	Poisoning, Backdoor attack
[115]	BC	Synchronous	Public	Proof of Knowledge	sharding	differential privacy	Double spending attack, Dishonest Behaviors

Undoubtedly, these solutions adhere to security and privacy protection using federated learning combined with a distributed ledger to decentralize the process. However, the majority fosters off-chain storage to alleviate the pressure of the ledger by maintaining the hash of the parameters on-chain, and the model data are kept secure off-chain, this scheme is no longer effective if the number of blockchain nodes increases, which affects the blockchain transaction throughput and latency. Otherwise, some of the existing solutions handle the issue of scalability by adopting the sharding technique, which demonstrates its efficiency with large traditional databases; when it is conveyed to distributed blockchain according to the performance results, the system throughput increases steadily with the increase in shard numbers. Therefore, most solutions adopt a static shard, which is, per se, a vulnerability in the system that requires a purifying round from compromised and malicious nodes. Another challenge that may face scalable or non-scalable Flchains is proving the model's ownership and protecting it from intellectual theft.

To overcome the limitations of existing approaches by introducing federated learning data sharing privacy preservation based on sharding blockchain, in which the sharding technique has to be dynamic, which serves for each training round, the shard participant are changed dynamically. However, the use of IPFS as secure storage for maintaining model data updates is secure and tamper-proof. Blockchain technology is an underlying component that manages data authentication and confidentiality by creating a private network (permissioned). The proof of ownership of the learned model is a controversial challenge that can be addressed by generating a non-fungible token after the federated learning task is completed.

3.5 Chapter Summary

This chapter covers the relevant existing solutions to our thesis in two paradigms: *direct data* and *indirect data sharing*. Direct data sharing is when the system component inquiries about raw data however indirect data are inferred data, such as artificial intelligence model parameters. The related works fall into three categories: privacy-preserving data sharing based blockchain in service-oriented architecture per se, which is bifurcated into two branches: privacy-preserving data sharing inter-atomic services based blockchain and privacy-preserving data sharing in service composition. However, the third category discusses existing approaches when blockchain meets federated learning for data sharing, privacy preservation and scalability. The chapter characterizes each method and highlights a summary comparison along with the challenges and issues relevant to each category. To this end, in the following chapters, we provide solutions to the aforementioned challenging problems.

SDGchain: When Blockchain meets Service Dependency graph to Enhance Privacy

4.1 Chapter overview

In the previous chapters, the motivation, background and related works probed in this thesis were discussed. The objectives of Chapter 1, which are associated with blockchain-based privacy preservation in service-oriented architectures, are also discussed. According to the first objective, a privacy-preserving model based on the blockchain for data sharing is required. Therefore, this chapter discusses how service dependency graph enhances privacy preservation based on permissioned blockchain (Hyperledger fabric), which is known as SDGchain for independent services. Section 4.2 embraces the introduction to the main objective. Section 4.3 comes after the corresponding algorithms for the proposed SDGchain, with a scenario to elucidate the work presented on the stated topic. This section also includes a performance evaluation of theSDGchain along with the required configurations. Moreover, a security analysis is conducted by comparing the SDGchain and the relevant solutions. Finally, Section 3.5 sums up this chapter.

4.2 Introduction

Owing to technological advancement and service evolvement since the inception of the Internet, individuals' data have been massively collected through numerous web applications. In addition, the daily use of social media platforms allows the daily collection of personal data. According to a global digital overview report in 2024 [126], more than five billion users are creating and exchanging their data daily, which increases the risk of violation. Thus, owing to the service's ubiquitous colossal benefit upon easily and rapidly performing tasks, there are growing concerns and challenges regarding individual data privacy protection because they have poor control over how to collect the data and to what end, without declaring the legitimacy of the process, even though maintaining the same privacy standards even among other services. The service claims to adopt a strategy to eliminate the collection of data without authorization and/or sharing data that can be wrong [54] would lead to developing methods and means that grant user empowerment to track and undertake control over the processing of their own personal data.

In 2018, the introduction of the General Data Protection Regulation (GDPR) indicated that personal data "can only be gathered legally, under strict conditions, for a legitimate purpose" [73] in such a way that full control over data has been removed from the service provider to fulfill theuser-centric control model intended to overcome privacy challenges and issues. Centralized structures face privacy issues to ensure both confidentiality and control, which remains a big challenge, as individuals have somewhat or even no control over their personal data [22].

The emerging technology blockchain is put forward as a distributed and secured database that was born with Bitcoin, which paves the way for many researchers to invest its adoption from the financial sector to other domains such as IoT, e-healthcare, smart grids, smart cities, industrial IoT and even intelligent Connected Vehicles, regarding its integrity, immutability, confidentiality, decentralization and traceability. Moreover, owing to its security and attractive features, blockchain has been utilized as a mechanism to preserve user privacy [81]. According to Blockchain taxonomy, within a permissionless blockchain, the data and verification process are transparent to all nodes, which calls for privacy concerns, low efficiency and low transaction rate, which leads to a suspect 51% attack [56]. Regardless of privacy breaches in the public environment, the user's transaction may be tracked by adversaries by linking her address and analyzing the transaction rules, as a result, the real identity can be inferred using external information of the network [116]. Otherwise, permissioned blockchain helps maintain privacy by defining authorized nodes that only work and access the data.

In the service-oriented architecture (SOA) model, services are a self-contained, loosely coupled, and independent entity that stands for a unit of software functionality or aset of functionalities designed to realize a desired task, which makes them available over a network to be invoked through defined interfaces and merging them into business solutions [57].In the service provider-centric model, accomplishing user-specific tasks requires the involvement of multiple services, which results in accessing, processing and even sharing user personal data with unauthorized parties without permission, because the user has no control over its data. Therefore, it is unaware of its data destiny, which can lead to privacy breaches. Owing to loosely coupled features, services communicate independently as service providers of data and consumers. User privacy is affected when the invoked service's privacy is affected, as well as in service interactions. A service owner's privacy violation may occur where its data may be shared with unauthorized and malicious services, which leads to the inference of user-sensitive data, that calls for maintaining privacy even during data processing. In other words, the service requester must follow the submitted rules before and after data processing to preserve the privacy of the service owner. Consequently, these interactions cause a challenge

among services, which requires defining a privacy strategy that assists the user in holding secure control over their personal data. This concern elaborates on how to maintain the privacy of atomic services while working in a similar workflow.

To confront the issues stated above, we integrate a permissioned Blockchain such as Hyperledger fabric which plays the role of a trust data server and, even more, a trusted authority (trust third party). As per the proposed design use cases were proposed to preserve privacy in the SOA environment, and we targeted a model to hold entire control over the interaction among services using a service dependency graph based on blockchain technology. The aforementioned solution cooperates with off-chain storage, which improves system scalability. Among Blockchain functionalities, it is considered a mechanism for authentication by employing public key cryptography, which assists in controlling service data interactions and then grants an auditable history for all operations that occur, which would help to detect vulnerabilities. The main contribution of this chapter are as follows:

We propose a framework to protect data sharing privacy in a decentralized architecture called SDGchain, drawing on a secure service dependency graph (SDG) and combining it with permissioned blockchain viz Hyperledger fabric. In addition, the design boosts the security and scalability of off-chain storage. The service dependency graph contributes to gaining control of service interactions, measuring the trust level, and calculating the quality of service. All of these functionalities are realized by building dependencies that ensure privacy preservation of the service.

4.3 Blockchain enhances privacy inter-service using service dependency graph (SDG)

Blockchain technology has proved its efficiency in maintaining data integrity and application transparency when it comes to confidentiality, authentication, and especially service ‘access control. These features have evolved owing to their distributed nature, which contributes significantly to decreasing malicious attacks. In this section, SDGchain is the proposed design, and is presented in detail. Service interaction in an SOA context based on a blockchain is illustrated in the figure 4.1. Moreover, this section discusses the system model, a scenario for authorization and service access control and how to select a service according to computing a measuring formula, all of which are illustrated in detail, along with their algorithms.

- ***Definition of Service Dependency Graph*:** *SDG* is an oriented graph defined as $G = \{S, E\}$, where $S = \{S_1, S_2, S_3, \dots, S_N\}$represents a set of services that work in similar conditions, E is

a set of dependencies between services where every dependency links two services and every edge is weighted by attributes $\{a_1, a_2, \dots, a_n\}$,

We can say that S_j depends S_i implies $E_{ij} = \{S_i, S_j\} = \{a_i, a_j, \dots, a_l\}$. The SDG renders flexible control; indeed, it helps to guarantee system privacy for both user and service actors. Every service is distinguished by a set of incomes and outcome edges that allow the dependencies that drive the graph to manage and control every interaction for instance requests for specific attributes from the service owner of the attributes to the service requester. Those with permission can access data to avoid distrusted services.

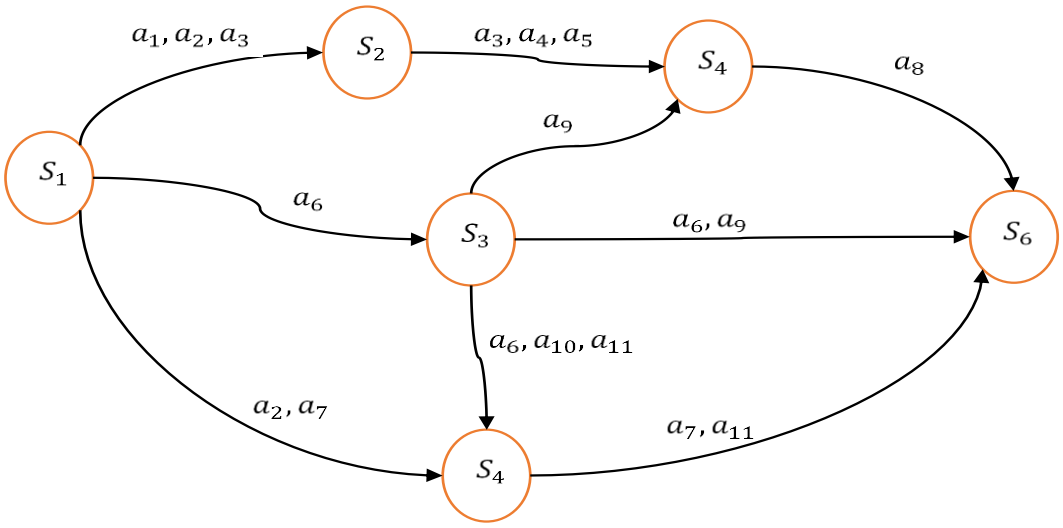

Figure 4.1: An example of a service dependency graph that contains six services

Figure 4.1 illustrates service interaction in the SDG. $S_1$and S_2 are dependent (after S_1 permission) since the S_1'output attributes a_1, a_2, a_3 are S_2 inputs. Building upon SDG, the attribute requester S_2 does not have the right to grant permission without referring to the real owner S_1 thus, in this case, the owner's privacy will be affected; however, especially when the attribute is considered sensitive, so S_2 must request another permission.

The most important advantage of using such a graph is that it enables the system controller to determine an attack spot, which results in mitigating propagation to other nodes. The SDG is used not only for controlling, however, it could be a measuring tool when it comes to reducing the negative impact of vulnerabilities. In suspicious cases, it calls to cut the dependencies as an initial

solution, and then, it leads to selecting the best countermeasure [58], measuring the risk assessment [59] and even preserving privacy in web service composition [60].

4.3.1 Service Dependency Graph Preserves Privacy

The proposed method is discussed in detail in this section. Furthermore, the system design, illustration scenario, and proposed algorithms are discussed for securing theSDG.

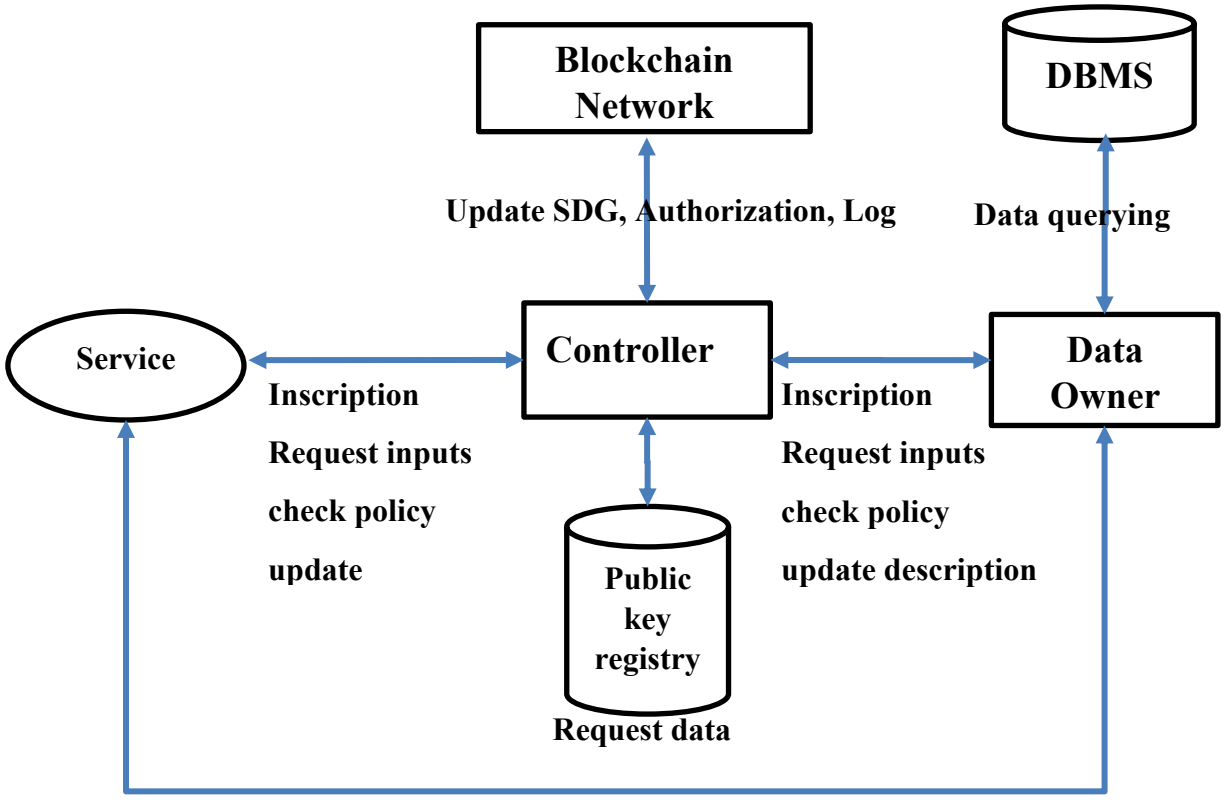

Figure 4.2: SDGchain Desing and its main components

4.3.1.1 SDGchain System Design

The proposed design is illustrated in Fig 4.2. where the model includes seven fundamental components that interact with each other and the service is an active entity. The architecture is empowered to cover all its interactions with other entities of the system, especially when the service demands data for a specific permission. Thus, we illustrate how to put these interactions into terms, building upon a secure dependency graph that empowers to provide a comprehensive view of system data sharing. In the following section, a detailed explanation of the principal elements is proposed.

- ***Controller entity*:** This is a network node called a manager entity, among the main functionalities is which manages data access control by checking requests. It is in charge of establishing the dependency graph after assessing a set of services in order to pick solely the best dependencies, and it is reliable to register a new entity by assigning a key pair, which enables entity authentication to be checked by drawing on a recorded public-key dataset.

- ***Service*:** An entity furnished by stakeholders, service providers or third parties to process user data and obtain results in aspecific field. According to SDGchain design the service deals with users and other services that would allow the treatment of data.
- ***Blockchain***: A serverless distributed database that is mainly adopted to maintain data confidentiality and integrity through access control and identity verification. In this design, blockchain is the core component that includes three ledgers: the first one has the role of storing data integrity and authentication where it stands for Autledg, while the second ledger saves the graph information named SDGledg, and finally maintains a log of all the operation in which Logledg is reliable.
- ***Off-chain storage***: It is used along with blockchain as a secured database with the aim of alleviating the block size by maintaining the personal data ciphertext, and solely returning a ciphered pointer to the data owner (MongoDB) that is recorded in blockchain.

Regarding the system overview, the service owner is the only one responsible for its data, so it delegates the controller to trail all operations, such as access requests, sharing data with third parties, modifying or even deleting, and SDGchain grants the opportunity to log all metadata in the distributed ledger.

4.3.1.2 Identity Management

The privacy-preserving framework SDGchain is valid for use by both users and services, where each interactive entity in the system is assigned to a digital identity, before interacting, a registration step is triggered and associates the generated key pair (pk_i, sk_i) for each entity in order to hide the real identity and validate the authenticity. The corresponding public key pk_i is fostered as a pseudonym. An asymmetric RSA algorithm was used to generate the tuple $(pk_i, sk_i)$where all the public keys are stored in a dataset, however, each entity secretly and securely retains its private key sk_i. To validate the authentication of the legal system member, the controller launches a signature-verifying algorithm (Eq 4.1).

$$V\left(S_{sk_i}(m_i, pk_i)\right) = [m_i, pk_i] \qquad (4.1)$$

To secure critical data that represents critical attributes, they are encrypted using the data owner public key then the outcome will be kept in a secure off-chain storage, after authentication validity and granting access permission the owner would decrypt the request data utilizing his own secret key.

4.3.1.3 Service Dependency Graph Security

The working of the SDGchain design is based on establishing a weighted service dependency graph by building the dependencies among services that have a direct relation to shared attributes. We can deduce another graph stands on permission dependency, where the idea behind is to set a graph using access permission of the shared attributes [81], [41], which is seen as an outcome of granting the service access attribute, and it is more likely to derive the graph only from Autledg. Consequently, an adversary can infer and extract sensitive information by linking different snapshots of the ledger that could succeed in creating the dependencies, only if we settle for and count on the anonymization mechanism that would lead to exposing the real identity of the data owner. Consequently, the two types of graphs are secured using graph modifications such as anonymization, edge and node modifications, where the goal is to disturb the dependency graphs, which prevents intruders from extracting sensitive information. Modifying the original graph was first proposed in [61], where the idea behind graph modification is to maintain entity privacy. This technique is distinguished by the irreversibility property that renders restoring the original graph likely impossible. Otherwise, the SDGchain controller is empowered to retrieve the edges and nodes pre-modification, and then it uses a reversible graph modification as follows:

- ***Anonymized Node***: Every service/attribute is associated with an identifier. The controller can retrieve the real service by issuing a query to the blockchain, including the corresponding ID.
- ***Modified attribute***: the controller works by hiding the plain attribute name (dependency). An encrypted algorithm was employed.
- ***Modified edge:*** The controller works by disguising the real output edges of each node by cutting the outgoing neighbor dependencies by adopting a reversible technique.

4.3.1.4 Access Control and Authorization Management

This section highlights the momentous scenario that is launched to access requests for attributes and then obtaining sensitive data along with the main algorithms that elaborates onhow the process is conducted.

Algorithm 4.1: *RequestAttributes*

Input: Identity pk_{Si} , Signature sg_{Si}, Operation op, Attributes attr;

Output: $dataId_{(attr,pk_{Sj})}$;

Init: Set of services s=null;

if ((registered (pk_{si}) **and** (verify (pk_{Si}, sg_{Si})) **then**

depLedger← getChannel("SDGledg");

AuthLedger← getChannel("AuthLedger");

logLedger← getChannel("logLedger");

$s \leftarrow$ depLedger.queryMatch(attr);

$dataId_{(attr,pk_{Sj})} \leftarrow Max\{quality(pk_{Sj}) | pk_{Sj}\}$

If ($op \notin AuthLedger.defaultPolicy(dataId_{(attr,pk_{Sj})})$) **then**

If (ownerOf($dataId_{(attr,pk_{Sj})}$) is pk_{Sj} **then**

requestPermission($dataId_{(attr,pk_{Sj})}, pk_{Sj}$);

else if (ownerOf($dataId_{(attr,pk_{Sj})}$) is pk_{Si} **then**

requestPermission($dataId_{(attr,pk_{Sj})}, pk_{Si}$);

else if (ownerOf($dataId_{(attr,pk_{Sj})}$) is pk_{Sj} and pk_{Si} **then**

requestPermission($dataId_{(attr,pk_{Sj})}, pk_{Sj}, pk_{Si}$);

$dataId_{(attr,pk_{Sj})} \leftarrow$ null;

Else if($trust(pk_{Sj}) > Threshold$)**then**

depledger.updateDependency($pk_{Sj}, dataId_{(attr,pk_{Sj})}, pk_{Si}$);

Authledger.updatePolicy($dataId_{(attr,pk_{Sj})}, op, pk_{Si}$);

logledger.updateLog ($pk_{Sj}, dataId_{(attrpk_{Sj})}, op, pk_{Si}, curTm,$"$grant\ per$");

end else

end if

Return $dataId_{(attr,pk_{Sj})}$;

A. Scenarios and proposed algorithms

To obtain the desired data, the service requester starts by issuing a request to the data service owner, which requires executing two possible scenarios: the first requesting a specific attribute and the other

requesting data. Algorithm 4.1 demonstrates the first scenario launched by a requester by providing the following information (requester pk_{Si} , signature sg_{Si}, permission needed op and attribute name $attr$) in the form of a transaction. Once the authentication process is valid (4), the controller interferes by querying SDGledg to gain a result as a list of all services that hold the requested attributes (5-7). Next, when the result is more than one, the controller selects the best data ID by running *equation 4.2* (8). If the permission needed does not belong to the data default policy, the controller immediately orient the request to the data owner (9-13). In contrast, updating the ledgers (15-17) and then assessing the requester trust using *equation 4.3,* then replying by the data ID (20). Currently, the service requester has the attributes ID andit can request at any moment for the corresponding data attributes. Before garnering the plain data attribute, S_i queries the controller. Figure 4.3 illustrates a sequence diagram that demonstrates the necessary steps for the request for the data process.

- ***Step 1:*** Service S_i requests the controller when it receives its corresponding attribute ID.
- ***Step 2:*** Before producing the attribute plain data, the controller checks the existence of the dependence between services by querying the blockchain to access the SDGledger and then Autledg to verify the permission.
- ***Step 3:*** When the result is a positive, S_i has the right to access the data; therefore, the controller asks attribute data owner S_j to transmit the raw data.
- ***Step 4:*** S_j service issues a query to off-chain storage utilizing the decrypted pointer (dataId) En_p_j, then the database responds to S_j with the encrypted data.

Step 5: Finally, the desired data is decrypted by owner S_j employing its private key; here, the requester S_i responds.

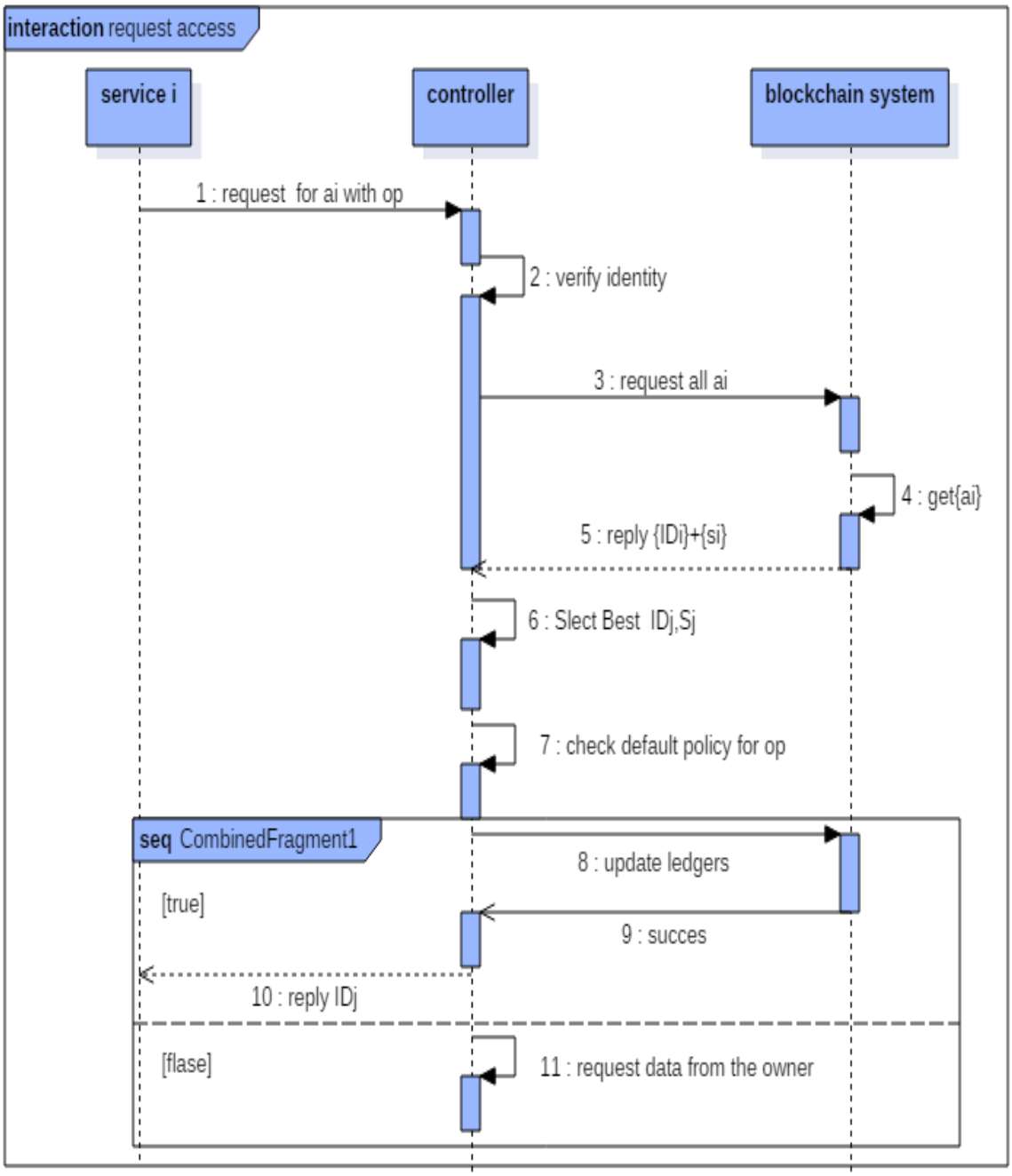

Figure 4.3: Request access permission to personal data

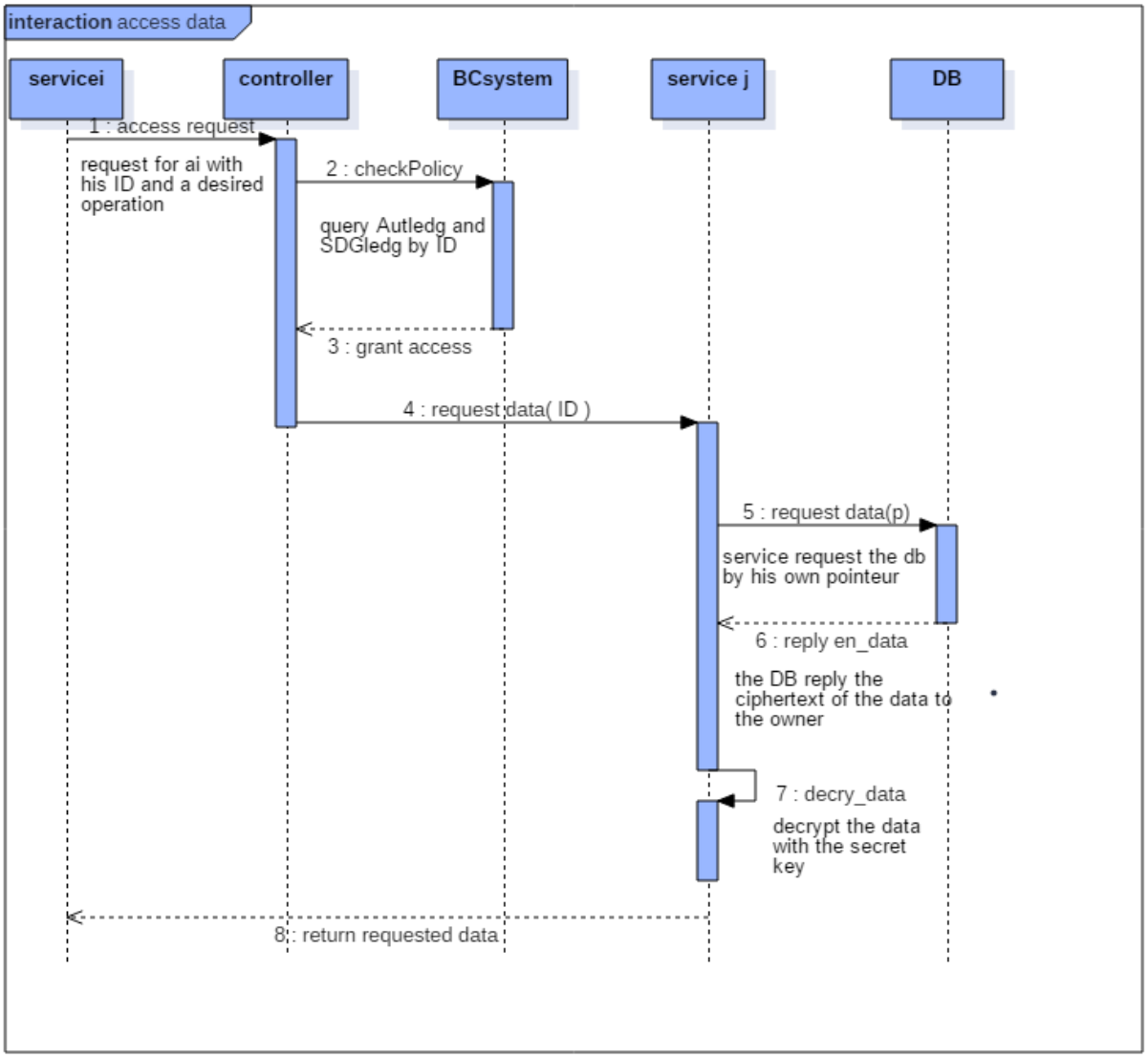

Figure 4.4 : Grant access to personal data

Because the service is an independent entity, it has the right to build, revoke, and remove dependencies. Algorithm 4.2 illustrates the removal process. It starts when the service owner is willing to remove a specific permission. In this case, the service owner provides the controller with the identity, signature, data ID, permission, and service target. The controller must authenticate requester (4). It removes the service and target permissions from the set of policies of the target data (9,8). When the target data have no other permission for the service target (10), the latter is removed from the target data dependencies and updated in the SDGledg (11-13). Finally, all ledgers are updated.

Algorithm 4.2: *RemovePermission*

Input: Identity pk_{Si} , Signature sg_{Si}, Operation op, Identity pk_{Sj}, data identity $dataID$

Output: boolean $result$ ← false;

Init: dataIdPolicy←null, $dataIdDep$ ←null;
if ((registered (pk_{si}) **and** (verify (pk_{Si}, sg_{Si})) **then**
    depLedger← getChannel("SDGledger");
    AuthLedger← getChannel("AuthLedger");
    logLedger← getChannel("logLedger");
    dataIdPolicy ← Authledger.getDataPolicy ($dataId$);
    dataIdPolicy.getTarget (pk_{Sj}).removePermission(op);
    If (dataIdPolicy.getTarget(pk_{Sj}) is empty **then**
        $dataIdDep$ ← depLedger.getDataDependencies(pk_{Si}, $dataId$);
        $dataIdDep$.remove(pk_{sj}) ;
        depledger.updateDependency (pk_{si},$dataIdDep$,$dataID$);
    end If
    authledger.updatePolicy ($dataID$, $dataIdPolicy$);
    logledger.updateLog (pk_{Sj}, $dataId$, op, pk_{Si}, curTm, "remove per");
    result ← true;
end if
Return result

B. Attribute Selection Method

In a service-oriented architecture, the service is a distinctive atomic entity regarding a set of features that determines its behavior in the execution environment, which is the foremost reliability and availability [117]. Instead of assessing the performance of each service according to these two features, it is better to render the evaluation relevant to service dependency graph updates that assist in boosting the finest overview of service trust and quality to facilitate the controller assessment. Only the negative impact is taken into consideration owing to changes and updates in the graph. Building on the SDG graph dependencies, the controller can determine the best service by calculating two metrics: quality QS and trust TS. They are invoked if the service asks for specific attributes and the outcome more than one or for the sake of eliminating bad selection from a set of services, the quality-of-service QS for a service S_j is calculated as follows:

$$QS(S_j) = \frac{ND_{ou} - RD_{ou}}{ND_{ou}} \quad (4.2)$$

ND_{ou} represents the number of outgoing dependencies, while RD_{ou} refers to the number of outgoing dependencies removed by other services as a result of negative behavior. The controller applies the QS metric to assess S_j quality based on the number of given accesses to other services. Consequently, the QS value increases steadily with the number of outgoing dependencies ND_{ou}, and vice versa. In this step, a list of services that have the required attributes is given to the controller, where the selection operation is standing on the one that has the highest QS value.

After selecting the best quality value, in other words, the best service that holds the desired attribute, the service requester remains in a suspended state until the control assesses its trust before accessing the data attribute. The S_i trust was calculated as follows:

$$TS(S_i) = \frac{ND_{in} - RD_{in}}{ND_{in}} \quad (4.3)$$

ND_{in} and $RD_{in}$are the number of ingoing dependencies and ingoing removed dependencies, respectively. The trust TS metric is applied to service requester S_i as a result of the number of accesses given by other services. The TS value grows steadily with the ongoing dependencies, and diminishes with the increase of revoked access by other services. The value of TS value empowers the controller to determine whether it accepts or rejects granting permission by referring to a predefined threshold. Both metrics are computed using thethree main matrices utilized by the controller. The following provides the definitions of matrices that assist in calculating the QS and TS values.

Definition1.*Dependency Matrix (DM):*

DM is defined as a squared matrix of size $n * n$ where n is the current number of services in the Dependency Graph. Each entry a_{ij} refers to the number of the outputs granted from S_i to S_j. Thus, the diagonal will have 0 values.

Equation 4.4 showcases an example of a dependency matrix in which three services interact with each other:

$$DM = \begin{pmatrix} 0 & 2 & 4 \\ 5 & 0 & 3 \\ 1 & 2 & 0 \end{pmatrix} \quad (4.4)$$

Building upon the matrix content above, it is deduced that service S_1 gives S_2two output access. The number of outgoing and ingoing were calculated using the rows and columns, respectively.

Definition2*.Revoked Output Access Matrix* (ROM)

ROM is a squared matrix with size $n*n$ where n stands for the number of the current services in the SDG. Every entry a_{ij} is the number of outputs revoked by S_i for S_j. Thus, the diagonal will have 0 as a value.

Definition3.*Removed Input Access Matrix* (RIM)

is a squared matrix with size n which is the number of the current services in the dependency graph. Every entry a_{ij} is the number of the inputs from the service S_j removed by the service S_i. Hence, the diagonal will have 0 values.

The controller fosters these matrices to calculate service $S_i$trust and quality. To calculate the quality metric, both DM and RIM matrices figured into the calculation of outgoings and removed dependency numbers, respectively. QS is given as follows:

$$QS(S_i) = \frac{\sum_{k=0}^{n} DM(i,k) - \sum_{k=0}^{n} RIM(k,i)}{\sum_{k=0}^{n} DM(i,k)} \qquad (4.5)$$

To calculate QS metric and from the matrix DM,the value of ND_{ou} can be replaced by the sum of all output values of the row i.Whilst the sum of all the values of column i of the matrix RIM is replaced by the value of RD_{ou}.Otherwise, to evaluate the trust of a given service requester. DM and ROM matrices are taken into consideration to calculate the number of ingoing and revoked dependencies, respectively; thus, TS metric is given as follows:

$$TS(S_i) = \frac{\sum_{k=0}^{n} DM(k,i) - \sum_{k=0}^{n} ROM(k,i)}{\sum_{k=0}^{n} DM(k,i)} \quad (4.6)$$

In contrast, to use the rows to determine QS value, TS value is determined using DM columns for calculating the ongoing number ND_{in}.Though RD_{in} is replaced by the sum of all values of the column i from the ROM matrix.

4.4 Implementation of the Proposed Framework SDGchain

This section provides a detailed description of how service interaction building on the SDG graph is added to the SDGchain framework. For the implementation of the proposed architecture, we used Eclipse[1] environment with Java and Jersey[2] REST API that allow to create web service and it developed using permissioned Hyperledger Fabric[3] Blockchain. The hyperledger fabric network includes two organizations, one peer node for each organization, and a CouchDB[4] as a World State Database.

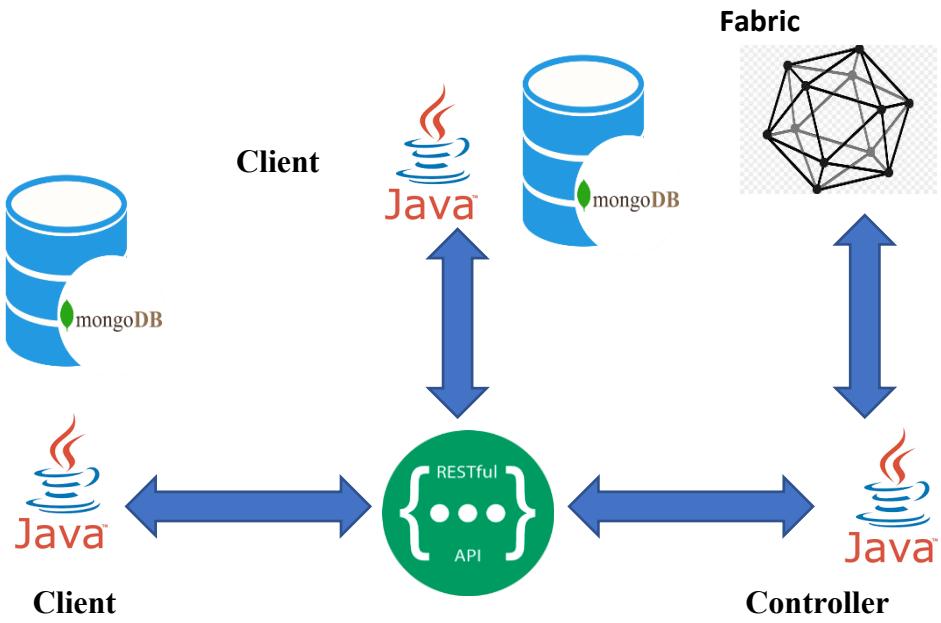

Figure 4.5: Implementation design of SDGChain and its main elements.

Besides three channels are created authorization, graph dependency, and logs. Three smart contracts (fabric chaincodes) are deployed using the Go language (one for each channel), and the off-chain system storage is MongoDB[5] which maintains service-encrypted data as JSON resources. An illustration of theSDGchain implementation is shown in Figure 4.5 in conjunction with the

[1] https://www.eclipse.org/
[2] https://eclipse-ee4j.github.io/jersey/
[3] https://hyperledger-fabric.readthedocs.io
[4] https://couchdb.apache.org/
[5] https://www.mongodb.com/

interaction between the components. The controller is deployed as a Java REST web service that uses Tomcat V9.0 as the server. Each client interacts with the controller using HTTP requests (port 8080) along with thepath names and query parameters. The data owner saves personal data in the off-chain by querying the MongoDB server. The controller issues a query to the Hyperledger fabric network whenever it wants to update the world states of blockchain.

Table 4.1 : The main functionalities that are given by SDChain

Functionality	**Path**	**Query parameters**	**Response**
Inscription	/inscription	“description” “outputs”	Success or fail
Request attributes	/request_data	“pubkey” “sign” “attributes” “operation”	dataId or null
Check policy	/check_policy	“pubkey” “sign” “dataId” “pubkeyTarget” “operation”	True or false
Update outputs	/update_outputs	“pubkey” “sign” “outputs” “dataId”	Success or fail
Remove permission	/remove_permission	“pubkey” “sign” “dataId” “pubkeyTarget” “operation”	Success or fail
Request data	/request_data	“pubkey” *“sign” “dataId” “operation”	Decrypted data

Table 4.1 presents the main functionalities of SDGChain along with the path name and query parameters. A JSON format key value is adopted to save the descriptions and outputs of the clients (graph dependency and authorization ledger, respectively) within the Hyperledger Fabric network. Therefore, every interaction between clients, client-controller or controller-blockchain is in JSON format. The service owner launches the check policy whenever it wants to check that a data service requester already has the required permission. Listings 4.1 and 4.2 illustrate a sample of the content of the JSON description and theoutputs of a service. As an example, an entry in the Autledger is shown in Listing 4.1, whilst the SDGledger's entry is shown in Listing 4.2. The listings demonstrate that the target service already exists in the dependency of the data ID. In SDGchain, the owner of the output data may differ from the ID of the service description. For instance, one service may save the output data of another service. In such a situation, the controller returns to the original owner of the attribute permission.

```
{"DataId":"Ssn+W7ipKw……",
  "policy":[
     { "target":"MIGfMA0GCS……",
       "op":["read","write"]
            }],
      "DefaultPolicy" :["read"]}
```

Listing 4.1: Example of an output policy entry represented by JSON key-value

```
{"service": "MIGfMA0……",
   "outputs": [{
      "id": " Ssn+W7ipKw……",
      "owner": [
        "MIGfMA0……"
      ],
      "atributes": [
        "name",
        "age",
        "disease"
      ],
      "dependencies": ["MIGfMA0GCS……"]
            }]}
```

Listing 4.2: Example of a service description entry represented by JSON key-value

In the implementation, we opted to select CouchDB as world-state storage for the Fabric nodes rather than adopting LevelDB for these reasons: compared to CouchDB, LevelDB has a poor query expression, which has more expressive power of embodying queries to the ledgers. First, we attempted to use LevelDB, but encountered some issues that prevented the selection of all the service descriptions that matched the desired output. As a result, all service descriptions are requested, and a matched output is searched. However, CouchDB enables the use of query selectors with the key expression “$elemMatch” to immediately recuperate all thematched outputs.

4.4.1 Framework Performance Evaluation and Results

4.4.1.1 Experimental Configuration

Numerous tests have been conducted to demonstrate the functional capability and evaluate theSDGchain. However, in the absence of realistic datasets, we generated a sample of a set of service descriptions and their relevant outputs as a dataset. All the experiments are executed on a local server in a machine with an Intel Core i7 processor running with a 1.8 GHz clock speed, 16 GB memory, 128 GB SSD and 1 TB storage HDD. In the blockchain network, all elements such as organizations, certificate authorities, peers and CouchD, are integrated as Docker images. Experiments are carried out on Hyperledger Fabric blockchain with varying numbers of SDGledg entries (nodes in theSDG). Tests were conducted using the JAVA REST API to allow interactions between clients and controllers using ten simultaneous requests.

4.4.1.2 Results Analysis

The experimental results are shown in Fig 4.6, Fig 4.7, and Fig 4.8. Fig 4.6 shows the average response time values for theinscription check policies and request data, the execution time is strongly correlated with the time required to execute a blockchain transaction. We evaluated SDGchain by considering transaction types, such as reading or writing data into ledgers. For example, the inscription process must be written. However, check policies settle for a reading transaction from the ledger only, whereas data requests require two types of transactions, read and write, which results in an increase inthe average response compared to others. Hence, Fig 4.6 is seen as a validation of these three types of transactions.

Fig 4.7 demonstrates the evaluation of the ledger’s size in the world state. A different number of SDG nodes are taken as SDGledger entries to evaluate the sizes, and it is obvious that for 800 nodes, we obtain 1 Mb for SDGledger, and we deduce that for 800000 nodes, we will consume almost 1 GB, which may overcome with CouchDB. Therefore, adopting off-chain storage would affect the world state size because the data ID and hash are kept in Autoledg. Fig 4.8 presents a detailed

evaluation of the chaincode invocation for theSDGledger. Several Go functions in chain codes are deployed, among the paramount ones are CreateAsset, ReadAsset, UpdateAsset and GetAllAssets. It is remarkable that the execution time is not affected by the number of entries in either ledger.

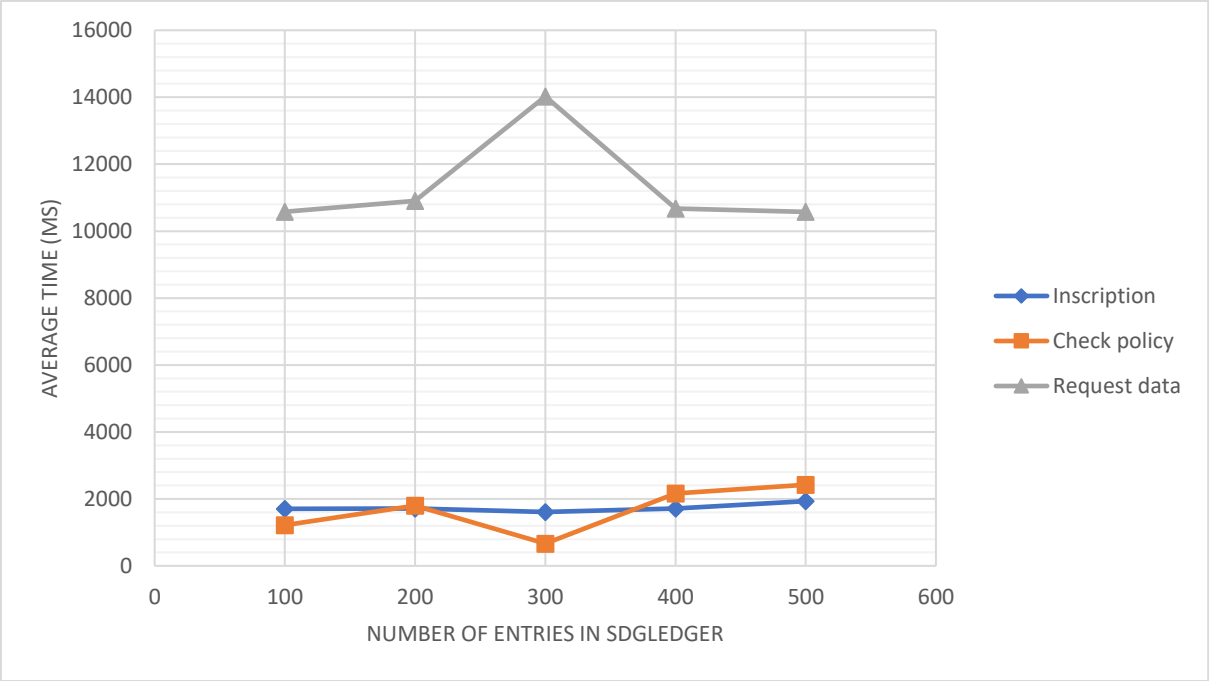

Figure 4.6: Evaluation Results of the execution of SDGChain Main functionalities

Figure 4.7 : Evaluation results related to CouchDB size evolution

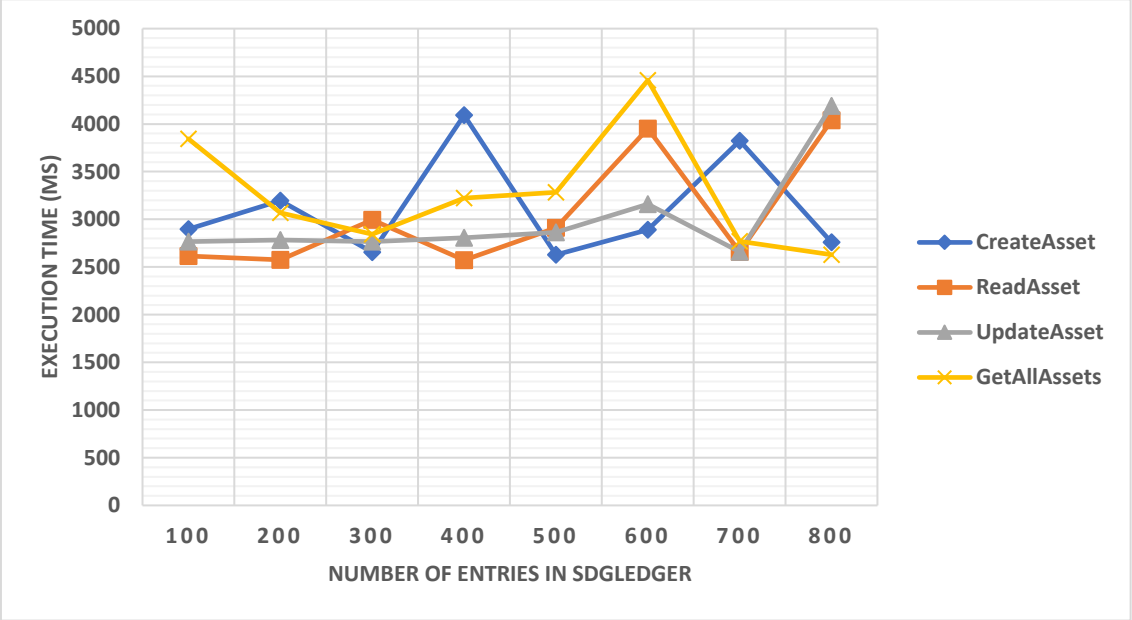

Figure 4.8: Evaluation results of the execution of SDGChain Smart Contract

4.4.2 Comparative analysis of the proposed framework with state-of-the-art

It is necessary to ensure the security objectives in every proposed model, such as authentication, confidentiality and integrity, where the security requirements and satisfactory proposed solutions are presented briefly in Table 4.2. The adoption of blockchain in our system ensures several features, the most important of which is a tamper-proof distributed ledger.

4.4.2.1 Data Security

Because the controller is a manager entity, it is qualified to run several tasks, which render the worst threat scenarios when taking it over, which leads to the modification of everything in the system. To address this issue, we assume that the controller is honest but curious. In contrast, SDGchain is an owner-centric model that allows the determination of the degree of data sensitivity by setting up default permission; the more sensitive the data, the more secure it is. Whenever data are critical, the service requester must obtain the owner's permission.

Table 4.2 : The Security Requirement

Security objectives	Proposed solution
Confidentiality	Asymmetric encryption
Authentication	Pseudoidentity (P_k)
Authorisation	Pseudoidentity +policy
Integrity	Hash

We assume that an intruder succeeded in generating a key pair that may use a fake identity to access the system. To address this issue, the controller maintains a legitimate public key dataset. We say that a party is legitimate when a corresponding public key exists in the list. When a legitimate entity loses its pseudonyms by violating its public key, the adversary cannot interact with the system owing to asymmetric cryptography, which empowers the use of digital signatures using private keys. Another possibility is that the adversary can be a legitimate entity, such as a malicious service that gains data access for the first time, andwe avoid frequent access by decrypting the data only by the owner, which would address the issue of sharing the secret key with third parties. The SDG contributes to alleviating the attack impact by protecting the system from suspicious services that behave abnormally, where the controller interferes and cuts all relevant dependencies, which is considered a main shortcoming in all related works. Moreover, only the controller is qualified to access ledgers (updating, creating, and deletion). etc.). Hence, when a service is taken over, it does not cause any effect, whether creating or even modifying both dependencies or permission in Autledg or SDGledg. In this case, the compromised service can only request access to other services; however, in our design, it stands on evaluating trust.

4.4.2.2 Data Privacy

The controller is empowered to maintain service privacy through a centric mode (the owner has full sovereignty concerning its data) by utilizing an SDG graph. Because cryptographic methods are employed to secure data service, and without the need for a third party, only the data owner and requester who previously had access permission can learn about the raw data. The process of selecting the best attribute has two goals: the most important is the selection, according to the evaluation of trust and quality. Simultaneously, the second is to maintain the privacy of the service owner. We mentioned earlier that the data were protected using asymmetric cryptography; now, considering identity privacy, it is protected using pseudonyms (public keys) that avoid linking and inferring the real identity. The owner manages his data by granting or rejecting permission to the requester, and this is a strength point in preserving privacy, where the controller is empowered to make decisions on whether to accept or reject permission for suspicious services, though with owners' demands, that would assist the owner because down the line, it would not have global knowledge about the SDG and the interactions between services.

4.4.2.3 Comparison

A comparison analysis of the proposed framework with state of the art methods in terms of blockchain implementation, entity privacy, access control method and protection of the dependency

graph. The SDGchain framework is compared with existing blockchain privacy preserving frameworks such as [81] and [41].

Table 4.3 : Comparative Analysis between SDGChain and similar works

Method	Permissioned Blockchain	Privacy	Access control	Dependency Graph	Graph Protection
[81]	No	User	Policy	Permissions	Anonymization
[41]	Yes	User	Policy	Permissions	Anonymization
SDGchain	Yes	User, Service	Evaluation, Policy	Services, Permissions	Anonymization, Graph modification

From Table 4.3, previous studies have focused on sharing permission between users and services. Thus, the data owner can grant or revoke permission directly to the requester by asking the system. According to [81] and [41], the solutions ignore the privacy of the service among other services or even users. Using an access control policy and granting access directly without evaluating the trust level, we treated the issue of service privacy as follows:

- The service's privacy is maintained through determining the level of trust.
- Because early solutions do not use an explicit dependence graph, they only settle for using anonymization to protect the services, which results in attackers being able to infer dependencies from a set of policies. In contrast, our design protects the SDG graph by employing an encryption mechanism for both attributes and graph modification.
- As a strong point, the controller counts on SDG graph for alleviating the negative impact of an intrusion by cutting off the dependencies.

4.5 Chapter Summary

In this chapter, a privacy-preserving platform, SDGchain, is proposed in a service-oriented architecture to achieve the first objective. More specifically, we discussed our proposed SDGchain, a new personal data protection model-based permissioned blockchain using hyperledger fabric. To allow control over confidential data exchanged inter-services and presents a global view of system interactions. Our SDGchain can achieve a good feature in terms of execution time compared with related works. Finally, intensive experiments were performed to validate the effectiveness of SDGchain, despite the absence of a real evaluation dataset. The results validated that using off-chain storage has a positive effect on the world state size because only light data (ID, hash) are maintained.

PrSchain: A Blockchain Based Privacy Preserving Approach for Data Service Composition

5.1 Chapter overview

This chapter discusses the privacy-preserving problem in service composition by presenting the proposed framework called PrSChain. It considers the problem of maintaining the privacy of service providers during aquery plan execution based on apermissioned blockchain. Therefore, an introduction to the new problem is presented in Section 5.2. Then, section 5.3 puts forward the proposed PrSChain in detail. The performance evaluation and discussion are addressed using an execution simulation in Section 5.4. Finally, Section 5.5 sums up this chapter.

5.2 Introduction

To achieve auser query response, multiple data services are unified to produce a new composed service that requires data from various service providers, which they allow to incorporate to fulfill the desired query. In SOA, service composition usually requires providing services to achieve user queries, which claims to share data from multiple data service providers [62]. In earlier work, to preserve data service privacy, the composition process-based mediator that prevents direct interaction between service providers;, it carries out a sentence of functionalities such as rewriting the query and creating and executing the composition plan by selecting the most adequate services. The mediator performs these functions without any guarantee of trustworthiness, and opting for an untrustworthy mediator may drive the disclosure of critical information. A composite service [74] can be defined as a service that unifies two or more services to fulfill a user query. Service composition consists of five fundamental steps, as depicted in Fig 5.1 [63]. It is launched when a user issues a specific query by requesting a service, and the mediator, in turn, analyzes the query and determines the list of services that can be composed. To realize this process, the mediator adopts

a matchmaking algorithm to select required services. Subsequently, a plan composition is generated to accomplish the query. Srivastava et al. [118] have seen the composition plan as a Direct Acyclic Graph (DAG), in which nodes refer to service participants, while edges correspond to the dependencies between participants, each service could be a parent, child, or both at the same time. The execution can be run inparallel when there is no dependency between the services. Finally, the results were sent to the user requester.

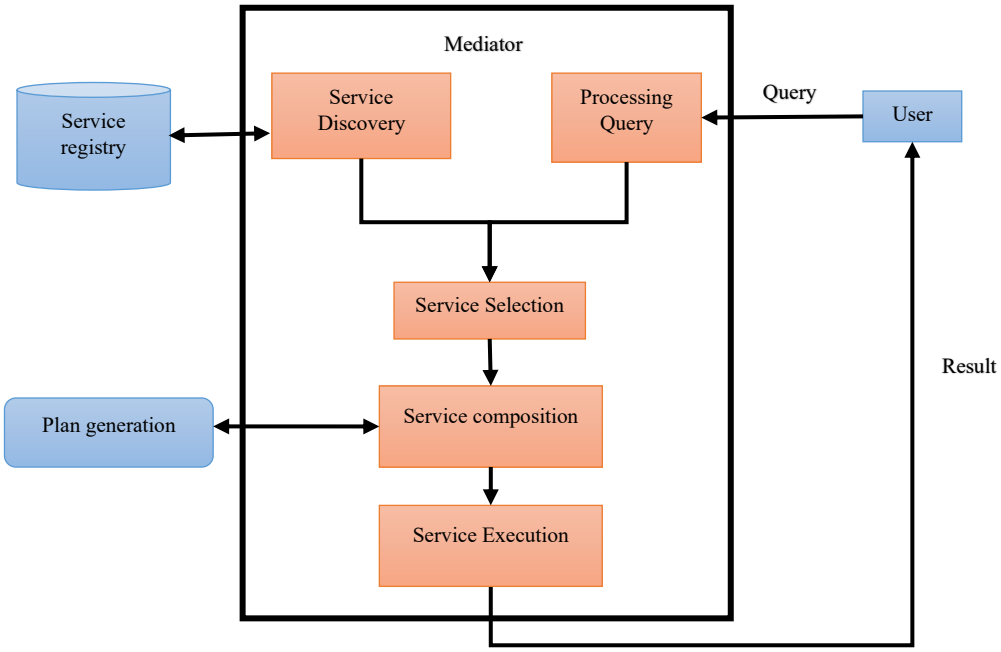

Figure 5.1: Conventional Service Composition Process [63].

Many approaches intended to secure data service composition via traditional methods, such as differential privacy, secret sharing, and even k-anonymity, have been proposed to tackle data breaches and preserve privacy during service composition. Most of these studies abandoned mediators to provide privacy. Recent works have fostered the k-anonymity method, which involves trading between using a big k to ensure privacy and the time of data release. Consequently, a system scalability issue appears. Recent solutions adopt k-protection, which stands for an extension of K-anonymity that remains the same issue, and aim to share the service composition plan with all service providers that have taken part. Blockchain technology is invested to ensure trust security while maintaining the data privacy of the Service Providers (SPs) during the service composition process and to leverage the benefits of blockchain. This work presents a solution to the issue of untrustworthy mediators. We propose a novel solution that demonstrates the integration of permissioned blockchain in data service composition partakes by constructing a privacy-preserving system based

on Hyperledger Fabric for data service composition, which is known as PrSChain. Our approach leverages the decentralization feature of blockchain to generate, create, and even execute a composition plan by using smart contracts without interfering with the central mediator. The contributions of this chapter are as follows:

- We propose a decentralized privacy-preserving system for the service composition process based on the blockchain in the SOA context, known as PrSChain. Through PrSChain, we enable thetraceability, verification, and integrity of the composition process without the need for a third party to prevent a single point of failure.
- The use of Hyperledger Fabric enables the execution of service compositions in a private environment, which encompasses the processes of authentication, authorization, access control, audit, identity management, plan generation, and execution. All of these processes are employed by integrating chaincodes (smart contracts).
- To overcome the inefficiency of using a mediator, we implemented a coordination entity with limited responsibilities. The coordinator is unaware of the service providers and real composition plan.
- To overcome thescalability issue, we aim to use an IPFS as off-chain storage for scalability purposes; as such, IPFS data access is restricted only to legal participants from theblockchain system.

5.3 Proposed Work: PrSChain

In this section, the proposed method is described in detail. Execution of a service composition process and an access scenario based on Hyperledger Fabric. This section also provides an overview of the system model and the relevant algorithms of service composition query using the proposed privacy preservation PrSChain.

5.3.1 PrSChain: blockchain-based privacy preserving in service composition

Service composition privacy preservation is needed to maintain the privacy of all service providers that have already taken part in realizing a user's query. The panoramic idea of the proposed design is centered on the fact that the proposed platform binds to aservice composition that counts on theblockchain, which is a novel privacy-preserving approach. Thus, the proposed system model PrSChain is presented in Fig 5.2, which is treated as a secure and privacy-preserving platform that intends to carry out a user query that requires data service composition to obtain the answer. An intermediate coordinator with low responsibility is used. Consequently, it is a distrust entity that

processes the query and is in charge of sharing the process result with the blockchain. A permissioned blockchain embodies a trusting system. It embraces all functionalities on behalf of an untrustworthy mediator. Among the main activities are generating the composition plan and replying to only a limited plan bythe coordinator without providing the necessary information (e.g., output and input attributes between service providers), whereas the coordinator launches a notification to the set of service providers to the aim of starting execution, as per the work of PrSChain the coordinator does not process any information that belongs to the services provider sub-query. Each selected service provider demands its proper subquery as well as information about theinput IPFS data and its children's public keys from theHLF Blockchain. Subsequently, it requests IPFS to obtain all the previous results that belong to all its parents for executing its proper subquery and also saving the outcomes into IPFS to recuperate the hash. The latter is encrypted with all thechildren's public keys and saved in the HLF system. After the execution of all sub-queries by all service providers, the coordinator requests the IPFS by using the recuperated hash from the HLF system to obtain all the final results. Finally, it joins all thefinal outcomes and returns the final response to the user.

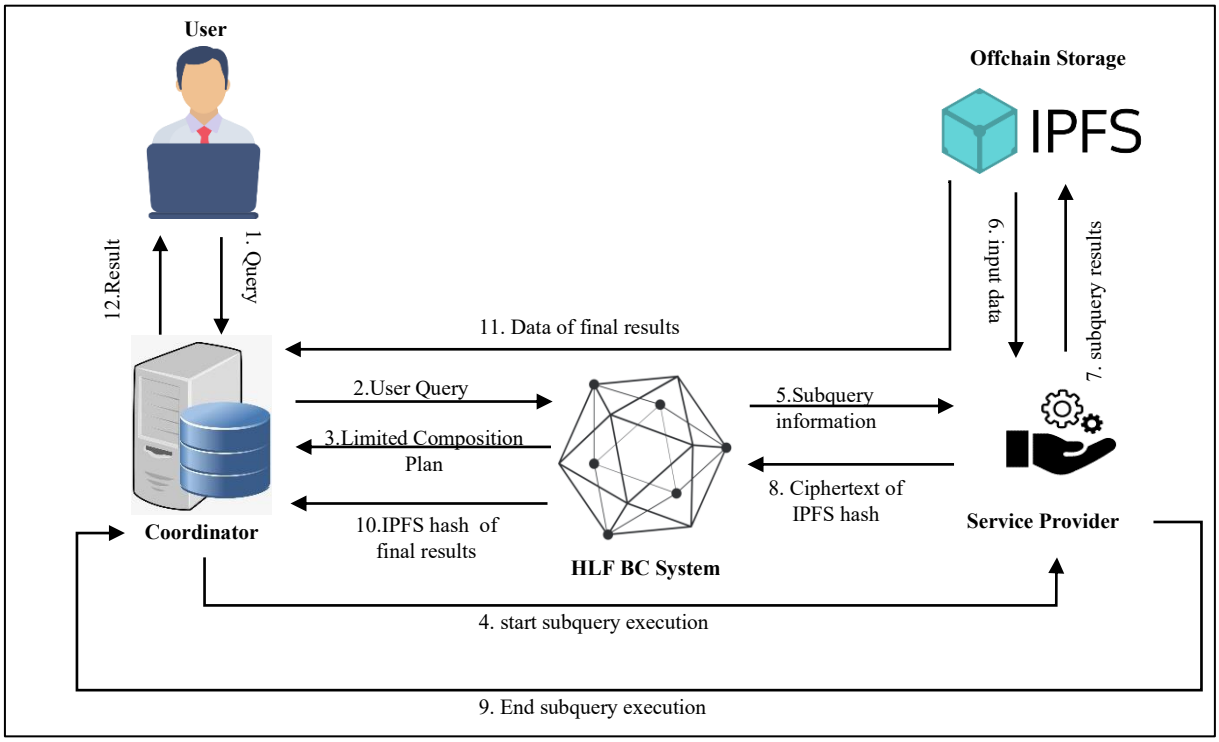

Figure 5.2: High-level system architecture of the proposed BC-based preserving privacy of Data Service composition

Example 1 supplies a typical case of service composition based on a real scenario, which will go along all the next sections to illustrate the use of the blockchain to preserve the privacy of the involved partners during service composition. This example is inspired by the healthcare field,

where the database of diabetes medications [131] provides the required data for executing service composition. We assume that the end-user (ex. Doctor) issues the query, "What is the number of days in hospital and number of medicaments given to Afro-American male patients who have been supervised by doctors in Cardiology and they have changed their diabetes medication? To obtain a response to this question, four data service providers ($DS1, DS2, DS3, \& DS4$) are involved, where all the data service DS provides medical records about patients with several medical attributes that are given as follows:

- D_{s1}: Patient ID (PId), Gender (Gn) and Race (Rc).
- D_{s2}: Patient ID (PId), Number of days spent in the hospital (NDH) and whether there is a change in his diabetes medication or not (ChD).
- D_{s3}: Patient ID (PId), Encounter ID (EId) and Medical Specialty (MS).
- D_{s4}: Encounter ID (EId) and Number of medications taken (NM).

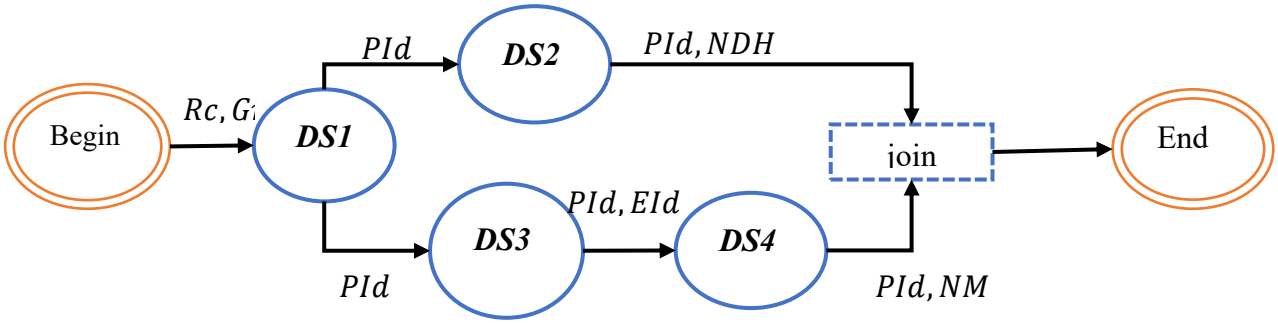

Figure 5.3 : An example of a data service composition related to the medical domain.

We attempted to generate a service composition plan involving four service providers. Fig 5.3 shows the outcome of the composition plan generation that belongs to the stated example, where each service provider is associated with a sub-query and is in charge of executing it. Otherwise, the coordinator supervises the execution of all sub-queries. Finally, it joins intermediate results to obtain the final query response. The aforementioned query can be split into four sub-queries, starting with $DS1$, which selects all male patients who were originally from Afro-America, while $DS2$ is liable for extracting all times in the hospital for input patient identifiers that already have changed their diabetes medication. Then, $DS3$ selects all encounter identifiers of the input patient identifiers, with the medical specialty being cardiology. Finally, $DS4$ concludes by selecting all medicamentstaken by the input encounter identifiers."

In order to clarify the work of PrSChain, Fig 5.4 illustrates the fundamental steps that help to provide a detailed overview of the data service composition privacy preserving protocol, where the next subsections describe each component and their main functionality interactions.

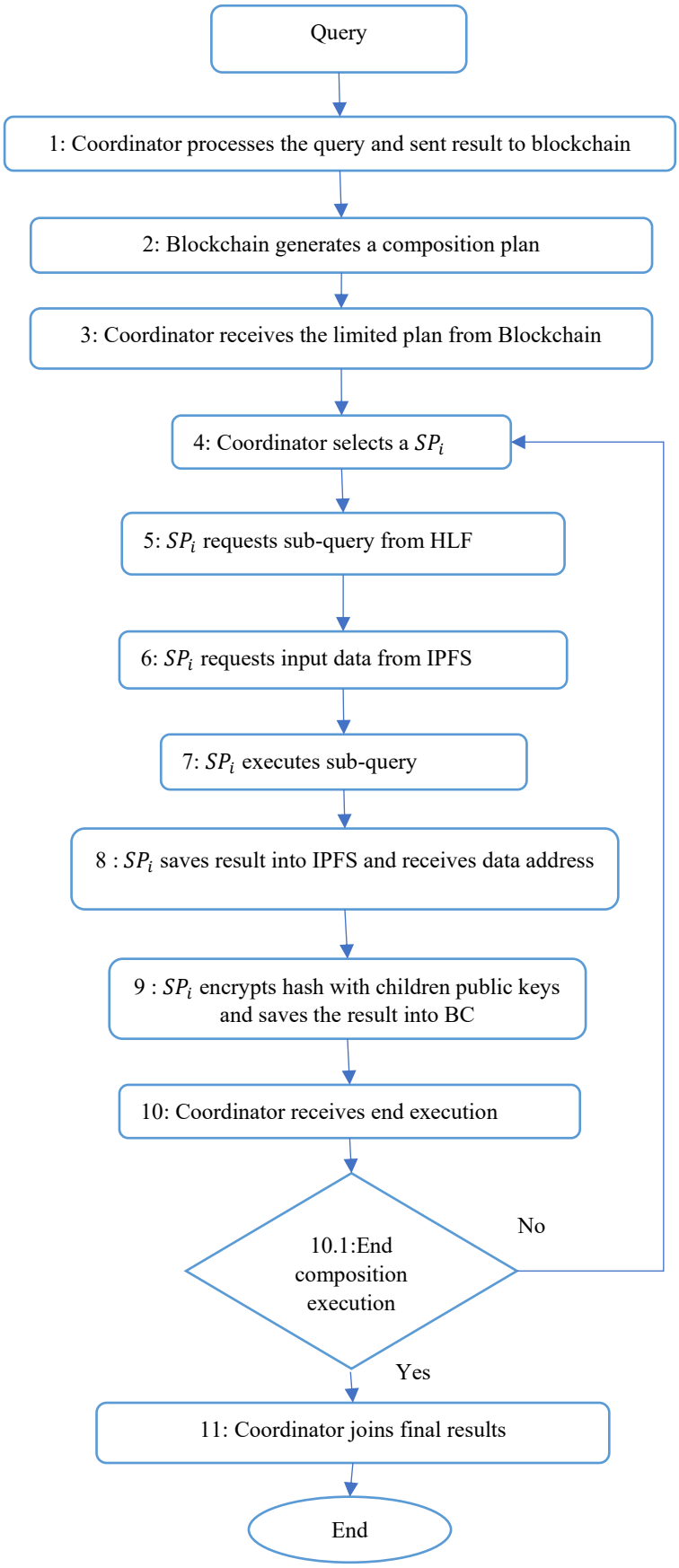

Figure 5.4: PrSChain Steps for Privacy Preserving in Service Composition

Algorithm 5.1 Coordinator's Steps of Execution

Input: $user\_query$

Output: $query\_results$

1	**var** *comp_plan* is the generated service composition plan without attribute names;
2	**var** $query$ is the result of query processing of user query;
3	**var** $completed\_SP$ is a set contains the service providers that complete the execution of the subqueries;
4	**var** $final\_results$ is a set that contains query execution results from the service providers that do not have any child;
5	Process $user\_query$ and store it in $query$;
6	Send $query$ to the blockchain by invoking specified smart contract;
7	Get the generated composition plan from the BC via smart contract and stores it in $comp\_plan$;
8	Initialize $completed\_SP$ with the service providers (in $comp\_plan$) that do not have any parent;
9	**While** (the size of $completed\_SP$ does not equal to the number of the SPs in $comp\_plan$) **do**
10	-Take a service provider (Say S) from $comp\_plan$;
11	**if** (all parents of S are in $completed\_SP$) **then**
12	Send a notification to S to execute its subquery;
13	**if** (execution completed is received from S) **then**
14	add S to $completed\_SP$;
15	**else**
16	abort;
17	Get final results from the blockchain via smart contracts invocation and store them in $final\_results$;
18	Process the final results and store the result in $query\_results$;
19	Return $query\_results$;

5.3.1.1 Processing and sharing the query by the coordinator

According to the PrSChain protocol, the coordinator plays an important role as a unique entity that undertakes component interactions in the form of a client/server architecture. The coordinator is in

charge of (steps 1,3,4,10, and 11) from Fig 5.4 starting with processing a user‘s query that conforms step 1, since it is untrustworthy, after which it shares the outcomes with blockchain peers to generate a composition plan that corresponds to lines 5 and 6 of Algorithm 5.1.

5.3.1.2 Blockchain generates service composition and maintains its integrity

After ending step 1, Blockchain receives query processing results, and thereby Blockchain recruits its peers to start executing smart contacts that correspond to generating the composition plan (step 2 in Fig 5.4) and [lines 5, 6 and 7 in Algo 5.2], they utilize all data about service providers for instance their registered services and their input and output attributes that they exist on-chain. Peers then attempt to select an optimal plan. As per our contribution, we did not concentrate on selecting the best services. Hence, we fostered the idea of [118]. Consequently, plan generation is executed in a decentralized environment without requiring a coordinator for many reasons, such as ensuring transparency. Fig 5.5 illustrates the result of the composition plan generation that corresponds to the running example. At the end of the plan generation, blockchain stores the outcome [line 8 in Algorithm 2] to maintain both data privacy and integrity. The process is continued, and blockchain issues an anonymized composition plan (step 3 in Fig 5.4) and [line 9 in Algorithm 5.2] to the coordinator to orchestrate the execution phase. Fig 5.5 is derived from Fig 5.3, which illustrates the service composition that belongs to Example 1, which includes only the required information that assists in the authentication process between theSPs and coordinator, and organizing the plan execution. Because the blockchain has replicas of its composition plan, the coordinator cannot tamper with it. In addition, it is a blind entity, and it is not aware of critical information about service composition.

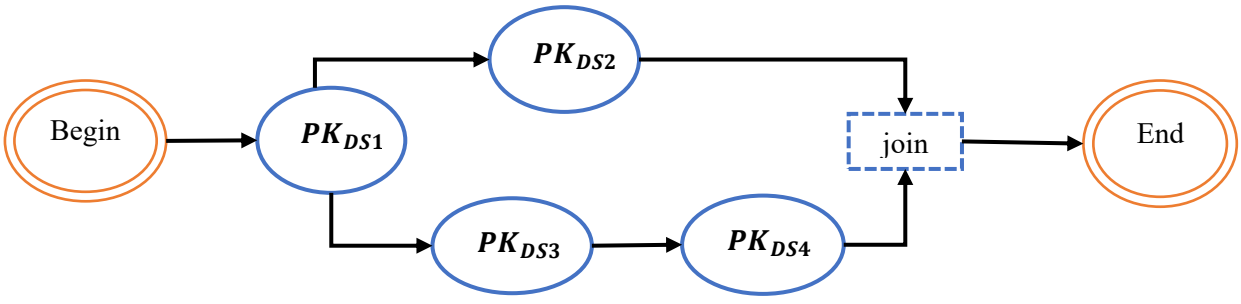

Figure 5.5: Anonymized Service Composition Plan extracted from the example in Figure 5.3.

Algorithm 5.2 Blockchain's Steps of Execution **Input:** $query$ **Output:** $comp\_plan$	
1	**var** $comp\_plan$ is the generated service composition plan;
2	**var** $service\_providers$ is the set of all registered service providers;
3	**var** $sub\_queries$ is the set of all generated subqueries;
4	**if** (Coordinator send a user query) **then**
5	Store the query in $query$;
6	Generate $comp\_plan$ using $query$ and all data provided by $service\_providers$;
7	Generate $sub\_queries$ using $comp\_plan$;
8	Store $comp\_plan$ and $sub\_queries$ in the blockchain using smart contracts;
9	Send $comp\_plan$ to Coordinator without the attribute names;
10	**if** (a service provider SP asks for its subquery SQ_{SP}) **then**
11	Query the blockchain for SQ_{SP} using smart contracts;
12	Send the subquery SQ_{SP} to SP;
13	**if** (a service provider SP asks for PK_Ch_{SP} the set of public keys of all its children) **then**
14	Query the world state for PK_Ch_{SP} using smart contracts;
15	Send PK_Ch_{SP} to SP;
16	**if** (a service provider SP asks for $addr\_Pr_{SP}$ the set of ciphertexts of the results of its parents) **then**
17	Query the world state for $addr\_Pr_{SP}$ using smart contracts;
18	Send $addr\_Pr_{SP}$ to SP;
19	**if** (a cyphertext of the address of the result of a service provider SP is received) **then**
20	Store the cyphertext in the blockchain using smart contracts;
21	return $comp\_plan$;

5.3.1.3 Service composition execution by the coordinator

In this phase, the coordinator receives the anonymized composition plan from blockchain (step 7 in Algo 5.1). First, it notifies all service providers (step 4 in Fig 5.4) that only play the role of parents to start sub-query execution (step 8 in Algo5.1). Once subqueries are terminated by service parents, the coordinator only notifies theservice providers that their parents have terminated their execution (lines 11 and 12 in Algo5.1). This process was carried out until all subqueries were executed. Finally, the coordinator joins all the results to obtain the final result and replies to the requester (steps 17, 18 and line 19 in Algo 5.1). On the contrary, the coordinator is bounded by limited functionalities compared to the traditional service composition ‘mediator, it orchestrates the sub-queries execution without learning about their content or even their result. At the same time, it is aware of the final results and has no correspondence with SPs. Thus, SP privacy is maintained by untrustworthy coordinators.

5.3.1.4 Service Provider Executes Sub-Query While Maintains Privacy

Every service provider aims to execute a subquery while maintaining data privacy. Our work takes advantage of blockchain as a trusted mediator along with the following important restrictions:

- Service providers do not learn about service composition plans.
- Blockchain ensures an authentication process among service providers, which results in SPs not knowing each other.
- Each sub-query is accessed only by its liable SP and even the required data for execution.

After applying these restrictions, the use of K-protection to maintain service composition privacy becomes unnecessary because each SP cannot learn who owns the data that are incorporated during sub-query execution. Detailed steps descriptions followed by each SP for accomplishing its partaking while executing the service composition are as follows.

a) Service provider requests subquery from Blockchain -step 5

The coordinator sends a notification to the service provider to start theexecution (line 8 of Algorithm 5.2). This latter request for its own sub-query and the relevant data that was recuperated from blockchain (lines 9 and 10 of Algorithm 5.2) to obtain the necessary information that allows it to be executed:

• Sub-query content, which includes input and output attributes (lines 10, 11 and 12 of Algorithm 5.2).

• Relevant children (SPs that are connected with input attributes) and their corresponding public keys (lines 13, 14 and 15 of Algorithm 5.2).

• The ciphertexts of IPFS addresses that correspond to data inputs relevant to sub-queries (lines 16, 17 and 18 of Algorithm 5.2).

b) Service provider acquires subquery input data from IPFS -step 6

In the event that the SP has more than one parent in the service composition, its corresponding sub-query requires data from all SP parents to terminate the execution. The SP uses its secret key to decrypt all the IPFS addresses (line 12 of Algorithm 5.3). Subsequently, the data IPFS is immediately recuperated (lines 13 and 14 of Algorithm 5.3).

c) Execute and store sub-query results on IPFS by the service provider -Steps 7 and 8

To execute the SP sub-query, the inputs require all decrypted parent data, after which the SP stores the results on theIPFS and recuperates the corresponding data address (lines 16 and 17 of Algorithm 5.3).

d) Service provider encrypts IPFS address and saves it on Blockchain -step 9

Asymmetric cryptography is used to protect data security, where the result address is encrypted for the sake of allow access only to SP's children. To achieve this, the SP utilizes each child's public key to encrypt the result address and saves it in the blockchain. The latter manages access to encrypted data by allowing only SP children to access the desired result, that is, data of their parent (lines 18, 19 and 20 of Algo 5.3, lines 19 and 20 of Algo 5.2).

Fig 5.6 shows an example of a query plan extracted from the service composition created by theblockchain. After generating the composition plan shown in Fig 5.3, the blockchain creates a query plan where each service provider is assigned to a specific subquery. As stated before, PrSChain ensures SP privacy by hiding the plan from both the coordinator and all SPs to keep any sensitive data about the SPs' sub-queries and their results secure and private. Fig 5.6 illustrates the manner in which blockchain stores the query plan where each subquery encompasses the following information:

- ✓ Subquery input and output attributes.
- ✓ The ID of the next SP that will execute the subquery.
- ✓ Subquery ID, which will be shared only with its SP.

- ✓ Children's public keys.
- ✓ Ciphertexts of IPFS addresses of the subquery's result.
- ✓ The hash of the subquery results.

Algorithm 5.3 Service provider's Steps of Execution

Input: $\boldsymbol{sub\_query}$

Output: $\boldsymbol{sub\_query\_results}$

1	We refer the current service provider as SP;
2	**var** SQ_{SP} is the subquery associated with SP;
3	**var** PK_{SP} is the public key associated with SP;
4	**var** PK_Ch_{SP} is the set of public keys of SP children;
5	**var** $Addr_{res}$ is the offchain address of the subquery execution result;
6	**var** $addr\_Pr_{SP}$ is the set of all ciphertexts of the offchain addresses of the query results related to SP parents;
7	**var** $data\_Res_{SP}$ is the set of all subquery results of SP parents;
8	**if** (Coordinator asks for executing the associated sub query) **then**
9	Get SQ_{SP} from the blockchain via smart contracts;
10	Get PK_Ch_{SP}, SK_Pr_{SP} and adr_Pr_{SP} from the blockchain via smart contracts;
11	**for** (address add: $addr\_Pr_{SP}$) **do**
12	Decrypt add by using PK_{SP};
13	Get the data from the offchain using the decryption of add;
14	Add the data to $data\_Res_{SP}$;
15	**End**
16	Execute SQ_{SP} using all data in $data\_Res_{SP}$ and store it in $sub\_query\_results$;
17	Store $sub\_query\_results$ in the offchain and get $Addr_{res}$;
18	**for** (public key pk: PK_Ch_{SP}) **do**
19	Encrypt $Addr_{res}$ with the pk;
20	Save the cyphertext in the blockchain via smart contract;
21	**End**
22	return $sub\_query\_results$;

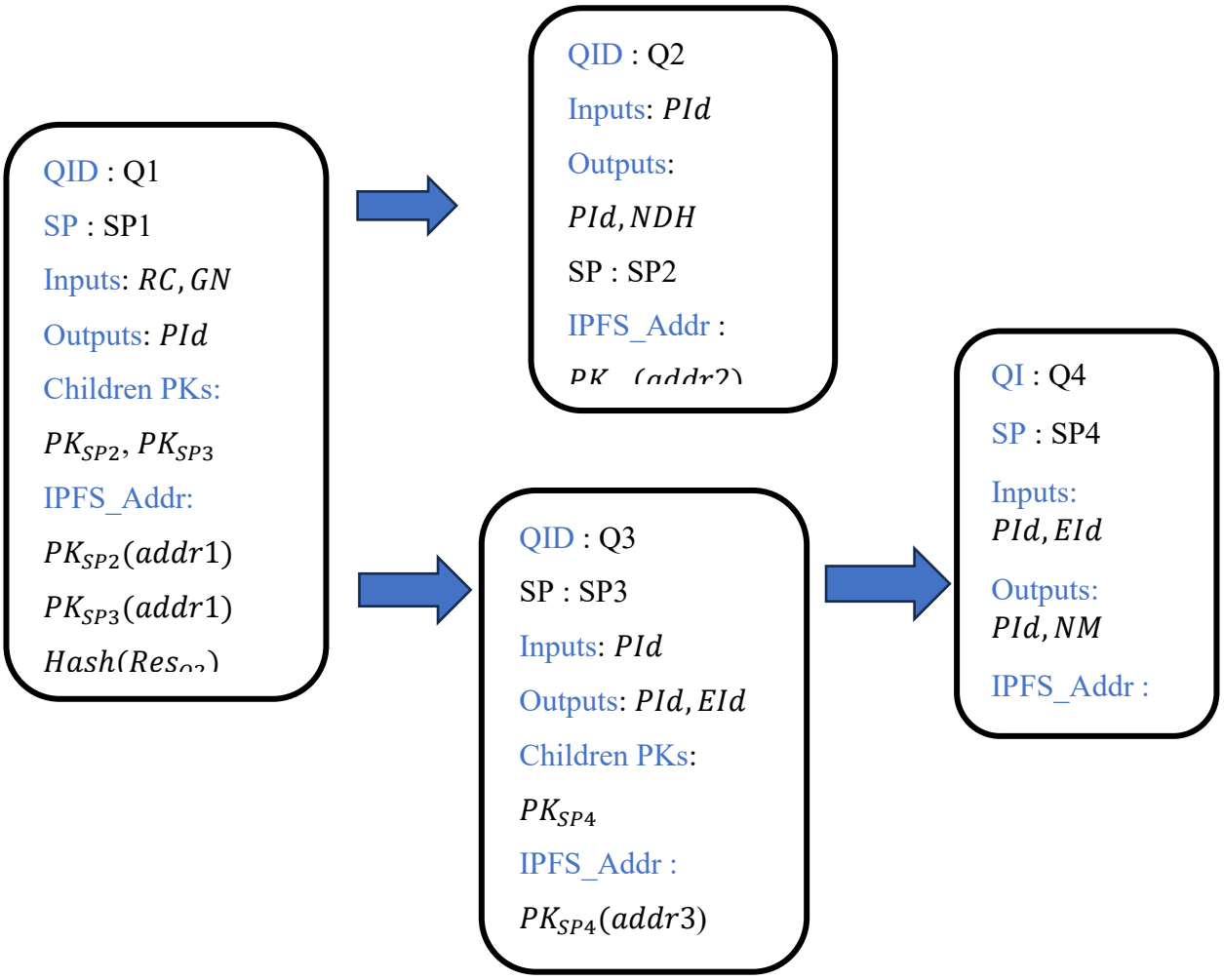

Figure 5.6: An example of a query plan constructed from the service composition in Figure 5.3

For instance, the sub-query $Q1$ consists of the following information: the service provider $SP1$,for inputs Rc (race) and Gn (gender),the output: Pid (patient identifier), $SP1$ has two children, public keys PK_{SP2} and PK_{SP3} , $PK_{SP2}(addr1)$ and $PK_{SP3}(addr1)$ are the ciphertexts of the IPFS address and its outcomes.

5.4 Implementation

The PrSchain approach is implemented using Eclipse and various Java APIs, such as JSON and Fabric SDK. Fig 5.7 illustrates the architecture of the framework, which demonstrates the fundamental components:

The crucial component is Hyperledger Fabric, which is used as a permissioned blockchain that allows only legitimate service providers to have valid certificates to interact with PrSChain. Owing to the properties of a private network, Hyperledger Fabric manages the client's identity by utilizing certificate authorities. The proposed blockchain network consists of two organizations, with one

peer node for each organization and one Couch database. The latter is used as a world state database and ordering service. The network was built with certificate authority for each organization. We created four channels for service providers,service Composition, query plan and logs, named respectively “Service Providers,” “Service Composition,” “Query Plan,” and “Logs”. We associated each channel with one chaincode (smart contract), which are deployed using the Go language. Fig 5.8 exposes the proposed network used by the proposed solution, where every channel is associated with its ledger and smart contract.

Moreover, the Interplanetary File System (IPFS) is defined as distributed file storage, where each file added to the IPFS has a unique address that is derived from a hash of the entire file’s content. In our proposal we opted to use IPFS as off-chain storage to safeguard the temporary data that are produced during sub-query execution. Our main aim is to alleviate the block size which guarantee blockchain scalability by maintaining only sensitive data, such as service composition and the corresponding subqueries.

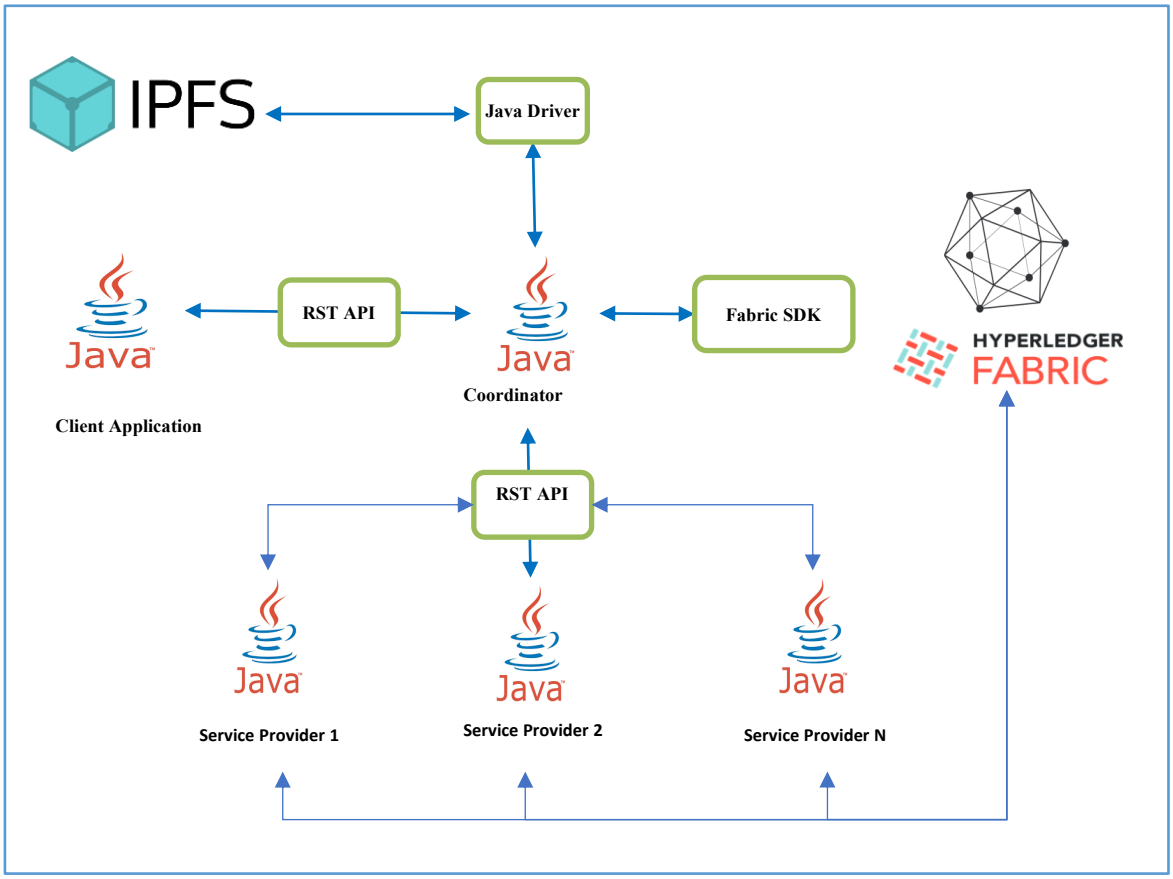

Figure 5.7 : Implementation Desing of PrSChain and its main components

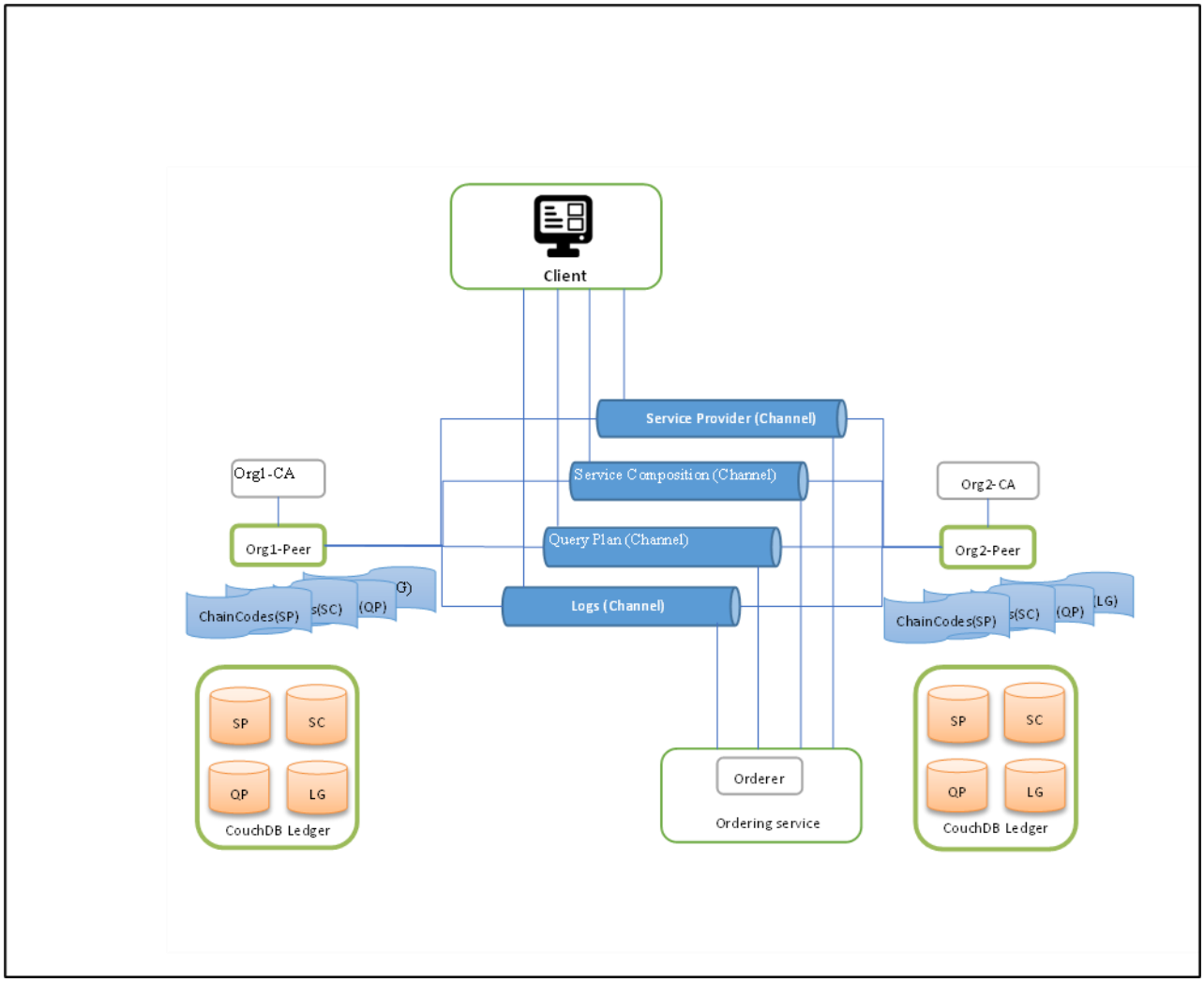

Figure 5.8 : The Hyperledger Fabric Network that is proposed by PrSChain.

5.4.1 Fabric Chaincodes and Distributed Ledgers

In Hyperledger Fabric, every peer has a copy of the ledger (local database), which brings together all valid transactions executed by the network via chaincodes. This results in each peer having many installed chaincodes in one channel. HLF ledgers are updated using smart contracts regarding the claims of external blockchain users. Our study allows the use of four distributed ledgers, each associated with one smart contract and multiple peers. These ledgers store sensitive data concerning implementation components, such as service providers, service composition, sub-query information, and operation logs. The following subsections discuss in detail the deployed chain codes that belong to each ledger from the four ledgers stated previously.

5.4.1.1 Service Provider Chaincode

The chaincode that belongs to the service provider defines some functions executed by Hyperledger Fabric peers to manage the participating service providers that are registered by the blockchain. The

channel holds the same name, “Service Provider,” as the installed chaincode, and it is associated with a local ledger that stores information about service providers. The SP chaincode supports the Golang structure, as illustrated in Listing 5.1, where some GO functions are provided in Table 5.1 in conjunction with restricted access.

```
type  ServiceProvider struct {
        SpID string `json:"SpId"`
        SpName string `json:"SpName"`
        SpAddress string `json:"SpAddress"`
        SpServices [] ServiceList `json:"SpServices"` }
type  ServiceList struct {
        ServiceID string `json:"ServiceID"`
        ServicePK string `json:"ServicePK"`
        Available string `json:"Available"`
        InputAttributes [] AttributesType `json:"InputAttributes"`
       OutputAttributes[] AttributesType `json:"OutputAttributes"` }
type  Attributes Type struct {
       AttrName string `json:"AttrName"`
       AttrType string `json:"AttrType"` }
```

Listing 5.1. The Golang Structure implemented by the Service Provider Chaincode

Table 5.1: Service provider Smart Contract Functions

Function	Description	Restricted Access
CreateServicePr	Create new service provider using the description given by the invocation parameters.	Administrator
GetMatchServices	Get a list of services that match the query attributes given by the invocation.	GenQueryPlan chaincode
updateSP	Updating existed service provider with new information.	Service Provider

5.4.1.2 Query Plan Chaincode

The QP chaincode determines the functions executed by Hyperledger Fabric peers to manage the generated query plan from theservice composition. The chaincode is installed into channel that known by the same name “Query Plan.” The QP chaincode utilizes the Golang structure which is demonstrated by listing 5.2, whereas Table 5.2 presents some GO functions along with restricted access.

```
type  QueryPlan struct {
        QueryID string `json:"QueryID"`
        SpID string `json:"SpId"`
        ServiceID string `json:"ServiceID"`
        ServicePK string `json:"ServicePK"`
        SpAddress string `json:"SpAddress"`
        ChildrenPKs [] string `json:"ChildrenPKs"`
        Address_Encryption [] AddressEncryptionType
                        `json:"Address_Encryption"`
        InputValues [] InputValuesType `json:"InputValues"`
        HashResult string `json:"HashResult"` }
type  AddressEncryptionType struct {
        Child_PK string `json:"Child_PK"`
        Address_Enc string `json:"Address_Enc"` }
type  InputValuesType struct {
        Attribute string `json:"Attribute"`
        Values [] string `json:"Values"` }
```

Listing 5.2. The Golang Structure implemented by the Query Plan Chaincode

Table 5.2: Query Plan Smart Contracts Functions

Function	Description	Restricted Access
GenQueryPlan	Generate the query plan that contains all the sub-queries associated to SPs.	Coordinator
SetAddrEnc_Hash	Save the cipher texts (using children PKs) of the address of the sub query results on the BC along with the hash of the result.	Service provider
GetQueryPlan	Get the sub query associated with SP which invokes this function.	Service provider
GetAddressEnc	Get all cipher texts created by all parents of the SP which invokes this function.	Service provider

5.4.1.3 Service Composition Chaincode

A Service composition chaincode is deployed to manage the anonymized composition plan; similar to the previous chaincodes, it is installed on a channel that is identified by the same name "Service Composition," whereas it is bound with a local ledger that stores information about the private service composition. The service composition chaincode is presented by the Golang structure in Listing 5.3, and Table 5.3 represents the synthesis of some GO functions with restricted access.

```
type  ServiceComposition struct {

        SpID string `json:"SpId"`

        SpAddress string `json:"SpAddress"`

        ServiceID string `json:"ServiceID"`

        ChildrenPKs [] string `json:"ChildrenPKs"`

         ServicePK string `json:"ServicePK"`

}
```

Listing 5.3. The Golang Structure used by the Service Composition Chaincode

Table 5.3. Some Smart Contracts Functions that are used by the Service Composition Chaincode

Function	Description	Restricted Access
GetPrivateSC	Get the private service composition which is generated from the query plan.	Coordinator

5.4.2 Performance Analysis: PrSChain

In this section, extensive experiments are conducted to evaluate the efficiency of thePrSChain. To demonstrate the robustness of our solution better, a security analysis range is proposed, which is accompanied by a comparison with the state of the art.

5.4.2.1 Experiments Set Up

To present the performance and robustness of our solution, a number of experiments were carried out in the machine with this hardware specification: Intel Core i7 processor running with a 1.8 GHz clock speed, 16 GB of installed physical memory, 128 GB SSD and 1 TB for storage. Regarding the implementation architecture, the coordinator is implemented as a JAVA REST application that uses Tomcat 9 as the resource server. All service providers and clients are presented as JAVA applications that interact with the coordinator byusing theREST API. The service providers and coordinator communicate with IPFS to recuperate sub-query outcomes and interact with the fabric network using Fabric SDK. ThePrSChain implementation deploys several Java APIs in various processes such as theIPFS APIand Fabric SDK.

5.4.2.2 Dataset

The dataset [131] is used to evaluate PrSChain, which consists of 101767 records, where each patient record has multiple attributes, a part of which is depicted by the running example in the section. The idea is to sample the entire dataset into four derived subsets, each using various attributes with the condition that two or more subsets partake in some attributes. Each service provider is assigned a specific subset to create the service composition shown in Fig 5.3, where the ultimate subset size is 101767 records. The first experiment started with 20k records, and then we add 20k to each experiment until all the datasets were terminated.

5.4.2.3 Experiment Results

To demonstrate the performance of PrSChain, an evaluation was performed over the execution of the issued query given by Example 1 in each experiment. Both the execution time and memory consumption in each experiment were recorded, whereas Figures 5.9, 5.10, 5.11, and 5.12 illustrate

the execution times (in milliseconds) that belong to SP1, SP2, SP3, and SP4 during subquery execution, and interaction with the blockchain network. The sub-query execution times consider the sub-query content, IPFS, and Blockchain network interactions, where the overall value is their sum. The subquery execution significantly affects the overall sub-query execution time. Owing to sub-query content, such as the number of variables, constants, and filters, we notice $SP1$ and $SP4$ have lasted less time than $SP2$ and $SP3$ after over the execution of 12k records if we return to the service composition, we find that both $SP1$ and $SP4$ only have two specific input values, which act as filters that lead to narrowing the sub-query search space. In addition, the number of SP parents is affected during execution because the child collects all input data from its parents for processing and reaching the sub-query execution. The relevant evaluation related to blockchain communication demonstrates that the values are slightly close, owing to the inward execution of the blockchain network. There exist two types of interactions between SPs and Hyperledger Fabric networks: initially request the sub-queries and then store the ciphertexts of IPFS addresses along with hash results. The SPs always execute the first one, which is related to the number of SP children. Figure 5.13 illustrates the entire execution time of the query by all SPs, where there were slight changes between theexperiments. We conclude that the data size affects the overall execution. The latter can be affected by the number of SPs that took part in the service composition and precisely by their structure. Hence, optimal selection techniques play an important role. We note that the sum of all sub-query execution times is higher than the overall value. Therefore, the parallel execution of $SP2$ and $SP3$ occurs because they are independent.

Fig 5.13 also shows the recorded times of the interactions between the coordinator and the blockchain network. The values are exceptionally close, as they are related only to Hyperledger Fabric's inward communications. The overall coordinator execution time is not presented here, and the only factor is the coordinator's final merging of all results. Large-sized results can augment overall time.

We recorded the memory consumption in Fig 5.14 of both the coordinator and all SPs, in order to demonstrate a precise evaluation. Otherwise, in [47], the evaluation focused on the overall consumption of the execution. In reality, the components are distributed in the network; therefore, it is practical to evaluate each application alone. We observed that the coordinator's memory space consumption did not vary, remaining at 600 megabytes. The main reason for this is that the coordinator's mission does not include the process of generating a query plan or even executing sub-queries. However, it only helps in query plan execution and joins the final results, which leads to less consumption. In contrast, the memory space allocated to each SP is relevant to the amount of

data, the cause of which is managed locally without a remote resource server. However, all execution outcomes for sub-queries are saved remotely in the IPFS.

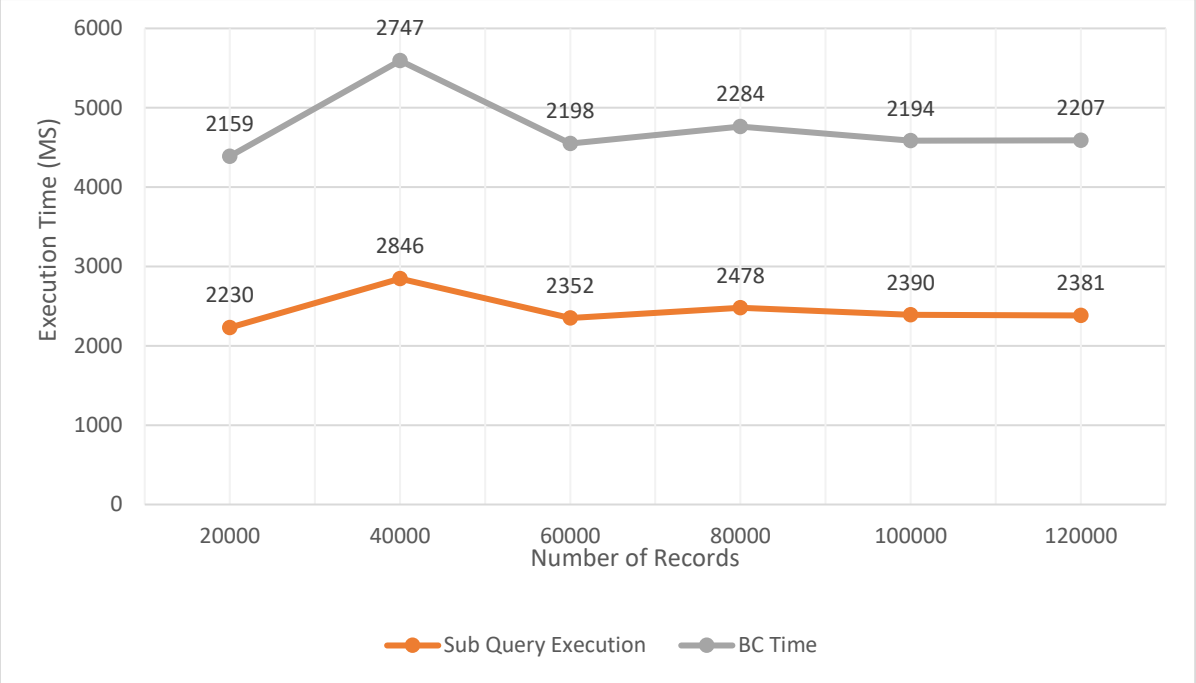

Figure 5.9: Performance Evaluation Results for SP 1 (Elapsed Times vs. Number of Records)

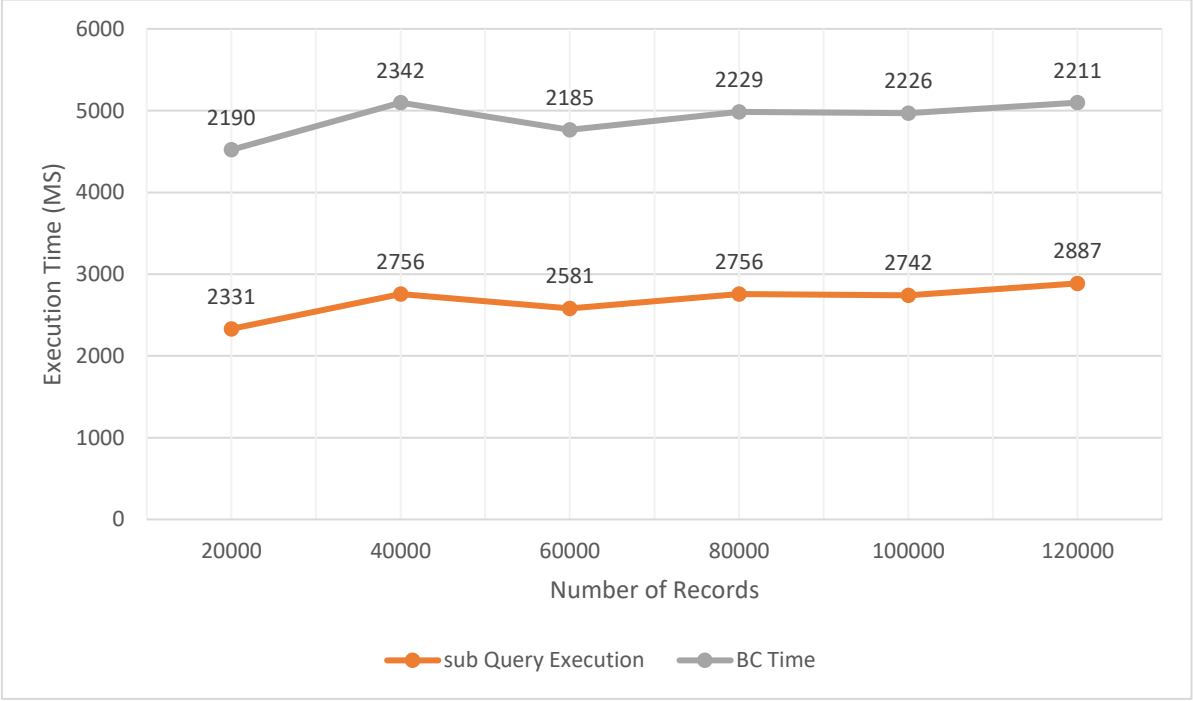

Figure 5.10 : Performance Evaluation Results for SP 2 (Elapsed Times vs. Number of Records)

Figure 5.11: Performance Evaluation Results for SP3 (Elapsed Times vs. Number of Records)

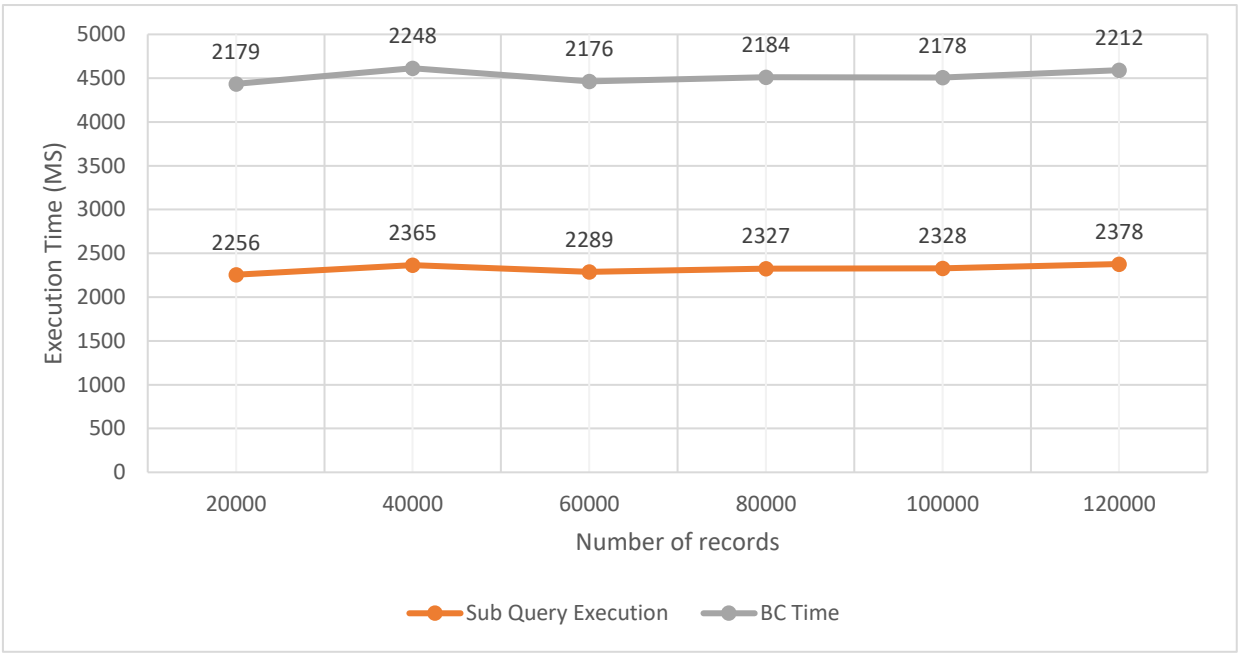

Figure 5.12: Performance Evaluation Results for SP 4 (Elapsed Times vs. Number of Records)

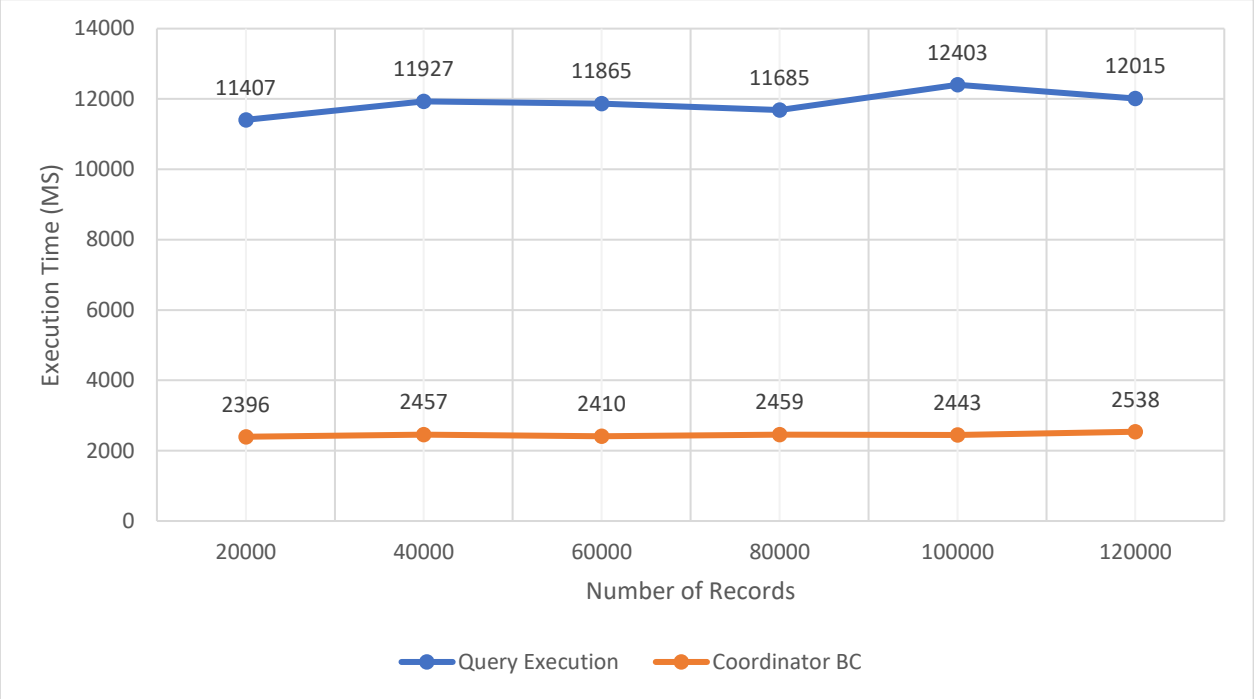

Figure5.13: Performance Evaluation Results for the Query Execution and the Coordinator (Elapsed Times vs. Number of Records)

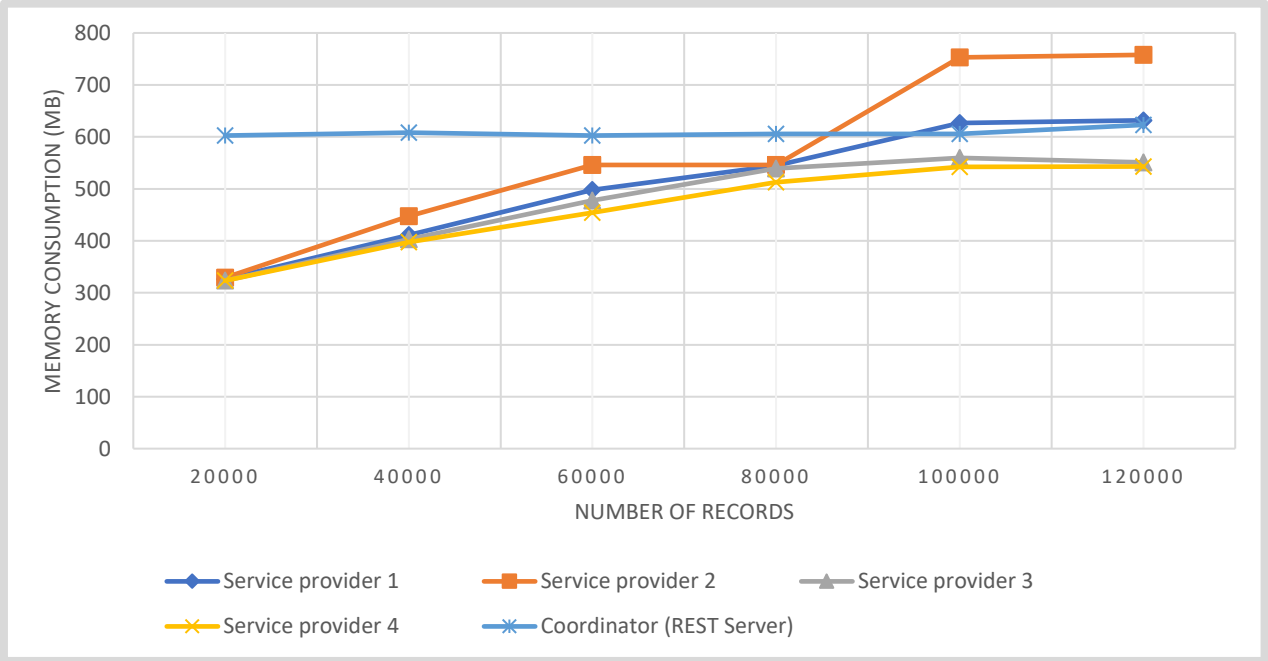

Figure 5.14: Memory Consumption Evaluations for the Service Providers and the Coordinator (Memory Used vs. Number of Records)

5.5 Security Analysis

A number of security analyses have been introduced to prove PrSchain's resourcefulness in tackling security and privacy issues related to service composition in a decentralized environment. Potential threats that could face the proposed solution are put forward, and suitable countermeasures are fostered to deal with them.

The distributed property of blockchain is one of a kind when safeguarding replicas of the ledger across multiple nodes, which leads to tampering with the ledger remaining impossible, resulting in numerous attacks. Using private permissioned blockchain enables maintaining participants' privacy by associating every entity with a public key that plays the role of pseudo-anonymity for the sake of hiding the real identity on the one hand, allowing the management of ledger access control and allowing only authorized members. In addition, to avoid data breaches, we opted to use a shared off-chain address among parent-child services, which means that only the true child can recuperate the data from off-chain storage. Moreover, for integrity goals, we kept the hash of the data in the -chain. We punctuate several assumptions that may bluster our method along with the appropriate solutions to thwart them while adopting a permissioned blockchain:

- ✓ **Assumption 1:** We assume that the coordinator attempts to tamper with the service composition or query plan by injecting new SPs or switching theSPs positions.
- ✓ **Resolution**: It is difficult for the coordinator to make any modification because it s entirely up to blockchain, which holds all data about service composition by ensuring their integrity.
- ✓ **Assumption 2**: A peer that participates in generating the plan attempts to make modifications by integrating anunauthorized SP.
- ✓ **Resolution**: It is possible for blockchain peers to make some changes by saving fault data into their local database; however, while invoking the chaincode, blockchain returns only the true service composition because Hyperledger Fabric uses thePBFT protocol to fight such attacks. However, all Blockchain implementations ensure data integrity unless, in the Hyperledger Fabric case, the membership service provider (MSP) undertakes controls the identities and access policies.
- ✓ **Assumption 3**: The service provider attempts to tamper with its own sub-query plan by adding a parent or child to the service composition.
- ✓ **Resolution**: Initially, the service provider cannot append parents because its executing turn starts once all parents have terminated. Then, on thechild side it is not allowed to

add any one because it is only allowed to invoke chaincodes that save the ciphertexts of its IPFS addresses encrypted via thepublic keys of its true children.

- **Assumption 4**: A service provider attempts to learn about the input data's owner for the sake to break the privacy rules. Consequently, the service provider's privacy can be disclosed.

✓ **Resolution**: It is not empowered to infer and link with the real owner regarding our strategy, which guarantees that each service provider has no idea about the composition of other participants.

- **Assumption 5**: After terminating the execution of all sub-queries, the coordinator is responsible for joining the final results, which leads to link the results with service providers; consequently, it can breach every service provider's privacy.

✓ **Resolution**: It cannot be linked because it holds only a private service composition that anonymizes the plan nodes. As a result, it can learn the relationship between the results and the final nodes with probability degrees.

- **Assumption 6**: Hyperledger Fabric peers attempt to obtain the ciphertexts of the IPFS addresses related to the sub-query outcomes, and they attempt to access the data result or tamper with their content.

✓ **Resolution**: We used asymmetric key cryptography to encrypt the results. Thus, to decrypt the ciphertexts, they need children's private keys, which results in blockchain peers not seeing or tampering with IPFS data.

- **Assumption 7**: In the case of decrypting the IPFS address and gaining access to a given sub-query result to manipulate it and affect all children that process the altered data.

✓ **Resolution**: In this method, the hash is used for the sake of maintaining data integrity, so every child has the opportunity to check the integrity using the digest generated by its parent, whether it carries on or stops the execution when it detects any suspiciousness. Consequently, this composition failed.

5.6 Discussion and Comparison

After evaluating the performance of PrSChain in terms of query execution, it is time to compare the proposed approach with related works in the same endeavor. As shown in Table 5.4, they suffer from numerous shortcomings that do not guarantee full security and privacy protection. We outline them as follows.

- Most of the proposed solutions consider centralization. In other words, they depend on a mediator to generate service composition and manage query execution. Although the mediator is trustworthy, there is a probability that it will exploit these responsibilities to tamper with a composition plan that requires a specific selected service provider.
- In the majority of studies, the participating service providers have learned about the generated plan, which enforces restrictions for authentication and maintaining privacy by using K-protection which could have a negative impact on terms of scalability by returning additional unnecessary values by the query.
- Access control and operation logs are not adopted for all the stated related works to maintain traceability for future audits and verifications.
- Most stated solutions exchange subquery results through a mediator. In this case, if the data were not secured, it would be easy to breach theparticipant data. Otherwise, if the exchanged data are secure and protected, such as byusing k-protection, the mediator can deal with the child service provider by sending raw data (without using K-protection).

According to Table 5.4, our method is compared to [46,47,48] regarding security and privacy requirements, starting with Tbahriti et al. [48], who pointed to improving service composition where a privacy model is frosted to check the compatibility between privacy policies and privacy requirements services. As per [48] the mediator is completely trustworthy for exchanging the intermediate data and without the need to employ any cryptographic method to secure the data. Otherwise, the coordinator exchanges the data ,we count on the blockchain for this mission.

Recently, [46] and [47] adopted K-protection to protect sensitive data from leakages. Bahramgi et al. [46] used OPES to encrypt only intermediate numerical data, whereas Tiwary et al. [47] employed a hash rather than encryption when the data were non-numerical. In the two solutions, employing K-protection lasts a significant time; it increases steadily with data size, and k becomes too large. In our situation, we initially address the issue of scalability by proposing blind service providers. Thus, a trustworthy blockchain ensures authentication among the participants. Another reason blockchain peers are in charge of generating and creating the composition plan thus tampering with service composition QoS is very low. In addition, we saved data hashes for integrity objectives and avoided data alteration. In [47], the authors declared the use of an in-memory table to treat the problem of re-executing the same query. Indeed, it does not object to privacy disclosure among service providers, particularly when k is extremely small.

Table 5.4: Comparative analysis between PrSChain with the state-of-the-art

Approach	[48]	[46]	[47]	Our Method
BC based	No	No	No	Yes
Decentralization	No	No	No	Yes
Data integrity	No	No	Yes	Yes
Mutual authentication	No	No	Yes	No
Access to intermediate data	Plaintext	Encrypt numerical data	Hash	Encrypt only the pointer
Service providers aware of other service providers in service composition	No	Yes	Yes	No
Privacy between service provider	No	Yes	Yes	Yes
Trust on mediator	Very high	Low	Low	Distrust

For instance, a service provider can retain in-memory tables of multiple queries, and it can learn about the shared values to obtain the real values of its service parent. We handle this when the service providers are unaware of the entire actual plan. Furthermore, when the query holds sensitive attributes, the works of [47] and [48] fail to obtain results owing to the mediator functionality of merging results, and they show only non-sensitive. In contrast, our work can easily handle the last issue regarding the coordinator being able to see each attribute without learning about its data owner. In recent studies, the mediator has been liable for generating service composition. In parallel, they declared it as untrustworthy. For example, the mediator can tamper with QoS by selecting SPs rather than others, which can improve the query execution. To avoid such issues, blockchain peers perform service composition and query generation without requiring a central entity.

5.7 Chapter Summary

In this chapter, the second objective is to preserve privacy in service composition using permissioned blockchain in the context of SOA through the proposed PrSChain design, which solves the issue of exposure of sensitive data among service providers, where the main idea

concentrates on eliminating trust in third parties and fostering blockchain for its high confidence and trustworthiness. This work targeted the composition plan generation and execution by Hyperledger Fabric peers, and for further improvement, the service provider's information and sub-query execution is managed via blockchain, which maintains its integrity and tamper-proof resistance. A service composition scenario is presented using a real-world data set that demonstrates the efficiency and resilience of all types and sizes of data. Finally, the proposed work is evaluated extensively in terms of service composition execution time and memory consumption. Compared with state-of-the-art methods, it outperforms recent works that last more time and memory consumption owing to k-anonymity, which increases both time and memory usage.

FLBCshard: a Scalable Blockchain-Based Federated Data Sharing

6.1 Chapter overview

This chapter discusses the improvement in blockchain scalability while preserving privacy in the federated learning paradigm by presenting the proposed framework called FLBCshard. It considers the problem of maintaining the model and participants' privacy while generating a learned model that is compliant with the task requester requirements. Therefore, an introduction to the new problem is presented in Section 6.2. Section 6.3 puts forward the system model and the problem formulation that covers the communication model, design description, and goals behind it. Following this, the FLBCshard workflow showcases the interactive actors and main steps in Section 6.4. Then, section 6.5 details the shard management and purification methods from malicious participants. Subsequently, to ensure a good system reputation, section 6.6 evaluates the participant contribution reputation. The performance evaluation and discussion are addressed using execution simulation in Section 6.7. Finally, Section 6.8 sums up this chapter.

6.2 Introduction

Federated learning (FL), a new breed of artificial intelligence, was developed by Google in 2016 [64] to address the concern of binding to the central mode of typical machine learning. This new paradigm is concerned with distributed collaborative machine learning that realizes privacy correspondence to international data protection regulations by building machines or deep learning models by learning local and private datasets without the need to share the participant's own data. However, the aggregation of large local parameters gives rise to critical issues, such as a single point of failure regarding over-dependence on a central aggregator, even though FL is a captivating paradigm in terms of tackling participant privacy issues. In the issue of having a single point of failure, the data may be inferred from local models, which threatens system security and participants ‘privacy, let alone trust dependencies and bottlenecks at the aggregator node [119]. Based on this, it is inevitable for decentralized federated learning to deal with the stated challenges by diminishing the risk of a single point of failure and tackling the bandwidth problem. Blockchain is a distributed database in which valid transactions are recorded in a chain of blocks that link cryptographically after reaching a consensus between network nodes.

Therefore, the blockchain maintains data integrity by duplicating blocks by every node. Many recent studies have used blockchain to maintain privacy and security in many research areas, such as data sharing [120], service composition [65], knowledge graph management [121], and even network countermeasure selection [122]. Blockchain decentralization is an attractive feature that drives the combination of the distributed database and federated learning [123,66] to gain secure decentralized, federated learning, eliminating several attacks, such as man-in-middle attacks. Indeed, combining blockchain and federated learning gives rise to new challenges by increasing additional communication and computation costs, scalability, and even rebuilding raw data from blockchain transactions (i.e., gradients). This is indeed a trade-off between learned model accuracy and clients' privacy preservation, which affects the overall model performance. Blockchain guarantees thatsecurity and privacy protection rely on the adopted consensus level, not the data privacy itself [102], which results in clients protecting the local updates for more security against network attacks. Integrating FL with Blockchain to ensure privacy does not ensure complete privacy preservation. Existing studies have proposed mixing with other effective cryptographic techniques, such as secure multiparty computation [49], two-phase secret-sharing-based aggregation [67], differential privacy [68,102], and homomorphic encryption algorithms [69]. Additionally, Blockchain nodes, after each transaction validation, are required to synchronize the ledger; the more the number of participant nodes, the less the blockchain response and performance decrease. As a result, the overall system scalability is affected. Another issue is that, after completing the FL task and issuing the global model, there is no guarantee of the task publisher's ownership, which creates intellectual property conflicts.

Therefore, we propose a decentralized privacy-preserving data model that shares the FLBCShard scheme for federated learning based on sharding blockchain, which performs FL tasks in a secure decentralized manner using IPFS and NFT-based data sharing to solve the above challenges by using smart contracts to manage model aggregation process. The contributions of our study are summarized as follows:

- Scalability: The proposed design solves the trade-off between scalability and privacy preservation. The sharding mechanism alleviates blockchain's consensus and ensures participant trust by removing potential malicious entities. The sharding mechanism reduces the communication costs by splitting the network into several groups and adopting a hierarchical architecture. The Raft consensus was adopted in the blockchain network because of its lower communication cost.

- Dynamic shard: Unlike previous static sharding methods, we use dynamic shard formation to improve privacy and security. The reformation process is based on several criteria, such as location, node reputation and contribution values, where nodes with low values can eliminated from shards. In addition, shards with a high model accuracy can cooperate with shards with a low accuracy.
- Reliability double check : the hierarchical design allows to decrease the communication with blockchain by delegating semi-trust node called proxy that would renders the FL trustworthy decentralized on one hand and deals with FL workers especially in large network(such as IOT),where these proxies act as honest but curious servers that assist in evaluating the workers(clients) before uploading the local trained model at this level a purification method is employed to eliminate stragglers and malicious client. The second contribution evaluation level occurs inter-proxies as a double check method as a collaborative evaluation by checking each model training accuracy based on test dataset where the Blockchain defines the most reliable to undertake the shard level local aggregation.
- Incentivizing mechanism: Before starting the FL process the task publisher defines a list of rules that includes the desired accuracy as well as the participant reward, in which each proxy selects the appropriate bid taking into account the worker resource capacity (memory storage, CPU, data quality).
- Model non-fungible token (NFT): To protect the intellectual property of the FL model, besides proving the ownership of the task publisher's model, a non-fungible token (NFT) is adopted to protect the model from digital theft.

6.3 System Model & Problem Formulation

As shown in Fig 6.1, our privacy preserving federated learning sharding blockchain model is constructed as a multi-layer hierarchical design to build a scalable and collaborative machine learning model to improve system performance. Our proposed system is subject to a few key specifications in terms of approving the learned models and the shard formation, as well as protecting the trained model property as proof of ownership using theNFT and potentially model trade.

6.3.1 Communication model

The FLBCshard design is built as a four-layer the hierarchical structure consists of the client, shard layer, global layer, and application layer at the top. The design consists of two chains: an S-chain and a G-chain. An S-chain is a sharding chain that maintains a participant's locally trained updates. Otherwise, the G-chain (global layer) coordinates between the task publisher and the FL participant by going through the shard chain and keeping the shards results.

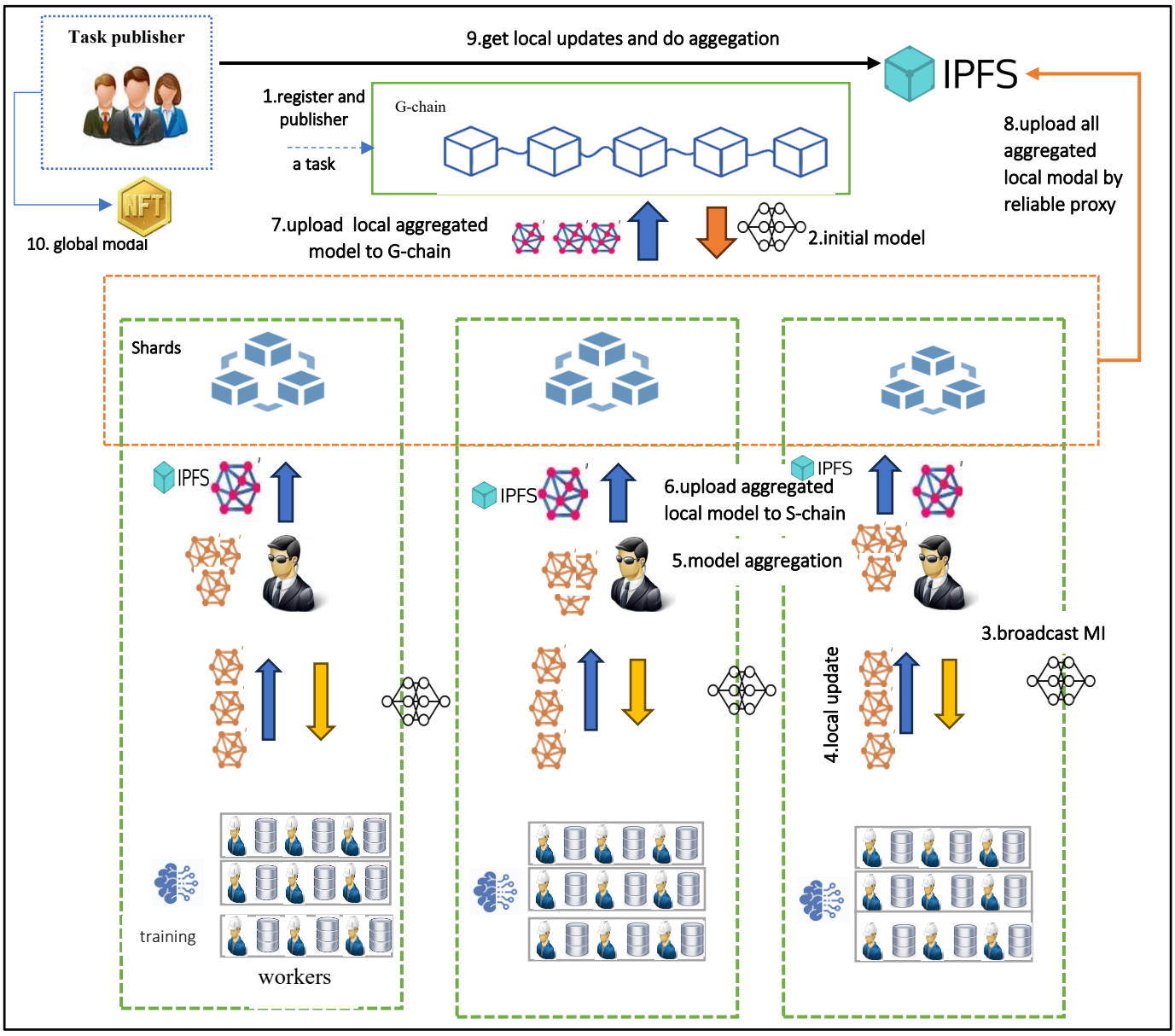

Figure 6.1: FLBCShard high-level system architecture

- **Client layer**

This layer joins the participating nodes that assist in producing FL tasks, in regard to the architecture, the participants fall into two categories: workers or proxy clients. Workers are

defined as network nodes that train their private local datasets regarding a private network (permissioned Hyperledger Fabric), which are registered and authenticated by the corresponding proxy before starting their tasks. Otherwise, Proxy Clients are qualified nodes and are considered as gateway that binds the workers and the shard-level chain S-chain, which per se reduces the communication costs, accelerate the authentication process (worker), eliminates suspicious workers, and even undertakes an outer aggregation. When the reference model is downloaded and local training is performed, each worker uploads the local update parameters along with additional information encapsulated in a signed transaction to its corresponding proxy client, which checks the model accuracy before transferring the aggregated updates to theS-chain.

- **Shard Layer**

This layer consists of a set of shards, each of which shard chain is defined by an individual chain that includes a finite number of peers, each of which is responsible for coordination between participants and the global chain by uploading the approved shard-level model. Blockchain peers play the role of either leaders, followers, or candidates according to Raft's consensus strategy. The S-chain leader retrieves the reference model from the G-chain to share the initial parameters with the Proxy Clients where they send it to related participants so that local training can start. The follower peers select their leader based on their reputation, which is calculated using the G-chain. The shard peers collaborate with the proxies to validate the local updates and compute the reputation of every proxy.

- **Global Layer**

The global layer includes the G-chain that maintains all the local approved models sent from each shard leader after completing every local training round. For this purpose, the G-chain employs the same Raft consensus by selecting a leader that coordinates the task requester and the federated learning framework in order to commit all the information in one block. It is obligatory to register each S-chain leader in the G-chain as a peer or even a candidate one more time as a leader. Each S-leader encapsulates the approval aggregated trained local model in the form of a signed transaction transmitted to the G-chain, which includes the G-committee that verifies the validity of each transaction before propagating the results with the G-leader, where the model accuracy pinpoints whether the transaction is approved or rejected. When the transaction is approved, the G-chain leader forms and appends a new block to the G-chain that belongs to one iteration of the FL task.

- **Application Layer**

It is an interface between the task publisher and the federated learning clients. The role of the application begins when a task publisher wants to build a collaborative federated learning model by launching a task proposal by interacting with the G-chain using smart contracts to customize each requester's needs and task requirements. Every interaction with the G--chain was logged and tracked. Therefore, the requester is a party of the network it registers and holds a digital identity to interact with the system. This layer also allows interaction with the off-chain distributed storage of the IPFS based on the NFT marketplace, which ensures data availability and is proof of ownership at the end of FL task the task publisher notifies in order to aggregate all the local updates to obtain the global model.

6.3.2 Design overview and main steps

Our design proposes the use of several actors ranging from participants to blockchain nodes where everyone has their responsibilities and will be punished for illegal processes. In the first step, the task publisher sends out to all proxy nodes a task request that contains model information, such as the desired accuracy and number of reward coins. The task publisher saves its request along with the initial model on the blockchain (global and local chain) by using smart contracts after valid authentication. In the second step, every proxy node creates a new training task (containing the initial model, which is obtained from its local chain) and publishes it to all registered participants to produce local updates. In the third step, after local training, the participation nodes send the results to their proxy node, where these later aggregates all received updates in one model update and send this latter along with additional information to all proxy nodes for validation. In the fourth step, every proxy node sends its validation results to its local chain for final validation by local chain peers. In these five steps, after receiving several validation results, every peer decides whether a given model update is valid by obtaining the number of positive validations. Therefore, if 2/3 of theproxy nodes have validated a given model update, then it is considered valid. Proxy nodes are rewarded or punished based on their validation results or model updates. Therefore, if a given validation result is considered invalid by a peer, every proxy node that has confirmed the correctness of this result will be punished. Otherwise, it is rewarded. In addition, proxy nodes that have sent a non-valid model update are punished by decreasing its reputation on its shard chain. A proxy node is selected from its shard according to its contribution value and computed reputation for aggregating valid model updates and storing the result on the shard chain. To get a global learned model, the more

reliable proxy node that has the maximum contribution score among all shards, it is qualified to aggregate all local model updates that have been stored in its shard chains and keep the outcome at global chain level. All steps are repeated until the target accuracy was achieved or all epochs were completed. After every epoch, a dynamic shard management process is performed to eliminate malicious nodes and improve system scalability (see section 5 for dynamic shard and reputation calculation).

6.4 FLBCshard workflow

Before initializing the system, the task publisher launches a request by proposing a task proposal to the G-chain, which is embodied as a smart contract to define and extract the task requirements and the necessary description. Regarding the network being permissioned, all participants were assigned a certificate authority from the Hyperledger fabric membership provider, including the task publisher, which forms a part of the process. The task publisher opts to share the parameters or relies on the G-chain to extract them. Before initializing the FL process the task publisher announces incentivizing contracts that steadily increase with the increase of model accuracy, then determining the federated learning task requirement and extracting the reference model parameters, this section elucidates the FLBCshard workflow, which includes the steps in which the network starts from scratch.

Algorithm 6.1 System Initialization

Input: Global chain $\boldsymbol{G}$, Number of proxies $\boldsymbol{P}$, number of workers $\boldsymbol{U}$, Task requester contract $\boldsymbol{TS}$, coordinates

Step 1: shard the global chain

split $\boldsymbol{G} = \sum_1^s \boldsymbol{S_i}$, s≥ $\mathbf{1}$

Step 2: Register proxies and allocate them to the appropriate shard

for each s shard ∈ $\boldsymbol{S_i}$do

for proxy $\boldsymbol{p} \in \boldsymbol{P}$ & $\boldsymbol{P} \neq \{\emptyset\}$ do

If $\boldsymbol{p} = \boldsymbol{accept\ contract\ TS}$ & distance$(\boldsymbol{s}, \boldsymbol{p}) = \left|\boldsymbol{C_s} - \boldsymbol{C_p}\right| \leq \boldsymbol{threshold}$ then do

Register proxy $\boldsymbol{p}$ into $\boldsymbol{s}$ and assign public key $\boldsymbol{k_{pub}}$ and private key $\boldsymbol{k_{priv}}$

end if

end for

end for

Step 3: Register workers and assign IDs

for proxy $\boldsymbol{p} \in \boldsymbol{P}$ & $\boldsymbol{P} \neq \{\emptyset\}$ do

for workers $\boldsymbol{u} \in \{\boldsymbol{1}, \boldsymbol{2}, \ldots, \boldsymbol{U}\}$ do

register $\boldsymbol{u}$ into $\boldsymbol{D}$ database and assign $\boldsymbol{ID_u}$

end for

end for

6.4.1 System initialization

Owing to the increased number of federated learning participants across the network, initially a number of shards $S = \{1,2, \ldots, N_s\}$ are formed where $|S| = N_S$ by taking into consideration several criteria such as the location of peers and participants in order to form a set of clusters. For the system setup Algorithm 6.1 a registration process is launches in a Blockchain FL network a set of proxy are denoted by $P = \{1,2, \ldots\ldots, N_p\}$, where $|P| = N_p$, represents the total number of proxies are registered into blockchain network by generating a certificate authority (public and private key) for proxies interested in participating in FL selecting the appropriate task requester contract (accuracy and reward). Then each blockchain shard $S_i = P_c$ where P_c is a subset of proxy nodes of the total set P allocated regarding their location coordinates. In the next step each proxy undertakes the process of recruiting a set of worker devices U has local dataset DS_u where $U = \{1,2,3, \ldots, N_u\}$ and $|U| = N_u$ regarding their resource consumption (CPU and memory storage).

6.4.2 FLBCshard One-Epoch execution

After forming a number of shards by clustering the participants based on their coordinates, each shard peer selects a leader peer (S-leader) among peer candidates by running the Raft consensus. Every blockchain shard receives the initial model M_I^0 parameters from G-chain via

the joined peer, each legitimate proxy notifies to download the untrained model in order to start the FL epoch.

Step 1: Local model M_I^0 training

Regarding the hierarchical communication, the shard S_i distributes the initial model to all legitimate proxies in order to start the local training operation, the delegated node (proxy) starts to authenticate the legitimate workers according to the registration dataset, after a successful process the proxy p_i redistributes the same parameters ω_i and the training starts off, where each worker u_i trains the local dataset $DS_i$for the aim to minimize the cost function $f(\omega_i, DS_i)$ and adjusting the learning rate η where the local model parameters ω_I^L is calculate at the global epoch L and the local I iteration as follow:

$$\omega_i^{(L,I)} = \omega_i^{(L-1,I)} - \eta \nabla f\left(\omega_i^{(L,I)}; DS_i\right) \qquad 6.1$$

Step 2: Local Aggregation and Contribution, Reputation Evaluation

After local training each worker communicates its local updates with the proxy ,for security objective the workers add a deferentially private noise to the private model (at SDG optimizer), in which at proxy level a secure aggregation based differential privacy is performed by adopting the FedAvg [124] Algorithm 6.2 to obtain a local aggregated model $\sum_i \omega_i$.

Algorithm 6.2 FedAvg

Input: $\boldsymbol{K}$ clients indexed by $\boldsymbol{k}$, $\boldsymbol{B}$ local minibatch size, $\boldsymbol{E}$ the number of epochs, $\boldsymbol{n}$ is the learning rate

---

***Server Executes*:**

Initialize $\boldsymbol{w_0}$

for each round 1,2….. ***do***

$\boldsymbol{m} \leftarrow \mathbf{max}(\boldsymbol{C}.\boldsymbol{K}, \mathbf{1})$

$\boldsymbol{S_t} \leftarrow (\boldsymbol{random\ set\ of\ m\ clients})$

for each client k ϵ $\boldsymbol{S_t}$ in parallel ***do***

$\boldsymbol{w_{t+1}^k}$ ←ClientUpdate (k,$\boldsymbol{w_t}$)

$\boldsymbol{w_{t+1}} \leftarrow \sum_{k=1}^{K} \frac{\boldsymbol{n_k}}{\boldsymbol{n}} \boldsymbol{w_{t+1}^k}$

ClientUpdate *(k,w)*

ß← (split $\boldsymbol{p_k}$ into batches of size B)

for each local epoch i from 1 to E **do**

for batch b ϵ ß **do**

$\boldsymbol{w \leftarrow w - n\nabla\, l(w; b)}$

return $\boldsymbol{w}$ to server

After the aggregation the proxy starts to define the evaluation metric in order to purify the network from malicious clients, for the proxy side it receives a statistical information such as the training data size $|D_k|$, training time T_k and computation resource RC_k of each worker in the form of a message through applying the equation 6.2 [71] the contribution is calculated in the followings:

$$C_k = \alpha * |D_k| \qquad (6.2)$$

Where $\alpha$is a predefined coefficient in [0,1].

The proxy in its turn starts the evaluation process computing the exact participant contribution. In this situation, it employs equation 6.3 [71] that leads to check the validity of the training time by depending on thedata training size and computational resources as follows:

$$T_k = {(|D_k| * \mu)}\Big/{RC_k} \qquad (6.3)$$

where μ denotes the number of CPU cycles required to train one data unit; however, the T_k validity indicates that other information is also considered valid .

The worker reputation is computed by its related proxy node based on its valid data contribution and accuracy. To define the accuracy improvement for a given participant, the local update was tested against a data set that distributed by the task requester. In this case no improvement occurred in addition to a decrease in accuracy resulting in the contribution decreases. Otherwise, the participant is rewarded with R_k by increasing its contribution to obtain the reputation, which is calculated as follows:

$$R_k = C_k * \left(1 - \beta * (ACC - ACC_k)\right) \qquad (6.4)$$

Where ACC is the desired accuracy related to the proxy node, ACC_k is the worker's accuracy and $\beta$is a predefined coefficient in the range [0,1].

If the FL training epoch contains several iterations, the final reputation of a given worker k in an epoch e is calculated as follows:

$$RP_{ek} = \sum_{n}^{i} R_{ei} \qquad (6.5)$$

Where R_{ei} is the worker's reputation in an epoch e during iteration i.The reputation value is considered very important in detecting whether a reliable participant or not using a predefined reputation threshold.

Step 3: Collaborative proxy contribution evaluation and Reputation

Each aggregated local model goes through an evaluation before passing to the next stage the global aggregation, it is known that each proxy bonds to one shard which it plays the role of delegated server to communicate with the S-chain for the aim to reduce the communication overheads. For security requirement each proxy p_i safeguards its result (model parameters) into the IPFS along with the correspondent hash (equation 6.6) for integrity and scalability purpose and adds a calibrated noise to the aggregated trained model, then the pointer is uploaded into the shard S_i besides to the generated hash that plays the role of a model identifier in the form of a transaction (equation 6.7), this operation is realized by all proxies in the same shard.

$$h_i = hash(\omega_{pi}) \quad 6.6$$

$$TX_{p_{iS_i}} = (h_i, pointer_{\omega_{pi}}, Sig) \qquad 6.7$$

Where $pointer_{\omega_{pi}}$ is the IPFS pointer of the file which holds the local aggregated model of proxy p_i, and Sig denotes the digital signature of $(h_i, pointer_{\omega_{pi}})$ together.

At this moment all the transactions are deployed at S-chain level, and regarding that the network is synchronized the evaluation will be done in parallel manner, thus the evaluation step is launched in a decentralized and collaborative way by employing differential privacy between proxies by adding a particular noise, and launching the evaluation based upon a test dataset the results decides whether the neighbor model is reliable or not. Figure 6.2 illustrates the main steps of the evaluation process. After the evaluation each proxy gives an assessment value about the model utility(accuracy) and save it at S-chain level, the peers decide approved or rejected model compared to the aggregated local update of each p_i.

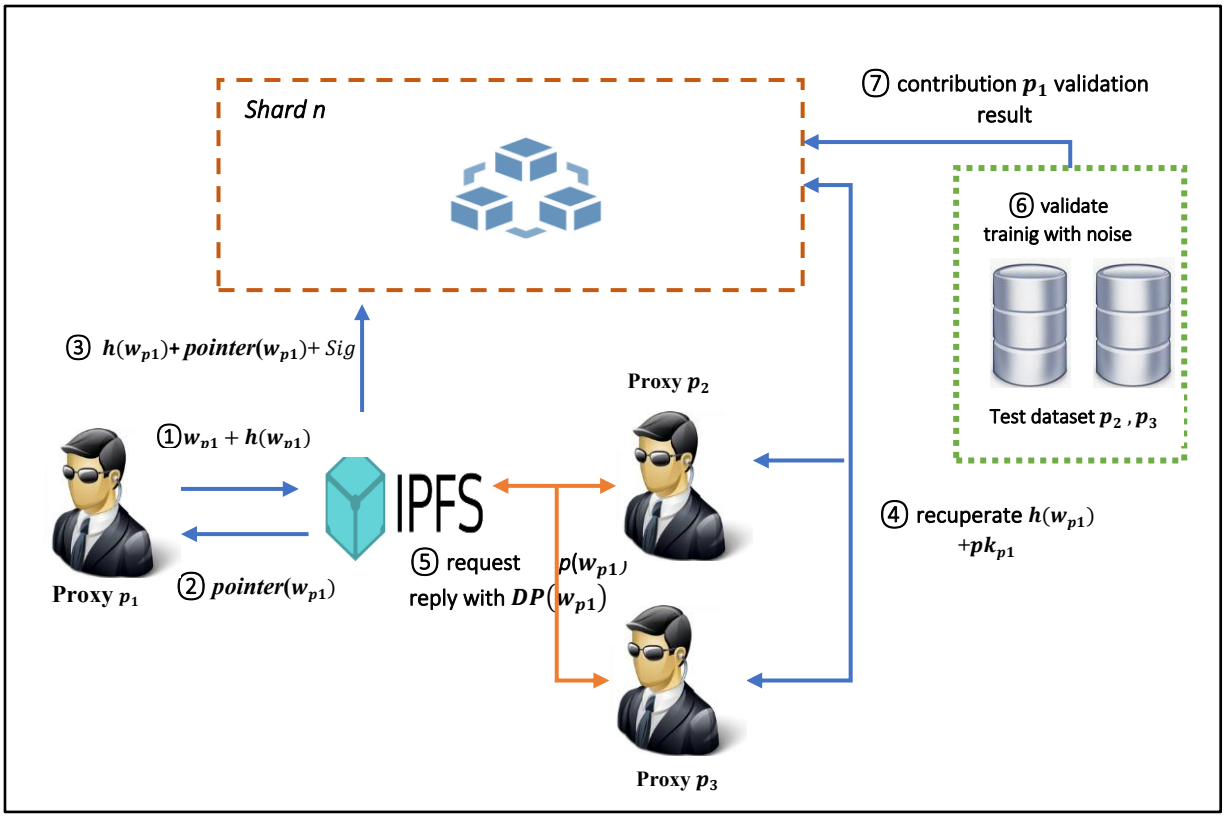

Figure 6.2 proxy node contribution validation

After evaluating the local models, it is time to calculate p_i reputation based on its contribution in order to reward reliable and honest proxy nodes for their contributions to the FL process to encourage them to work efficiently and honestly, which would increase the concurrence between them. The contribution value is mainly calculated from the accuracy improvement of the aggregation result as follows:

$$RPr_k = \rho * (ACC - ACC_i) \qquad (6.8)$$

Where $\rho$is predefined coefficient in [0,1].

The final reputation of a given proxy node i in an epoch e is then calculated using the following equation:

$$RP_{ei} = \sum_n^j RPr_{ej} \qquad (6.9)$$

Where RPr_{ej} is the proxy's reputation in an epoch e during iteration j.The reputation value is considered very important in detecting whether a given proxy node is malicious by using a predefined reputation threshold for theproxies.

Step 4: Shard level Aggregation

After terminating all the FL rounds each S-chain holds the valid trained model ID along with the correspondent reputation score RP_{ei} for each proxy, the S-chain peers are empowered to select the more qualified and reliable proxy to accomplish the shard-level aggregation using a chaincode as a trust third party the trusted proxy p_{tr} applies the same aggregation algorithm FedAvg under a secure manner equation 6.10 where the p_{tr} has no information about the neighbors updates owing to differential privacy.

$$p_{tr}, \{P_i\} \rightarrow \sum_i^n DP(\omega_i) \quad 6.10$$

After reaching a consensus (Raft) the S-chain leader in its turn committed all the local updates as well the aggregated shard level model in one block and broadcasts the results to all the peers of the same shard where the shard level block structure illustrates in Figure 6.3.

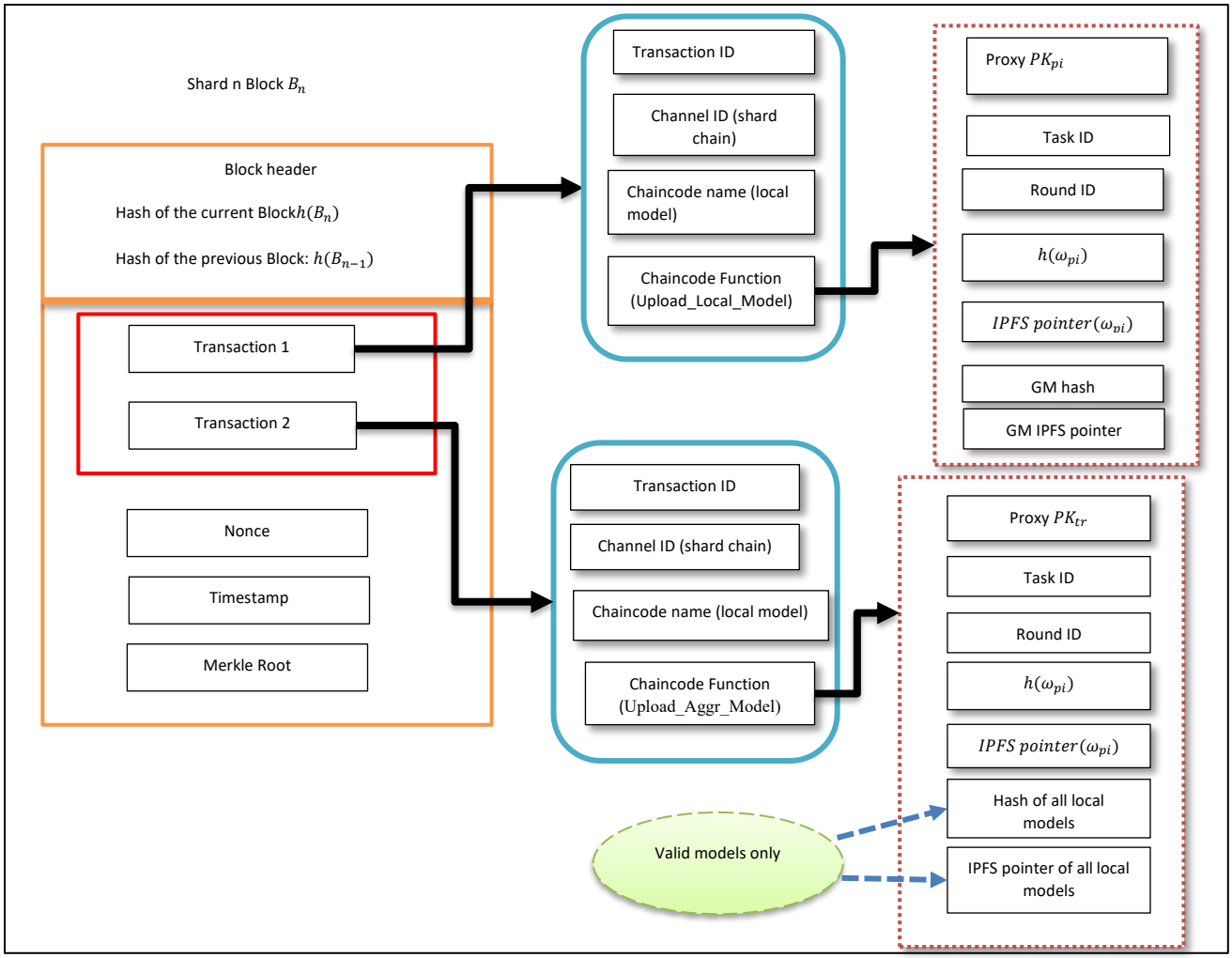

Figure 6.3 Shard chain Block components for two transactions about a proxy model and an aggregated shard model

Step 5: Global level aggregation

This step starts after each S-leader peer broadcasts the shard generated block to all peers of the same committee, regarding that the task requester has already allowed to access to all S-chains, when the last shard appended the block, the task requester download all valid aggregated local updates from IPFS to aggregate them in order to produces a global model, owing that G-chain is not empowered to aggregate the results where this process is performed by the task requester which is assumed to be a trustworthy entity that undertakes this operation. The G-chain main role is an interface between the task requester and FL participants (proxies and workers). Afterward the global aggregation process is launched knowing all the aggregated local models has evaluated and validated in trust manner. The global model faces the same fate of the initial model steps until the federated learning converges and reach the desired model utility by the requester where Fig 6.4 details the task requester transaction.

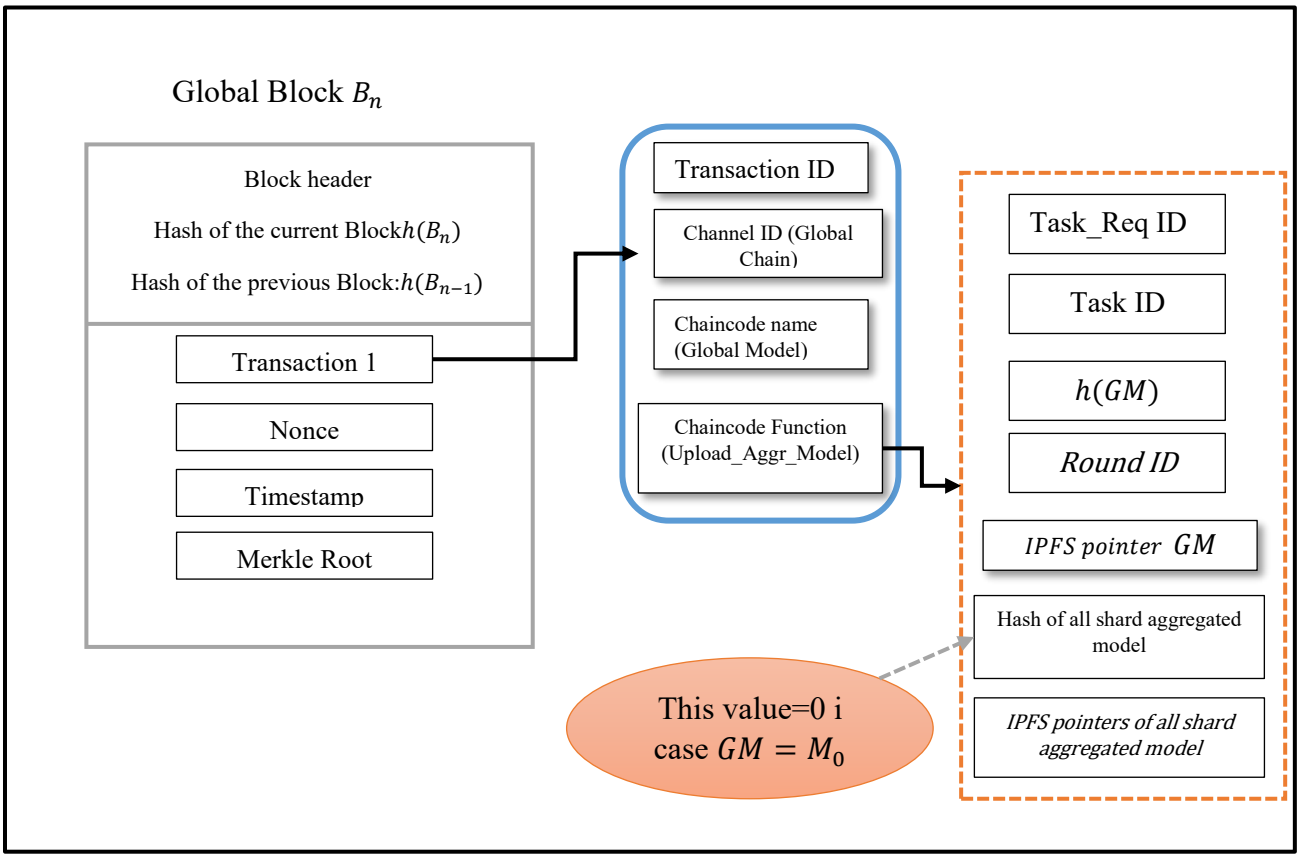

Figure 6.4 Global-chain Block Structure with one transaction which upload a global aggregation model.

Step 6: NFT-Based Model Sharing and Trade

This task belongs to the task publisher, where a non-fungible token is generated using the NFT marketplace as a proof of ownership where the purpose behind is to trade with the model instead of requesting it from blockchain which could help for future profit.

6.5 Dynamic Shard Management

Shard management is an extremely important and dynamic process. Its main goal is to eliminate malicious nodes and improve the scalability and accuracy of the system. Three main adjustments are applied to shards that have specific properties: low accuracy, long aggregation time, and ahigh fraction of malicious nodes. Therefore, Algorithm 6.3 demonstrates the dynamic management of a set of shards by splitting, merging, or eliminating proxies according to the system decisions made by analyzing the current shards. In the following, we describe the dynamic operation in detail.

Algorithm 6.3 Shard management

Input: Set of shards S, threshold on max number of proxies $maxPxTh$, threshold on shard throughput $TgpTh$, threshold on number of stragglers $StTh$, threshold on min number of proxies $minPxTh$, threshold on max shard model accuracy $mxAcTh$, threshold on min shard model accuracy $mnAcTh$

Output: New set of shards S'

1	**var** st_i is the set of proxies that are stragglers in a shard s_i;
2	**var** thg_i is the throughput of a shard s_i;
3	**var** Sp is the set of splitted shards of S;
4	**Var** acc_i is the local model accuracy of a shard s_i;
5	**Var** $accP_i$ is the local model accuracy of a proxy p_i;
6	// Split each shard that meet specific criteria into sub shards
7	**for** (shard s_i in S) **do**
8	**for** (proxy p in s_i) **do**

9	**if** p is marked as straggler proxy **then:**
10	append s_i to St_i;
11	**end for**
12	**if** ($\|s_i\| > \boldsymbol{nbPxTh}$ and $\|St_i\| > \boldsymbol{StTh}$ and $thg_i < \boldsymbol{TgpTh}$ and $acc_i < \boldsymbol{mnAcTh}$) **then:**
13	Remove st_i from s_i;
14	split s_i into new set of shards $Spl_i = \{s_j \mid \|s_j\| < \boldsymbol{nbPxTh}\}$
15	append all shards in Spl_i to Sp
16	append st_i to Sp
17	**end if**
18	**end for**
19	**var** Sn is the set of non splitted shards of S;
20	**var** Sm is the set of shards ready to merged;
21	$Sn = S - Sp$
22	**for** (shard s_i in Sn) **do**
23	**if** ($\|s_i\| < \boldsymbol{minPxTh}$ and $thg_i > \boldsymbol{TgpTh}$ **and** $acc_i > \boldsymbol{mxAcTh}$) **then:**
24	append s_i to Sm;
25	**end if**
26	**end for**
27	**var** Smg is a set of shards;
28	**var** $Sfmg$ is the final set of merged shards;
29	// Merge shard that meet specific criteria into new shards
30	**for** (shard s_i in Sm) **do**

append all proxies in s_i to Smg;

if ($|Smg| > minPxTh$) **then**

Create a new shard sd;

append all proxies in Smg to sd

clear Smg

append sd to $Sfmg$

end if

end for

$Sr = S - (Sp \cup Sfmg)$ // The rest of shards without merging or splitting

// Reassign proxies with higher accuracies into low accuracy shards

var Px is a set of proxies with max accuracies;

var Sl is the shards with lower accuracies;

for (shard s_i in Sr) **do**

if ($acc_i > \boldsymbol{mxAcTh}$) **then:**

append p_j with $max(accP_j | P_j \in s_i)$ to Px;

if ($acc_i < \boldsymbol{mnAcTh}$) **then:**

append s_i to Sl;

end for

for (shard s_i in Sl) **do**

Pick one proxy p_j from Px;

append p_j to s_i;

remove p_j from Px;

end for

54	$S' = Sr \cup Sp \cup Sfmg$; // The final set of shards
55	return S';

6.5.1 Shard Split

The main goal of this process is to split a given shard into two or more shards according to specific criteria. After a system epoch, all shards must be analyzed, and a shard split process is applied when the following conditions are satisfied:

- The number of participants was higher than the predefined threshold. In this situation, if a high number of participants have not sent their updates, this will delay the aggregation process and, therefore, affect the accuracy of the global model.
- The number of updates was below a predefined threshold. This situation can have a negative impact on global aggregation. In this situation, there are several straggler proxies that have not send theirs results.
- The delay in receiving the local updates aggregation by the related proxy node
- The shard throughput was below a predefined threshold. In this situation if the throughput is low, some participants do not correctly participate in the FL process.

6.5.2 Shard merge

The main goal of this process is to merge two or more shards into one shard according to specific criteria. After every system epoch, this process can be applied in parallel with a shard split process. The target shard must fulfill some properties to be merged with other shards that are also targets of theshard merge. An example of a target is a shard with a number of participants below a predefined threshold. In this situation, there is a very rapid aggregation time, and these shards are a good target for malicious nodes. The merging process is applied when the following conditions are satisfied.

- The number of participants is below a predefined threshold.
- The aggregation time is below a predefined threshold.

6.5.3 Reassign proxies

This is another strategy that is applied to participants by reassigning them to other shards or proxies. For example, in a given shard, some proxy nodes can have very good accuracy values

and other proxies can have a low accuracy value. In this situation, a set of participants good contributions is selected from the first type of proxy. Subsequently, its members are distributed to the second type of proxy. This last strategy can improve the accuracy and increase the concurrence between participants related to the same proxy, thus improving the overall global model accuracy.

6.5.4 Eliminate Malicious Proxies

The system can detect and remove malicious proxies according to the reputation evaluation of the proxy nodes. The main goal of this type of proxy is to perform several types of attacks, such as poisoning attacks, which try to perturb the training process with poisoning data to decrease the overall accuracy. The aggregation results of every proxy node are evaluated during the validation step using other proxies. Every proxy node is punished for poisoning attack detection.

6.5.5 Eliminate Malicious Participants

Participants' behaviors were evaluated using their contributions and interactions with their related proxies. If the negative behavior of a given participant is detected, it results in the removal of the latter. The evaluations of participants are performed by their proxies, and therefore, the latter can be negatively influenced by malicious participants if the latter are not detected.

6.5 Implementation

FLBCShard is implemented using Eclipse and various Java APIs and Python modules for instance JSON, PyTorch, CUDA, Fabric SDK, OpenFL[6] for federated learning, Opacus[7] for differential privacy. Figure 6.5 illustrates the architecture of the framework which demonstrates the fundamental components:

- The crucial component is Hyperledger Fabric[8], which is used as permissioned blockchain that allows only legitimate proxy nodes which they hold valid certificates to interact with FLBCShard. Owing to the property of private network, Hyperledger Fabric manages client's

[6] https://openfl.io/
[7] https://opacus.ai/
[8] https://hyperledger-fabric.readthedocs.io/en/release-2.3/

identity through utilizing certificate authorities. The network is configured for eight organizations and one peer node for each and one Couch database, this latter is used as a world state database and one ordering service. The network was built with one certificate authority for each organization. We created nine channels one for global shard named "G-chain" and one for each shard named "S-chain". We associated each channel with one chaincode (smart contracts) which they are deployed using the Go language, figure 6.6 exposes the HLF network architecture used by FLBCShard, where every channel is associated with its ledger and smart contract. The HLF network uses the Raft consensus as fast consensus for all chains to ensure rapid transaction processing and block generation and storing.

- Moreover, the Interplanetary File System (IPFS) [132] is defined as a distributed file storage, where each added file to IPFS has a unique address that is derived from a hash of the entire file's content. In our proposal we opted to use IPFS as off-chain storage for the sake to safeguard the temporary data that are produced during FL iteration. Our main aim is alleviating the block size which obviously leads to guarantee Blockchain scalability.
- Python clients (Workers) are implemented using Python, each one trains, and submits a model to its proxy for aggregation. OpenFL [70] was used as the federated learning framework.
- The proxies are created using Java, each connected to a S-chain of the HLF network. Local models were received from python clients.

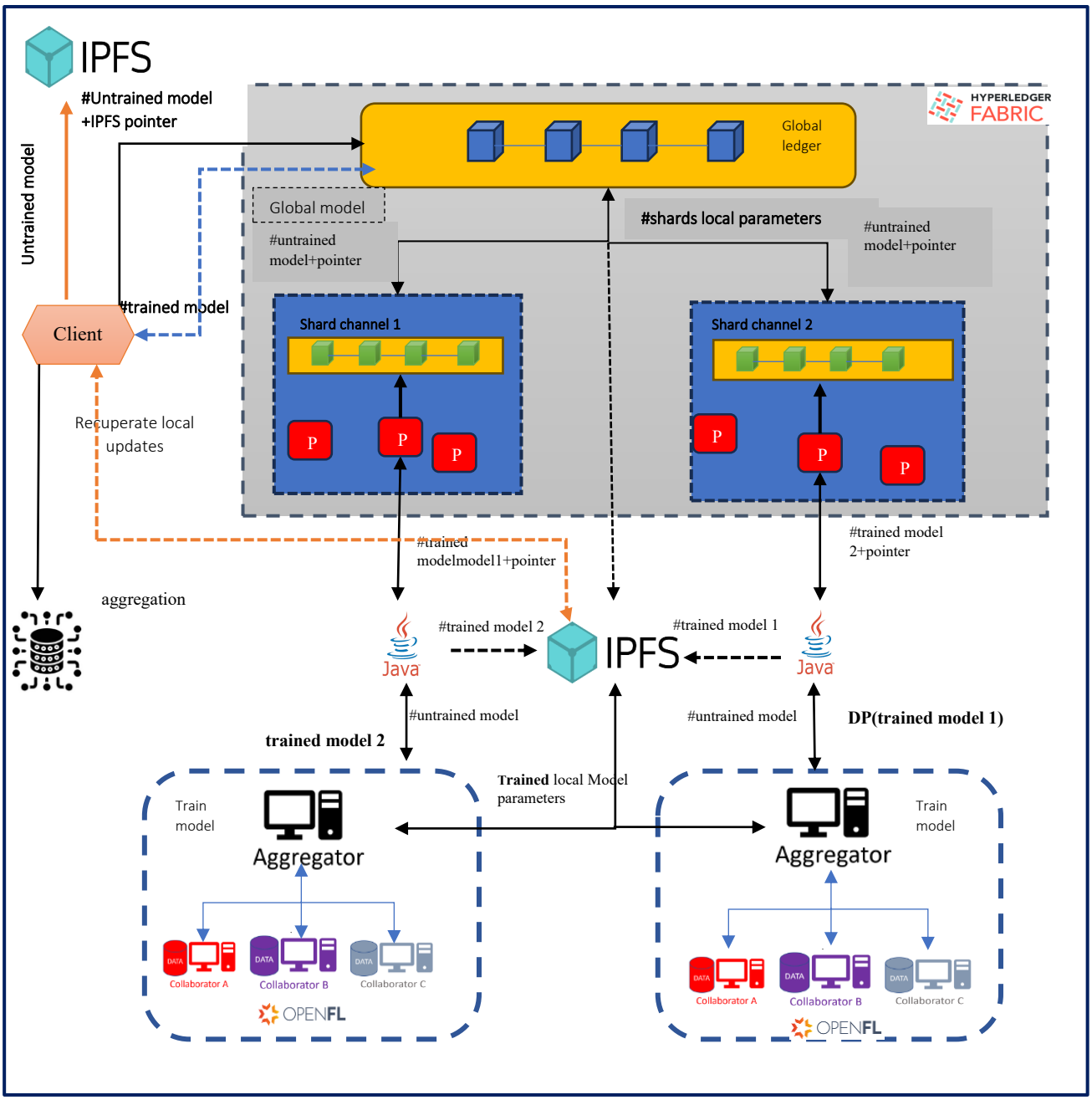

Figure 6.5: Implementation architecture of FLBCShard that contains two proxies and six workers

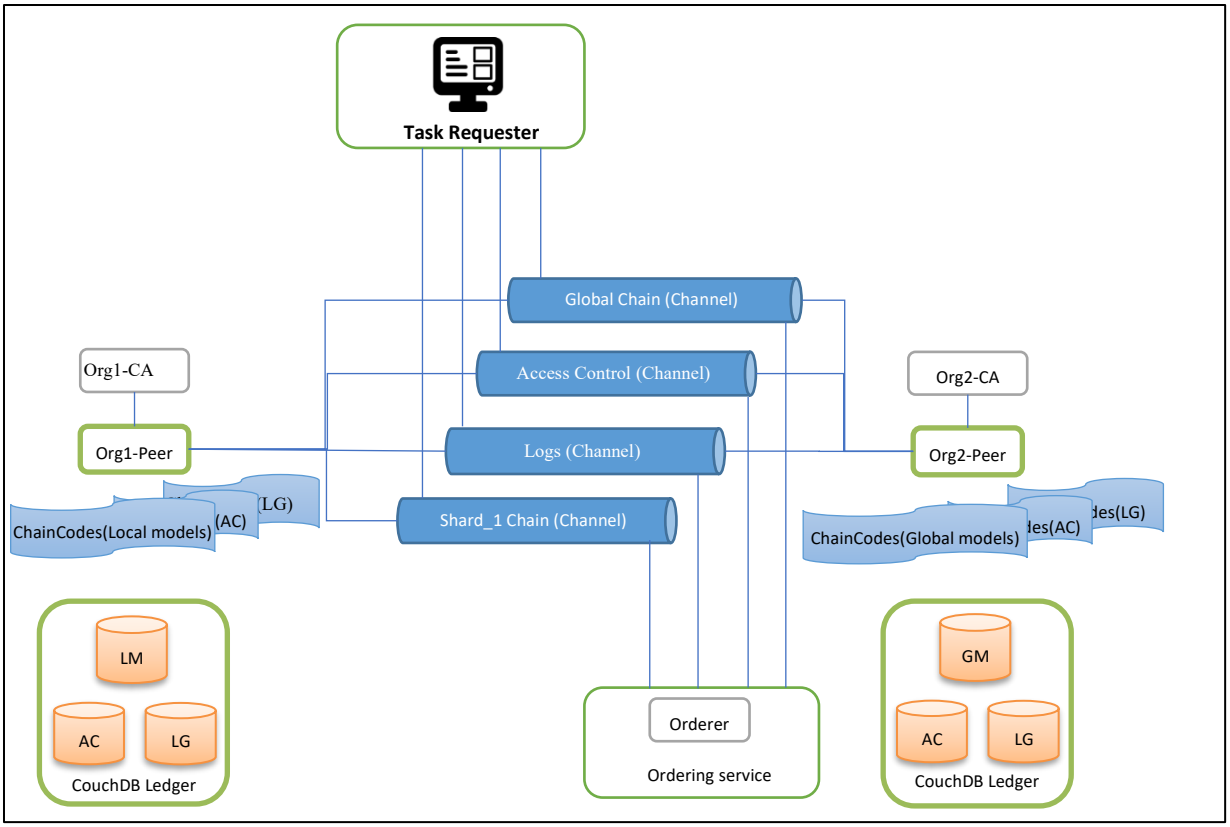

Figure 6.6. The Hyperledger Fabric Network Used by FLBCshard.

6.5.1 Fabric Chaincodes and distributed ledgers

In Hyperledger Fabric every peer has a copy of the ledger (local database), which brings together all the valid transactions that executed by the network via chaincodes. Which resulting, that each peer can have many installed chaincodes for one channel. HLF ledgers are updated using smart contracts regarding external Blockchain users claims (such as task requesters and proxy nodes). Our work allows using four distributed ledgers, each one is associated with one or more chaincodes and multiple peers. Those ledgers store data concerning FLBCShard transactions, such as global models, local models, aggregated shard models, access controls and operation logs. The next subsections discuss in detail the deployed chaincodes that belongs to each ledger from the four ledgers that stated before.

6.5.1.1 Global model Chaincode

The chaincode that belongs to global chain channel, it defines some functions that are executed by Hyperledger Fabric peers, in order to manage data related to global models that are registered by the Blockchain. The chaincode holds the name “Global model”, and it is associated with the global chain local ledger that stores information. The GM chaincode supports the Golang

structure, illustrated by Listing 6.1 where some GO functions are provided by the table 6.1 along with restricted access.

```
type Global_Model struct {
        TaskID string `json:"TaskId"`
        TaskRequesterID string `json:"TaskRequesterID"`
        Hash string `json:"Hash"`
        IPFS_Adr string `json:"IPFS_Adr"`
        LocalModels [] LModelsList `json:"LocalModels"`}
type  LModelsList struct {
        Hash string `json:"Hash"`
        IPFS_Adr string `json:"IPFS_Adr"`
    }
```

Listing 6.1. The Golang Structure used by the Global Model Chaincode

Table 6.1. Some Smart Contract Functions that are Implemented by the Global Model Chaincode

Function	Description	Restricted Access
Request_Task	Request a new federated learning task. The response contains information such as: task identifier, number of shards, proxies' public keys.	Task Requester
Upload_Initial_Model	Upload the initial model to the IPFS and save its hash and IPFS address on the global chain	
Upload_Aggregated_Model	Upload the aggregated model to the IPFS and save its hash, IPFS address and information about all used local models on the global chain	
Get_Latest_Agg_Model	Request the latest aggregated model from the global chain. The response contains information as it is given by listing 1.	

6.5.1.2 Local Model Chaincode

The Local Model chaincode determines the functions that are executed by HLF peers for managing the local models uploaded by proxy nodes. The chaincode is installed into the S-Chain channel. The LM chaincode uses the Golang structure which is demonstrated by the listing 6.2, whereas table 6.2 presents some GO functions along with restricted access.

```
type Local_Model struct {
        TaskID string `json:"TaskId"`
        Round_Number string `json:"Round_Number"`
        ProxyID string `json:"ProxyID"`
        LM_Hash string `json:"LM_Hash"`
        LM_ IPFS_Adr string `json:"LM_IPFS_Adr"`
        Global_Model_Hash string `json:"Global_Model_Hash"`
        Global_Model_IPFS_Adr string `json:"Global_Model_IPFS_Adr"`
    }
type Validation_result struct {
        TaskID string `json:"TaskId"`
        Round_Number string `json:"Round_Number"`
        ProxyID string `json:"ProxyID"`
        Res_LocalModels [] LModelsList `json:"Res_LocalModels"`}

 type  LModelsList struct {
        Hash string `json:"Hash"`
        IPFS_Adr string `json:"IPFS_Adr"`
        Validation_Result string `json:"Validation_Result"`
   }
```

Listing 6.2. The Golang Structures used by the Local Model Chaincode

Table 6.2. Smart Contract Functions for local model

Function	Description	Restricted Access
Upload_Local_Model	Upload a local model to the IPFS and save its hash and IPFS address on the shard chain associated with the proxy node that has invoked the function.	Proxy nodes
Upload_Aggregated_Model	Upload the aggregation model that created from the set of valid local models to the IPFS and save its hash and IPFS address on the shard chain associated with the proxy node that has invoked the function.	Proxy node with max reputation
Get_Aggregated_Model	Request the aggregated model that are created by aggregating all valid proxy local models in a given shard chain.	Task Requester, Proxy nodes
Get_Local_Models	Request a set of all local models from a shard chain by given a round number.	Proxy nodes
Send_validation_results	Send the validation results of all local models in a given shard chain.	Proxy nodes

6.6 Experiment Configuration

A number of experiments were performed to validate the functionality and test the performance of our approach. Experiments were performed on a machine with an Intel Core i7 processor running at a1.8 GHz clock speed, 16 GB memory, 128 GB SSD, and 1 TB for storage.

In this evaluation, our focus is not on improving the model accuracy but on testing whether the sharding solution can affect the overall evaluation. We considered federated learning under different evaluation settings by using several evaluation values of local rounds, batches, shards, poisoning ratios and differential privacy parameters. The table 6.3 givens the set of parameters along with their notations which are considered in FLBCShard evaluations.

Table 6.3: Evaluations settings

Parameter	Symbol	Task

Dataset	D	MNIST
Dataset size	$\|D\|$	60000
Model	w	CNN
Number of participants	n	20
Learning rate	μ	0.01
Number of shards	S	{1,4,10}
Loss function	ℓ	Cross Entropy Loss
Number of proxies in shards	Ps	1
Evaluation metric	Ac	Accuracy
Mini batch size	B	{16, 32, 64, 128}
Local rounds	E	{1, 2, 4, 6}
Malicious participant ratio	Po	{0, 0,25, 0.50, 0.75, 1.0}
Threshold accuracy for poisoning detection	$ThPoi$	0.60
Noise multiplier	Nm	{0.01, 0.05, 0.15, 0.25, 0.50}
Sample rate	Sr	1.0
Clip norm	Cn	1.0
Clip frequency	Cf	1.0
Delta	Dt	0.0001

6.6.1 Datasets

The learning evaluation was conducted using real-world datasets, that are widely used for data classification, such as the MNIST [125] dataset, to evaluate the performance of FLBCshard. The dataset consisted of 60,000 28× 28 images of handwritten digits. The IID data were divided uniformly among FL workers, and the data were distributed between participants in character or digits writers.

6.6.2 Results

With regard to the blockchain and centralized setting, there is little difference between the accuracy and training loss when the number of shards is increased. Therefore, blockchain integration has no effect on model performance.

Figure 6.7 presents results about evaluating FLBCShard global model accuracy with respect to the number of local rounds and fixed batch size B=64. This shows that the testing accuracy of the global model increases by increasing the number of rounds.

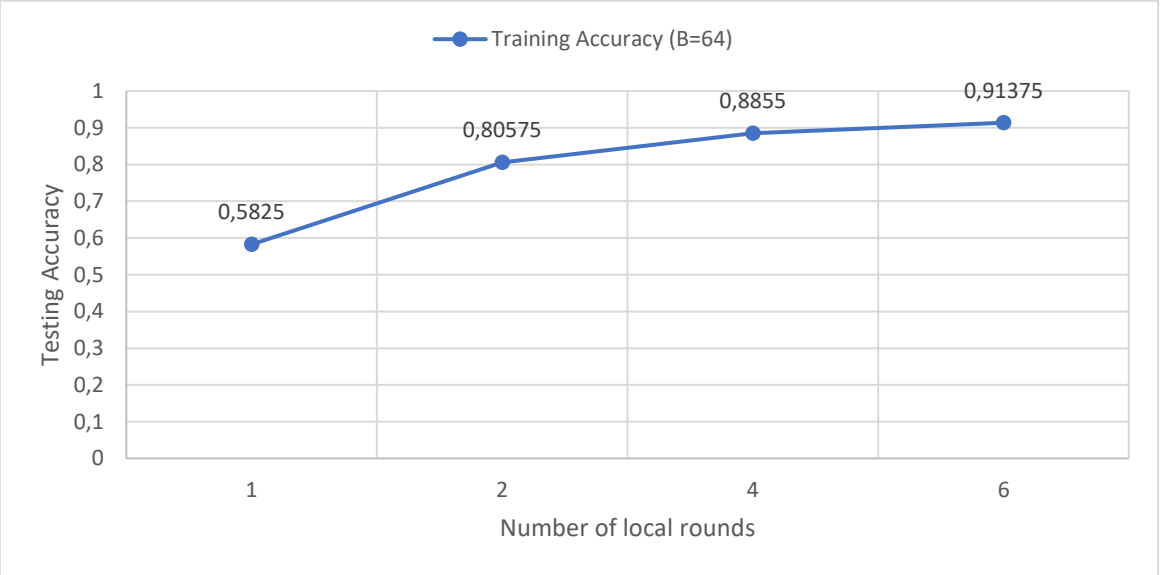

Figure 6.7: Testing accuracy vs. Number of Local Rounds

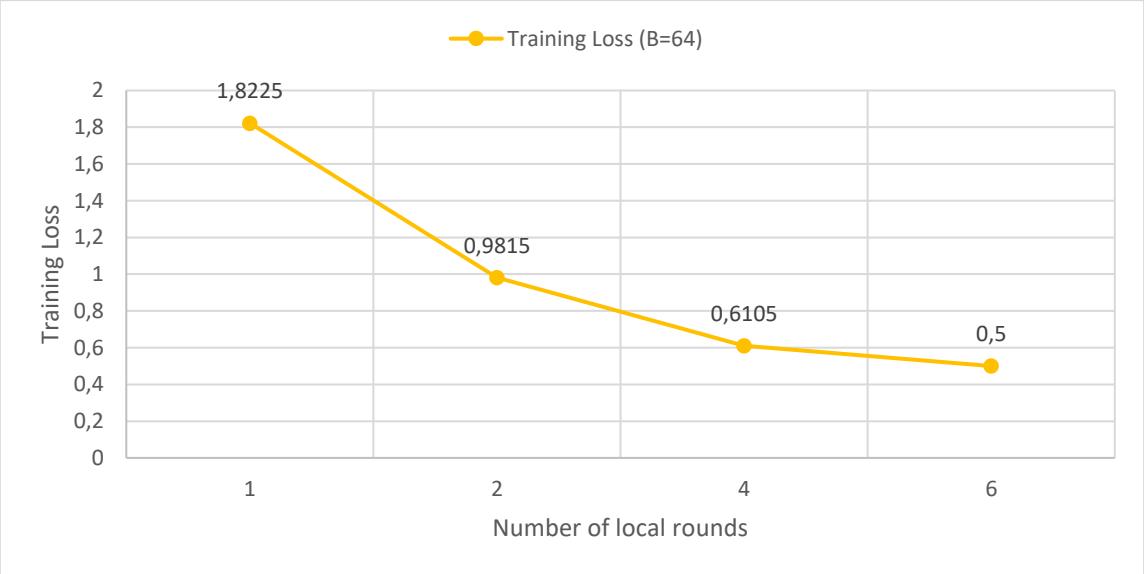

Figure 6.8: Training Loss vs. Number of Local Rounds

We evaluated also the training loss values by using the same previous setting, where the results are given by figure 6.8. We can observe that the training loss is decreased with respect to increasing number of local training rounds.

In the next setting, the number of local rounds is fixed (E=2) by varying the batch size from 16 to 128. Figure 6.9 show that the global model accuracy is decreased after increasing the batch

size. For the training loss evaluation, the figure 6.10 illustrates the results and it is observed that the loss values increased with increasing values of batch size.

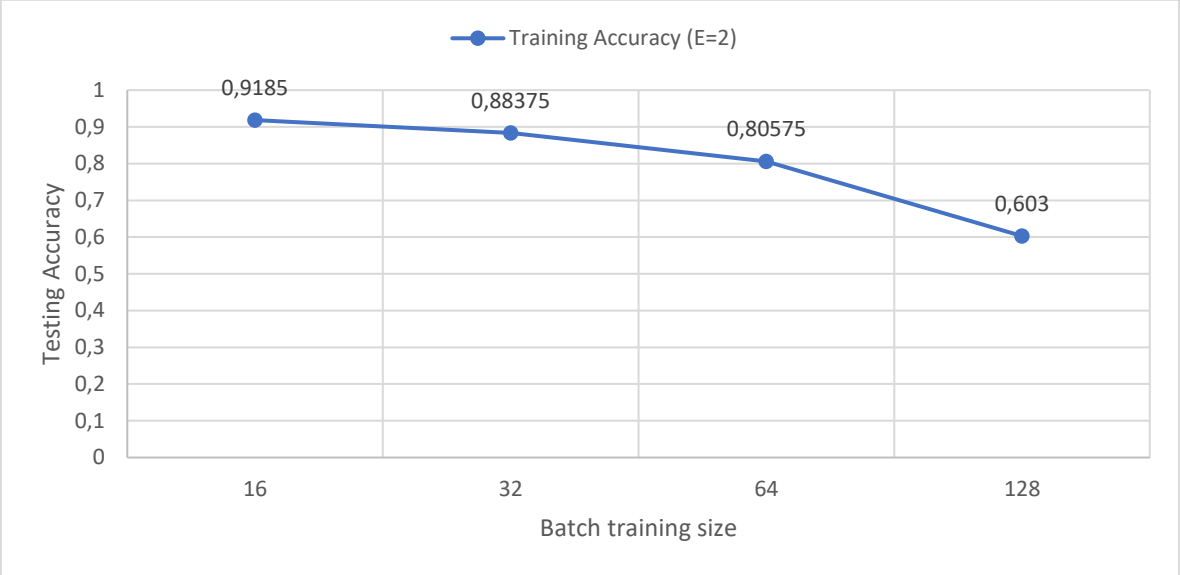

Figure 6.9: Testing accuracy vs. Batch Size

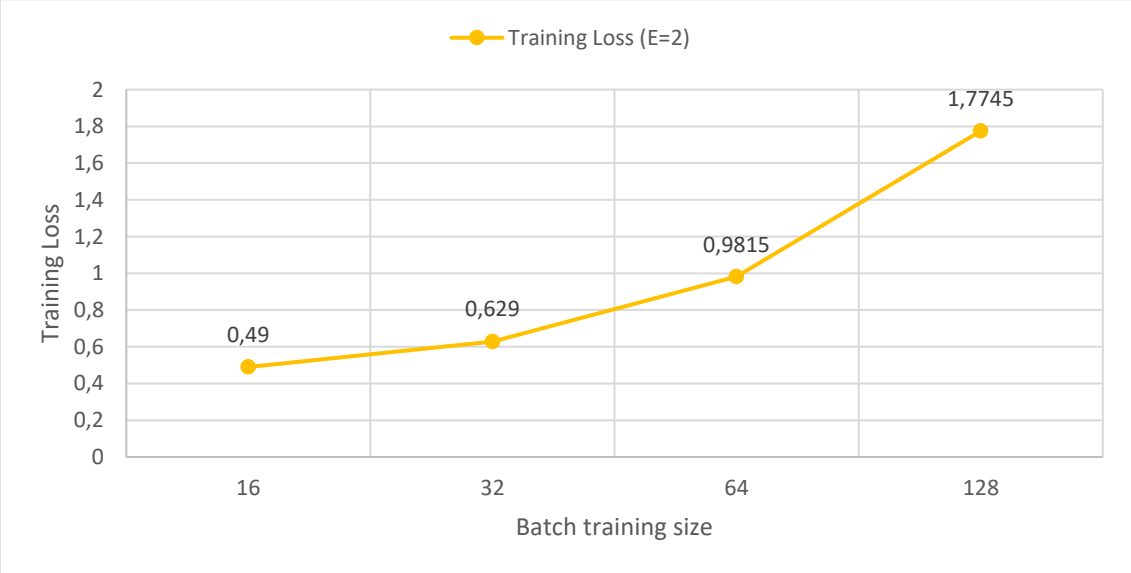

Figure 6.10: Training Loss vs. Batch Size

FLBCShard is tested under differential privacy settings by varying the amount of noise during training and after global model aggregation. The effect of noising on global model accuracy is given by figure 6.11 where increasing values of noise can affect negatively the model accuracy.

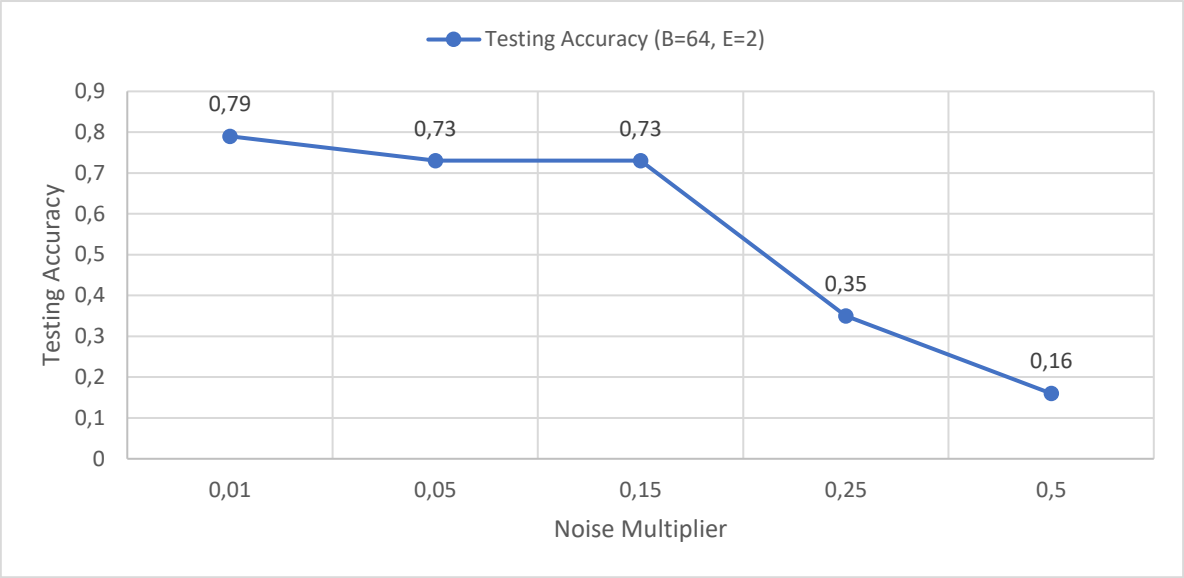

Figure 6.11: Testing accuracy vs. Noise Multiplier

In contrast to model accuracy evaluation, the training loss values increased with increasing noise values as it is given by figure 6.12.

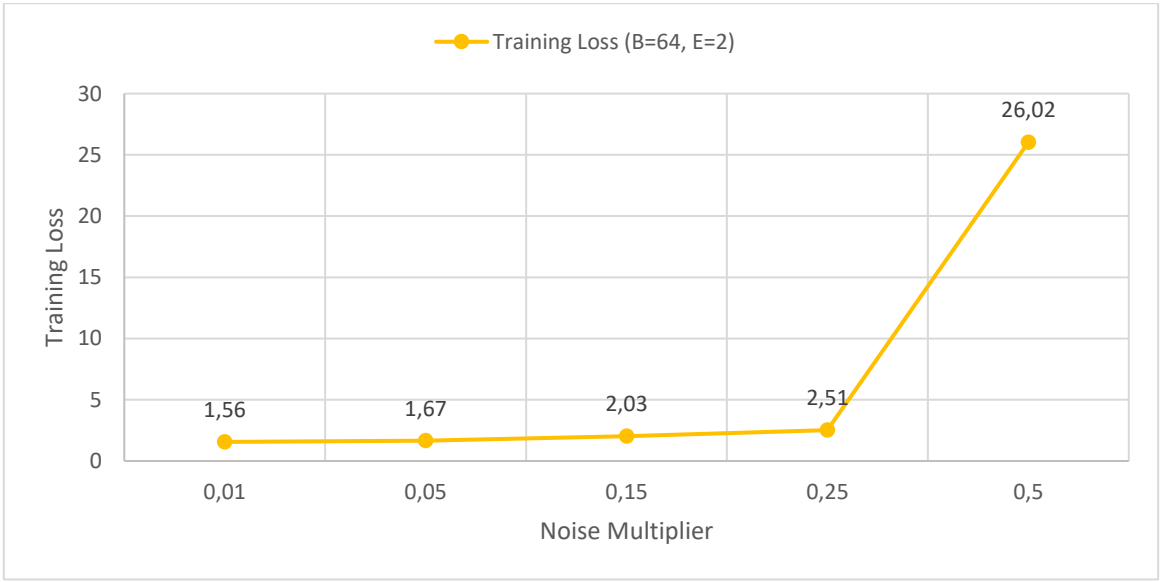

Figure 6.12: Training Loss vs. Noise Multiplier

The epsilon values of differential privacy are also captured with varying noise values. The results are illustrated in figure 6.13. the epsilon values are decreased with increasing in noise values

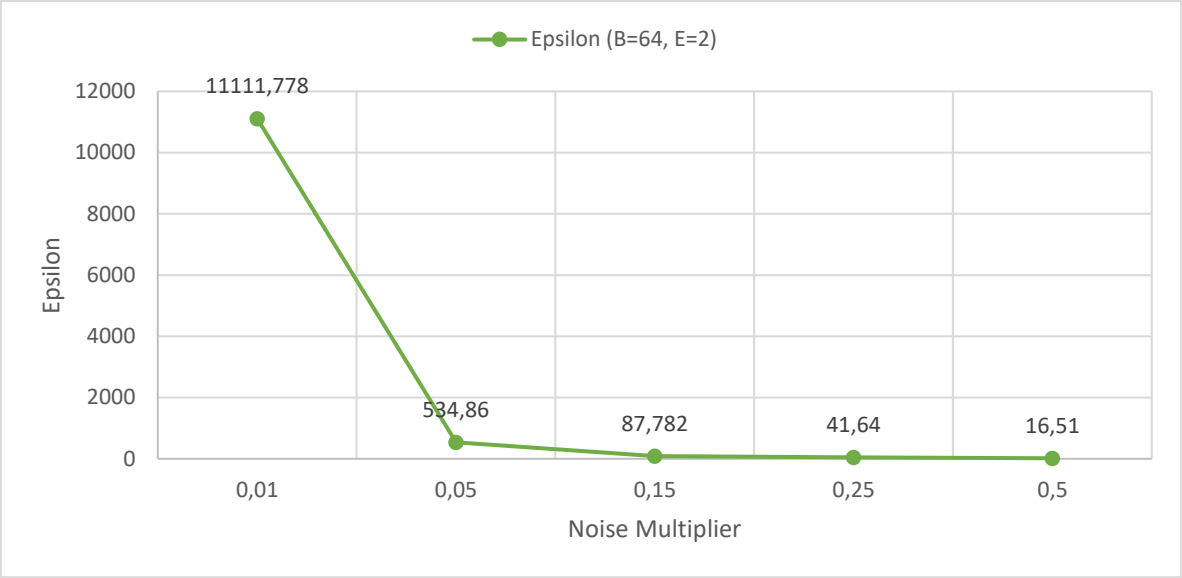

Figure 6.13: Epsilon vs. Noise Multiplier

In addition to global model accuracy values given by figure 6.7, we evaluate FLBCShard without poising attack detection by varying the malicious participant ration from 0% to 100%. FLBCShard uses 0.60 as an accuracy threshold for detecting poisoning models that are sent by malicious participants. Figure 6.14 shows that the testing accuracy of the global model is decreased with respect to increasing ratio of malicious participants.

In contrast, training loss values is increased with respect to increasing ratio of malicious participants as presented in figure 6.15.

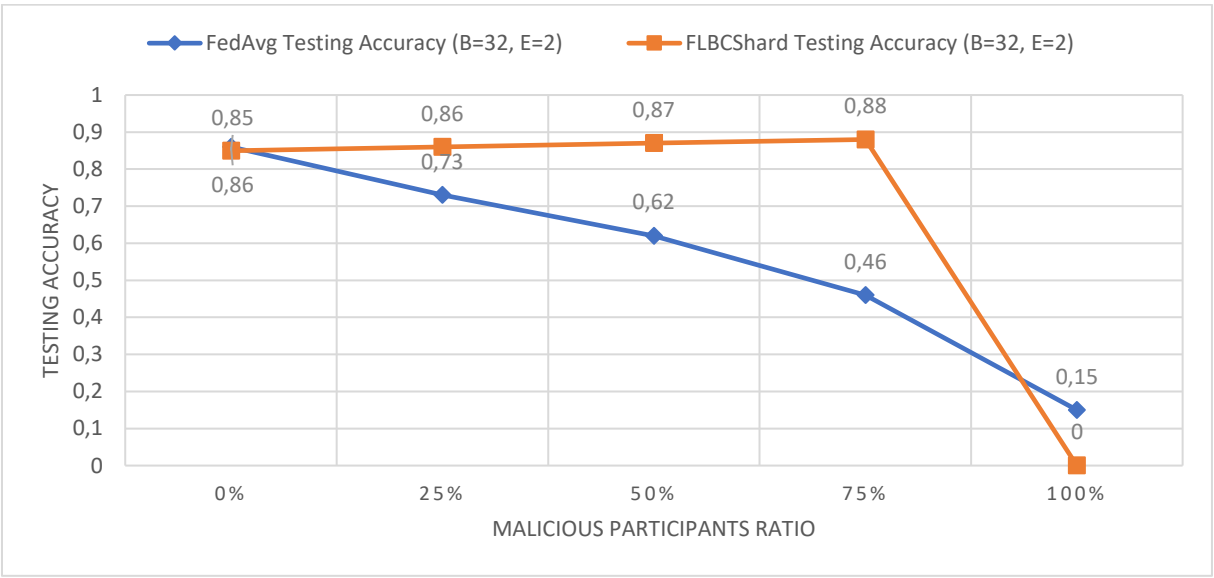

Figure 6.14: Testing accuracy vs. Malicious Participant Ratio

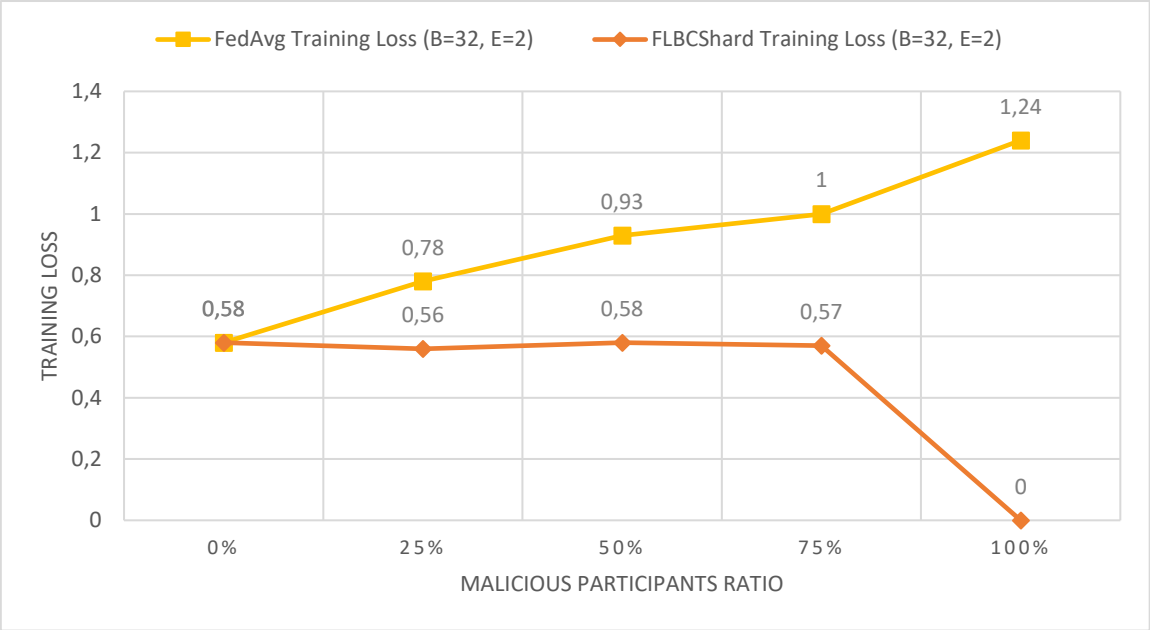

Figure 6.15: Training Loss vs. Malicious Participant Ratio

Fig 6.16 shows how the number of shards can affect the model accuracy for FLBCShard. It is observed that the number of shards can increase accuracy because additional shards provide more proxies and participants.

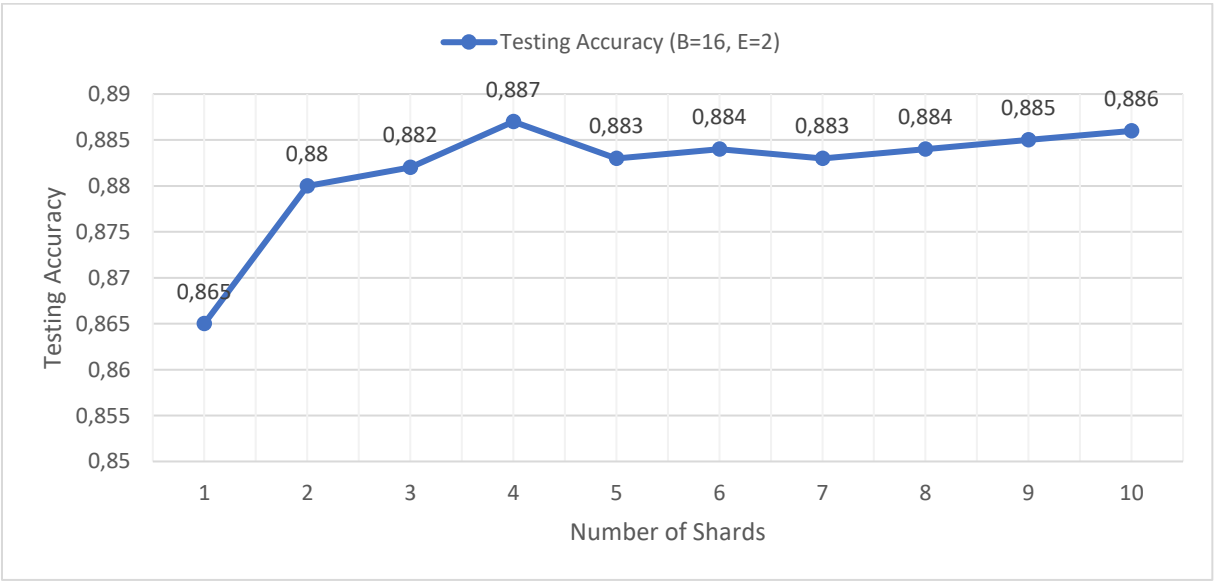

Figure 6.16: Testing accuracy vs. Number Shards

6.6.3 Blockchain Network Performance Evaluation

In this subsection, we evaluate the performance of the Hyperledger Fabric network of FLBCShard using Hyperledger Caliper[9] with a Linux open-source benchmarking tool. Caliper perform several tests to measure the performance of the BC network throw predefined configuration. The performance results contain several metrics such as system latency and throughput (TPS). We test the HLF network using different configurations with a varying number of shards, workers, transactions count (txcnt) and transactions per second (tps). The target smart contract is the local model chaincode and its function Upload_Local_Model (see table 6.2) which store the local shard models on the BC ledger. Therefore, the caliper workers send transactions by invoking the later function where everyone contains data about a generated local model. Table 6.4 shows the different parameters and their values supported by the testing benchmark.

Our objectives in this testing HLF network performance in order to get the followings:

1. **System throughput:** The number of transactions processed per second. In other words, the number of proxy local updates treated per seconds by HLF network.
2. **Average system latency:** The duration of a transaction to be processed and committed to the HLF ledger. Therefore, measuring the time of a proxy local update to be treated and saved onchain.
4. **System stability:** The consistency and reliability of the system in terms of transaction processing and maintaining the integrity of the ledger.
5. **System scalability:** The ability of the system to handle increasing amounts of transactions as the number of users and assets on the network grows.
6. **Transaction count:** The number of transactions that can be processed and committed to the ledger without fail.

Table 6.4: Configuration parameters of Caliper tests and their values

Parameter	Values
Shards	{1,2,3,4,5,6,7,8,9,10}

[9] https://github.com/hyperledger-caliper/caliper

Workers	{1,2,3,4,5,6,7,8,9,10}
Transactions Count (TXCNT)	{100,200,300,400,500,600,700,800,900,1000}
Transactions per second (TPS)	{5,10,15,20,25,30,35,40,45,50}

6.6.3.1 Caliper Test Results

In the first step, the HLF network is evaluated with a varying number of shards and 4 workers under 200 transactions and 30 transactions per second. In every step, the HLF is tested with number of shards ranging from 1 to 10. The figure 6.17 shows that the system throughputs are increased with an increasing number of shards, so the HLF network treats transactions rapidly in case of sharded blockchain where the global network peers are distributed across shard chains. Therefore, the FLBCShard can treat and save in fast manner the proxy nodes local updates that are sent to shard chains.

In addition to system throughput, the Caliper tests capture the system response latency and computes theirs averages under the same previous parameters. Figure 6.18 presents the average latency evaluation results where there is no big difference between these values, they are ranging from 2 to 5,5 seconds.

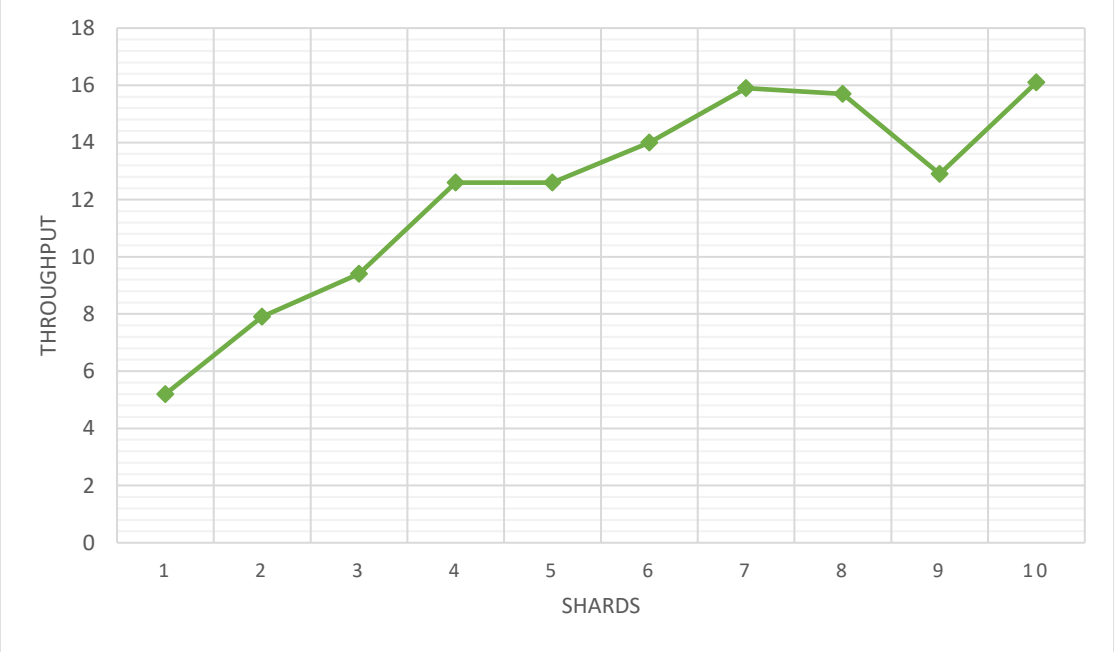

Figure 6.17: Throughput (in seconds) vs. number of shards

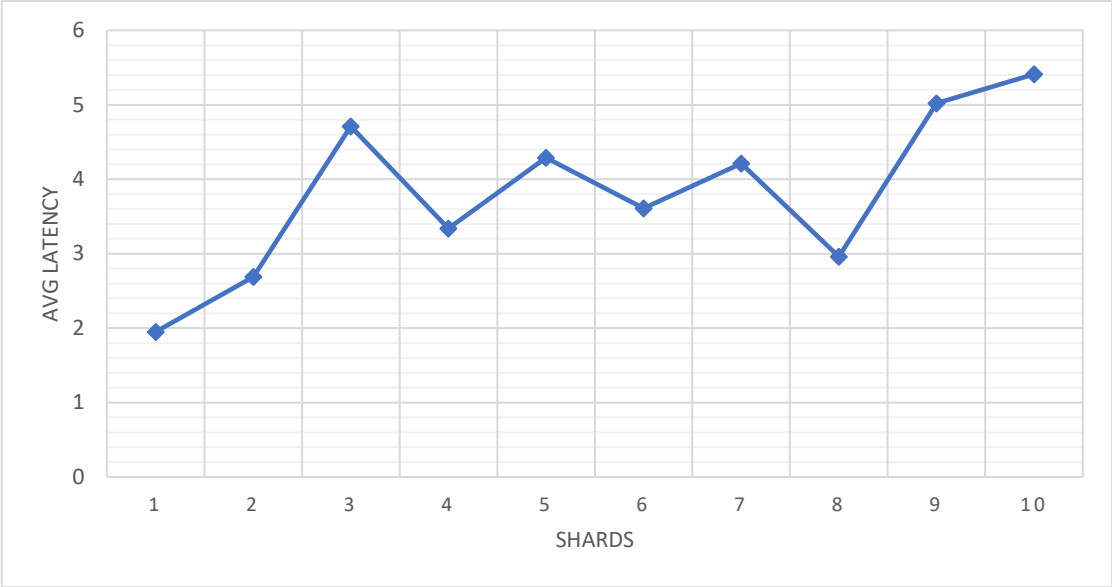

Figure 6.18: Average response latency (in seconds) vs. number of shards

In a second step, the HLF network of FLBCShard is evaluated under several Caliper workers ranging from 1 to 10 workers. The tests use 200 transactions and 30 ones per seconds to measure the system throughput and the average latency. In this step, the HLF network is tested in case of non sharding configuration where there is only one global chain and all transaction are sent to be committed to its ledger.

Figure 6.19 shows the throughput values of every configuration where there some stability of values changes after increasing the number of workers. We observe also that the throughputs of the system are enhanced after increasing the number of shards. The non sharding test has the minimum throughput value and this proves that sharding ensure scalability by treating several transactions in one second.

The figure 6.20 illustrates the average response latency values after testing under 1 to 10 shards and none sharding situation. There is not a big difference between the results of the sharding configurations in contrast to none sharding configuration where the latency is in its maximum value.

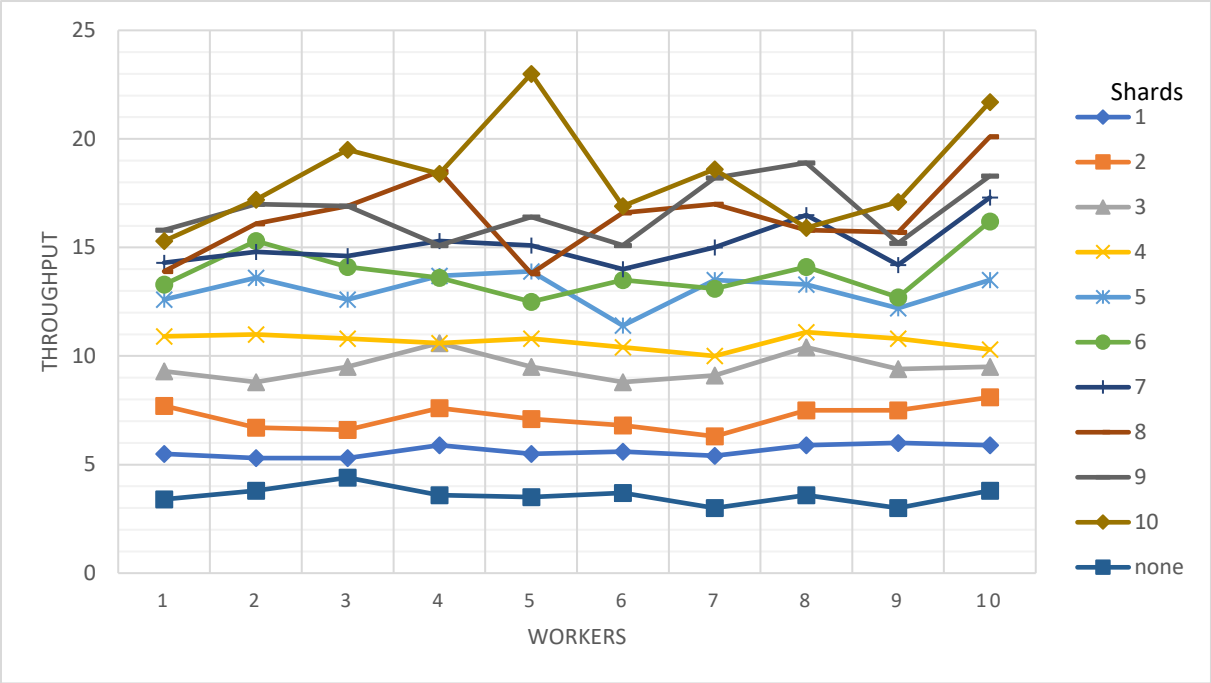

Figure 6.19: Throughput vs. number of workers

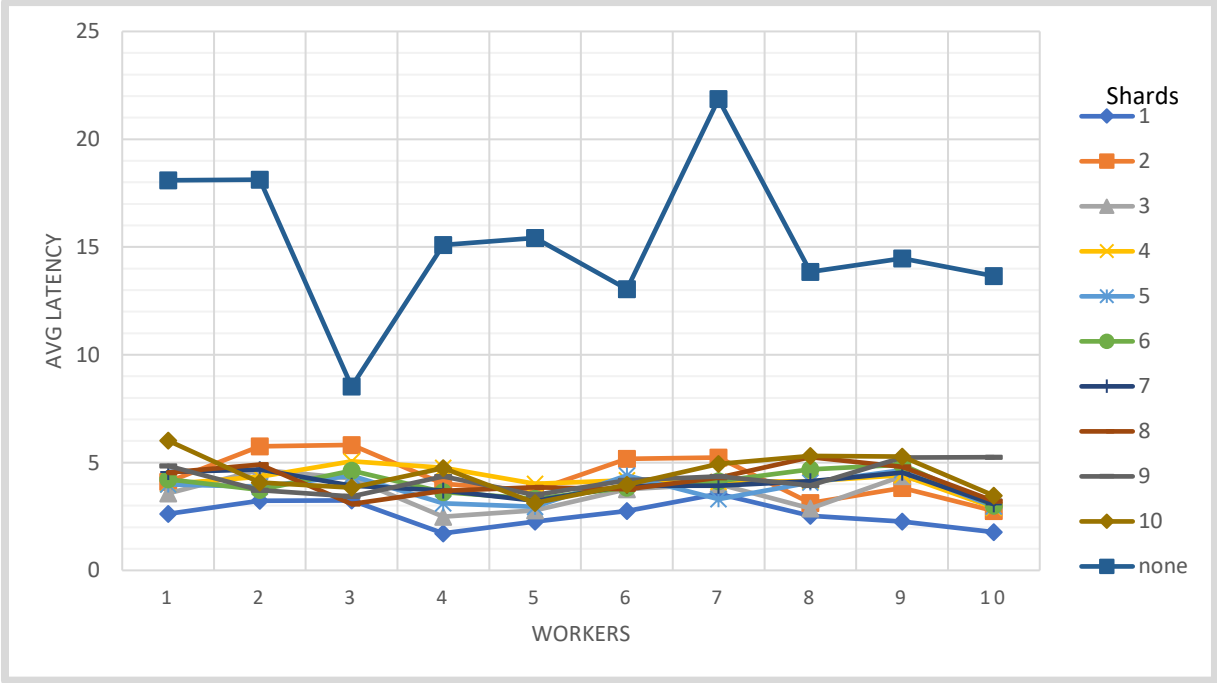

Figure 6.20: Average response latency vs. number of workers

Similair to previous results, figures 6.21 and 6.22 illustrates respectively throughput values and system latency under sharding and non sharding configurations with an increased number of transactions ranging from 100 to 1000 and 4 Caliper workers. The throughput in case of none sharding is in its minimum values and the response latency is in its maximum values. For the sharding, the throughput is increased with an increasing number of transactions and shards

showing that sharding can improve the system throughput in contrast to non sharding configuration.

The same results are observed in latency values where these latter are in their maximum values for the non sharding against sharding configurations that have minimum average latency values.

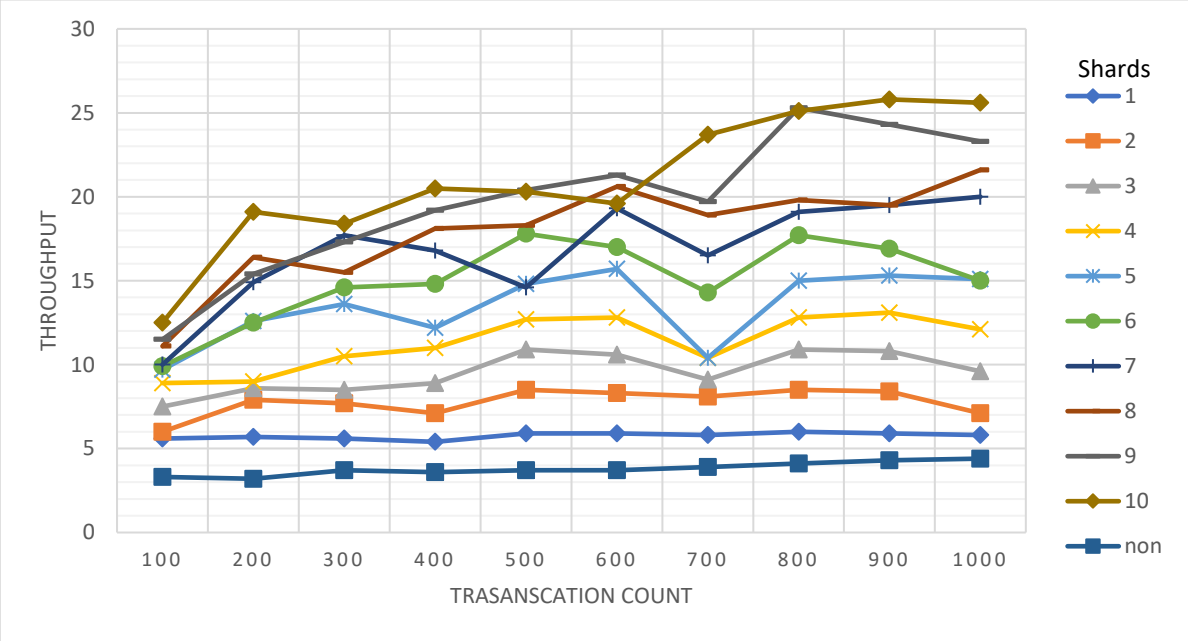

Figure 6.21: Throughput vs. transactions count

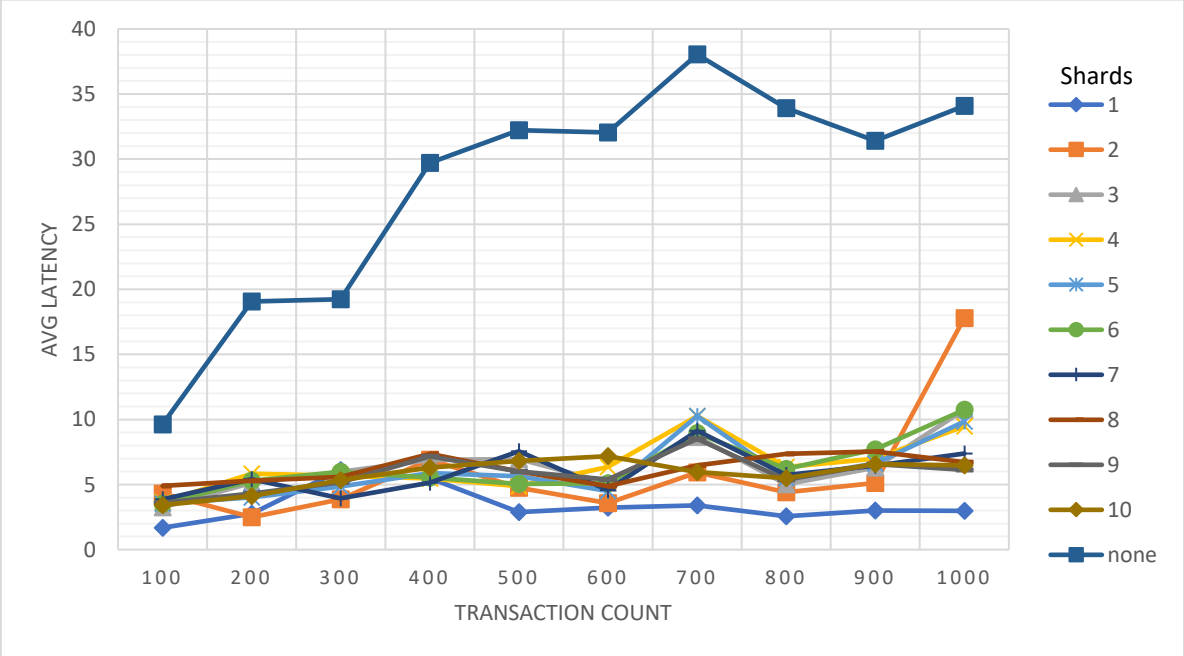

Figure 6.22: Average response latency vs. transaction counts

The Caliper tests are continued with new configuration which is based on the number of transactions per second ranging from 5 to 50 to asses more the performance of the HLF network of FLBCShard. The tests are performed with 4 Caliper workers and 200 transactions. The results are given by figure 6.23 and 6.24 where the first depicts throughput values and the latter are for the average latencies of the sharding and non sharding configurations.

The same observations are shown from the result values which proves that sharding is more scalable that non sharding situation and more shards improve the system throughput and therefore decrease latency values.

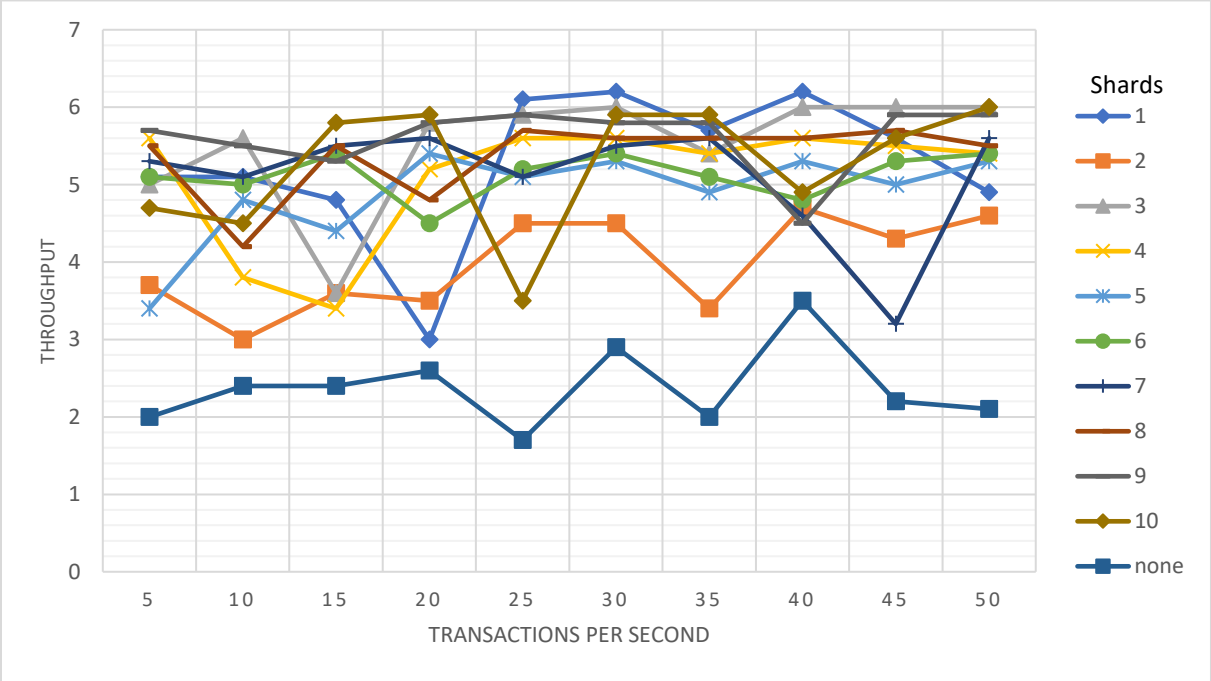

Figure 6.23: Throughput vs. number of transactions per second

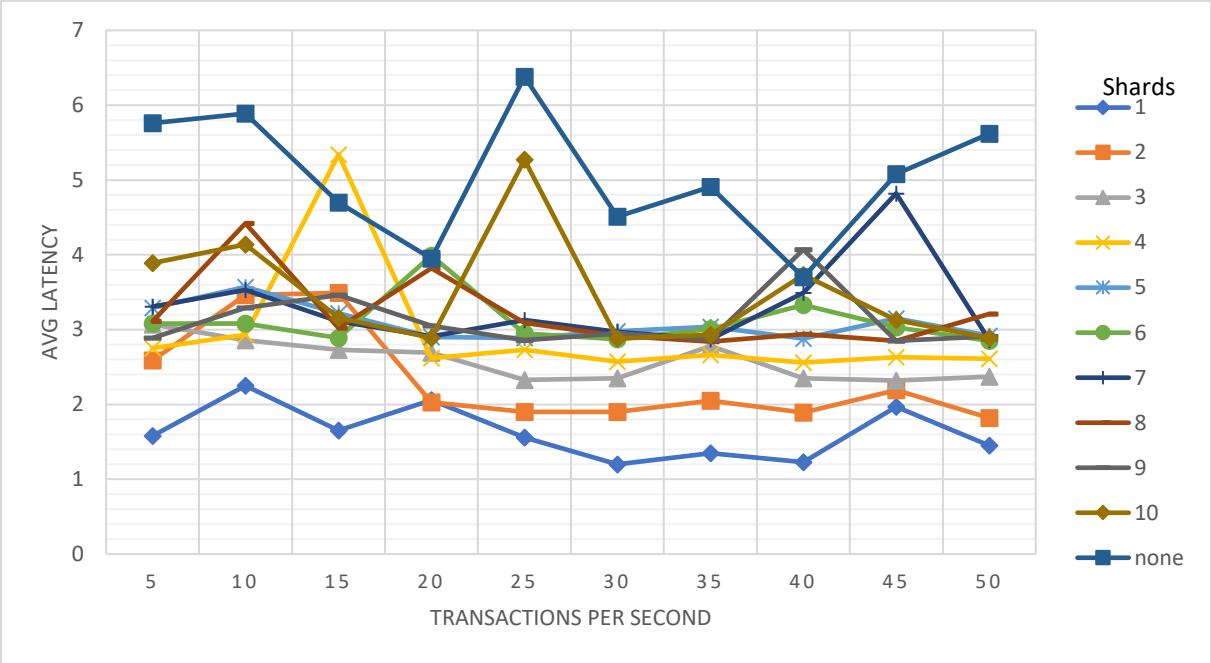

Figure 6.24: Average response latency vs. number of transactions per second

6.6.3.2 Comparaison of Caliper Test Results

In the end of Caliper tests, FLBCShard performance is compared to ScalesFL performance where the throughput and average latency values of the latter are taken from its original paper. Tests are performed under 8 shards and 200 transactions with a varying number of workers. The throughput and latency results are given respectively by figures 6.25 and 6.26.

From figure 6.25, it is observed that throughput values of FLBCShard are higher than the values that are given by ScalesFL in most cases. Therefore, our HLF network can treat more transactions in second compared to ScalesFL. For the average response latencies presented by figure 6.26, ScalesFL gives good values compared to FLBCShard in most tests.

For the none sharding configuration, throughputs and average latency are respectively none encouraging in comparison with FLBCShard and ScalesFL. Therefore, sharding solution gives more throughput and a minimum system latency.

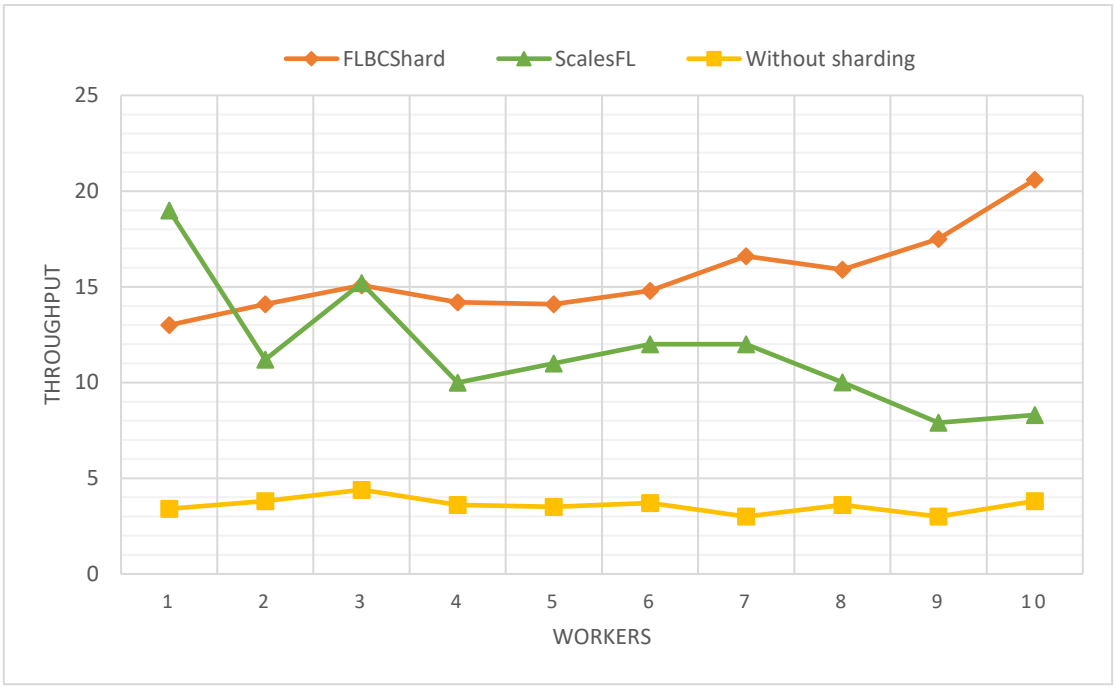

Figure 6.25: FLBCShard, ScalesFL and none sharding throughputs of 8 shards vs. number of workers

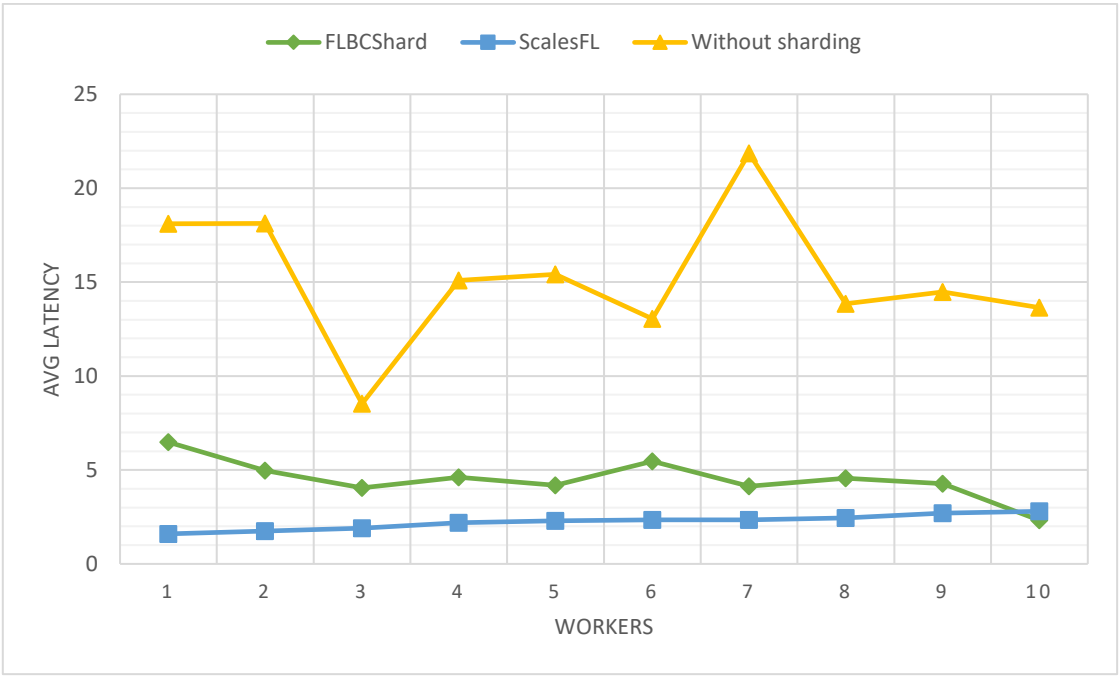

Figure 6.26: FLBCShard, ScalesFL and none sharding average response latency of 8 shards vs. number of workers

6.6.4 Discussion

We evaluated the performance of FLBCShard, and it was shown that blockchain-based FL had no effect on model performance. Therefore, integrating blockchain into FL can enhance the security, trust, integrity and privacy of the trained models, task publishers, and participants. The Raft consensus has given its advantage in reducing peer interactions and making new blocks appending faster.

The evaluated results show that the best setting of FLBCShard uses batch size = 16, noise multiplier =0.1 and an increased number of local rounds. For poising attack evaluation, our work validates proxy results and accept only local update that affect positively the global model accuracy and therefore reduce the effect of poisoned data. For other attacks like inference attack, FLBCShard uses differential privacy to add noise during training and after aggregation, thus reducing the potential of inferring data.

The security and privacy of FLBCShard are compared with those of schemes [7,53,114]. FLBCShard uses dynamic shard formation and supports the use of proxies to improve the privacy of participants, as shown in Table 6.4.

Table 6.5: Comparison between FLBCShard and other sharding-related schemes

Approaches	**Data structure**	**Data Asynchrony**	**Openness**	**Consensus**	**Model evaluation**	**Shard management**	**Data sharing security**	**Model Attak mitigation**	**Off-chain**
[53]	BC+ DAG	Asynchronous	Private	Raft/PBFT	Blockchain peers	Static	Not mention	Malicious stale model	Yes
[114]	BC	Synchronous	Private	Raft	Blockchain peers	Static	Differential privacy	No	Yes
[7]	BC	Synchronous	Public	Proof Of learning	FL workers	Dynamic	Not mention	poisoning attack	No
FLBCShard	BC	Synchronous	Private	Raft	Reliability double check	Dynamic	Differential privacy	poisoning attack	yes

Where the proposed solution [53] does not focus on how securing the model after training all its interest is to alleviate the transaction processing using distributed acyclic graph on one hand, another comparing point that [53] ChainFL shard mechanism is static however this may give raise to security issues. Although in [114] mentioned that the approach used differential privacy however the evaluation does not demonstrate the noise effect on the global accuracy, another shortcoming the ScaleSFL depends on blockchain peers to evaluate the model utility however in practice it's too hard and affects the system overall latency let alone in case 1% attack (taking

control of one shard), for [7] the approach supports using dynamic shard in regards to the huge amount of transactions generated for VANTS ,this scheme utilizes the blockchain (sub and main chain) as a database to save the generated intermediate data (local and global update) where they do not state how they securing the shared data. All the three schemes depend on global chain to aggregate and produce the global model in contrast we figure out it is not trivial in practice to adopt such strategy thus the more securing and trustworthy entity is the task requester owing that the tamper with global accuracy or model utility is meaningless knowing that the more model trains the more proxy rewards.

6.7 Security analysis

In this section we discuss potential threats that could face the proposed solution FLBCshard where suitable countermeasures are fostered to deal with them to demonstrate the design reaction to guarantee a secure, robust and efficient successful execution of FL tasks

Poisoning attack: regarding that each worker trains its local data independently, malicious workers attempt to send poisoned model, we deal with this issue by evaluating the model utility at proxy level based on statistical evaluation. A proxy is considered a semi-trust node, it could tamper with aggregated model accuracy via uploading stale models, in this case the double check method overcomes this issue by launching a collaborative contribution evaluation that calls all proxies of the same shard to evaluate and assess each participant by focusing on the accuracy metric which would prevent proxy to send legitimate old model by checking and comparing the accuracy at each system iteration.

Inference attack: The decentralized FL excites curious proxy to analyze the trained model in order to infer some private useful data about the worker, however each worker adjusts a calibrated noise that added to the optimizer stochastic gradient decedent at each iteration due to differential privacy learn about the private data is more likely impossible.

Sybil attack: regarding that the openness of blockchain network is really tight (permissioned network) all the interacted parties are legitimate (Certificate authority) however curious ones are attempt to create several fake identities and join more than one shard, in fact this is not allowed, where each entity is registered with its corresponding coordinates, in the event of receiving similar values the system alerts that this proxy is suspicious and not reliable no matter its computational resource.

Shard 1% attack: among the challenges of employing blockchain in many applications is 51% attack (one adversary node takes over 51% of the network nodes) this threat is more likely

impossible in practice because it requires huge computational resource, however in sharding blockchain it is potentially taking over an entire shard (1% attack), The success of compromising the shard ultimately depends on the robustness of the consensus mechanism adopted (Raft).

All the aforementioned problems are avoided by incentivizing mechanism, we use Hyperledger fabric platform that not support rewarding mechanism however each task has a price and the task publisher deals with proxies that are already registered into blockchain as legitimate mediators and negotiating the model utility against the reward, the proxy according to the workers capabilities select the appropriate bid, however the punishment level is increasing by increasing proxy faults (reputation).

6.8 Chapter Summary

In this chapter, we propose FLBCShard, a novel privacy-preserving framework based on a sharding blockchain for federation tasks. In our work, NFT-based model sharing is adopted as well, as the model's proof of ownership overcomes several issues, such as the model's rights being reserved, and it is easy to penalize any abuse that affects the model data without the owner's authorization. For better sharpness, FLBCShard is a decentralized FL that supports proxy nodes, which are semi-trusted and qualified clients, to alleviate the communication between participants and the blockchain, regarding the participants' reputation scores, they are selected and assigned to a specific shard. Almost all previous works have focused on how toshard the blockchain in a static manner. Otherwise, we utilize a dynamic shard formation based on clustering the blockchain according to the geo-location coordinates of its peers. We evaluated the effectiveness of FLBCshard using different metrics which demonstrated how the proposed design reacts by changing the experiments and how it withstood against poisoned data.

Conclusion and Future Work

In this chapter, we summarize the thesis conclusion by highlighting the main contributions and their impact on the cyber security field, the limitations of our research study, and propose an attainable future direction.

7.1 Summary of the thesis

With the rapid growth of data consumption across different fields, data sharing has become an economic and scientific fortune for many organizations and sectors, which constantly seek to render it a sustainable source for their own profit. With emerging technologies, IT systems have become smarter, more reactive, and more responsive than ever before. Nevertheless, they have experienced the burden of security and privacy disclosures. Several domestic and international legislations have imposed penalties because data sharing privacy is a fundamental right to practice, and there is a need to guard data from potential violations and misuse to ensure confidentiality and integrity. To overcome security and privacy concerns three approaches to data sharing and privacy preservation based on blockchain technology are proposed in this thesis.

Because individuals' perceptions of the concept of privacy are limited in hiding their real identity, they show little regard for engaging with service provider privacy policies on the possibility of sharing their data with other parties, and they do not undertake control over data destiny. Therefore, user engagement with service providers benefits from particular service claims to share personal or sensitive data. In this regard and in the worst case, the services abuse the privacy policy standards by selling and manipulating them, which may cause ethical and financial losses. Another situation in centralized architecture is when a user launches a query that needs the composition of a certain data service provider to obtain a new effective service for better results. The concern has shifted from protecting users' data privacy to preserving data service provider privacy. In the literature review, particular privacy-preserving mechanisms were adopted to protect data services during the composition process. However, they correlated on a central third party, the so-called mediator, where the main issue, even though using security methods, falls into the single point of failure that affects data availability, on the one hand, let alone internal infringement, for instance, tampering with service quality and data poisoning,

which motivated this thesis to devise a novel decentralized mechanism that ensures better security and privacy of sharing the data. Although federated learning is a new breed of artificial intelligence that takes part in preserving privacy and protecting participant model data sharing from illegal violation, several concerns have been raised relevant to the central aggregator as a single point of failure, a man in the middle attack, and security breaches as internal attacks, such as, lazy participantsand poisoning datasets, in order to tamper with global model accuracy. Blockchain has been incorporated with federated learning to leverage transparency, trustworthiness, and traceability, which would enhance the security and privacy requirements. Therefore, major concerns have arisen, and they have been shown through additional communication and computation overheads, scalability, rebuilding raw data from blockchain transactions, and a trade-off between model accuracy and participant privacy by weakening the overall model performance. The aim of our research was to devise new mechanisms that fill the gap in preserving privacy by leveraging decentralization and security requirements supplied by blockchain technology for raw data and learned model data sharing while providing a user-centric mode to control and determine her data destiny by implementing access control policies. The aforementioned shortcomings are resolved by our three blockchain frameworks, SDGchain, PrsChain, and FLBCshard, which have been discussed, implemented and tested apart in Chapter 4,5,6 and answer the stated issues respectively, and they jointly provide efficient data sharing scalability, secure data storage, and ensure security and privacy requirements that lack several existing solutions. The research questions posed in Section 1.6 were answered by the following contributions:

To protect raw data sharing a privacy-preserving framework based on blockchain that control and track the flow of atomic data services interaction where they run in the same workflow. We designed a prototype of a permissioned Hyperledger Fabric network empowered by a service dependency graph as a mechanism to control service interactions by simulating building the dependencies powered by smart contracts, as well as measuring trust level and proposing a metric that calculates the quality of service in real time to ensure the privacy preservation of the service and access control. The SDGchain framework is based on the fact that the user can manage access control and gain a global overview of his/her data destiny using a service dependency graph. She could infer new knowledge about service behaviors in the event that the service asks for permission to access it another time. In addition, the use of blockchain to protect both the service dependency graph and the pointer of the data contributes to the user having no hesitation in sharing their data in a secure manner while protecting their privacy from disclosure.

We answered the research question two by extending the idea of preserving data privacy inter-atomic services to the case of data service composition. We developed a permissioned blockchain framework for securing the composition process by generating the query plan composition in a decentralized manner (blockchain peers) that would eliminate trust in the central entity and keep the results within the distributed ledger to mitigate tampering with the plan quality. This work demonstrated the capability of blockchain for managing, executing, and preserving the privacy of data service composition. Smart contracts are employed to execute and save data sub-queries and access permissions. Furthermore, this study employed an efficient cryptographic mechanism (ECDSA) to secure data service composition storage and provide efficient access control between the parent service provider and its children through encryption techniques and logging operations that would assist the query issuer in verifying the log ledger of all the data about executed operations. The prototype was implemented based on the Hyperledger Fabric and Interplanetary File System, simulating the execution of the composition using a medical dataset that illustrated the feasibility of the system; consequently, the framework was shown to be efficient and successful in decentralized data service composition.

Then the research question three is answered by devising a new mechanism for trading off between data privacy-preserving and blockchain scalability using the dynamic shard technique. The proposed framework is based on a federated learning paradigm to perform a specific task while using hierarchical architecture based on sharding the main blockchain and it depends on proxy node to alleviate the communication overhead between participant and the distributed ledger ,besides we designed a reliability double check as a double evaluation to mitigate the rate of malicious participant at each level, moreover we used NFT Blockchain for the aim that the task publisher proves its ownership for the sake of eliminating many problems, as the model’s right is protected because any infringement of the model data without the owner’s authorization is penalized. FLBCshard is designed on the basis of dynamic shard which takes part in purifying the system from adversaries. The results of the prototype proves that combining blockchain as a tamper-resistant mechanism and federated learning would increase the level of privacy and the number of shards. Furthermore, this work employs differential privacy that has the potential to ensure privacy of the shared model where the model utility depends on the calibrated noise.

7.2 Thesis limitations

Simulation: The proposed frameworks were implemented based on docker images and tested in a simulated environment that would change the outcomes and performance evaluation when they are employed in a real-world application (true service webs).

Asymmetric key complicacy: Another limitation is the employ of asymmetric key cryptography in regard to the proposed frameworks are able to extend to a large number of network nodes, which renders the process relatively complicated and time-consuming, owing to key pair generation process being computationally complicated and resource consuming to generate random data. Considering that asymmetric key ecosystems are used on the stored data where the amount of the encrypted data is limited and depends on the keys 'size, the larger ones are exponentially costly to generate and use. Furthermore, asymmetric keys are secure and resistant. However, they are vulnerable to quantum computing attacks.

Security Threats: Our thesis attempts to overcome some security threats such as tampering with data, poisoning attack and inference attack; however, we did not address other serious kind of attacks as malicious smart contracts, and adaptive adversarial methods and backdoor attack.

7.3 Future directions

The quality and trust metrics are used to assess each participating service execution in the workflow, which can be enhanced by automating the process in which deep learning is used for faster and more accurate results in a realistic service environment.

To further improve privacy protection in service composition, it is better to adopt an incentivizing mechanism for each service provider that partakes in multiple service compositions. Considering the service node on a large scale, improving the performance evaluation and scalability using IOTA distributed and decentralized ledgers based on tangle technology with a directed acyclic graph ensures rapid transactions and applying differential privacy. With the constant proliferation in service composition to achieve user 'queries, we aspire to refine this prototype to support a realistic experience such as service governments.

Federated machine learning has been employed to protect clients' privacy in the proposed framework cooperated with blockchain, where the effectiveness is clearly defined by security and scalability terms. In future work, we would like to collaboratively train larger data sets to predict chronic diseases in order to improve the model's accuracy using explainability to make sense for the trained data which would assist non data scientist. A novel sharding method will also be proposed that covers multiple criteria, such as using subjective logic to compute reputations.

Bibliography

[1] K.-A. SHIM, A survey of public-key cryptographic primitives in wireless sensor networks, IEEE Communications Surveys & Tutorials, 18 (2015), pp. 577–601. DOI: 10.1109/COMST.2015.2459691

[2] Ram Mohan Rao, P., Murali Krishna, S., & Siva Kumar, A. P. (2018). Privacy preservation techniques in big data analytics: a survey. Journal of Big Data, 5, 1-12.DOI : 10.1186/s40537-018-0141-8

[3] Machanavajjhala, A., Kifer, D., Gehrke, J., & Venkitasubramaniam, M. (2007). l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD), 1(1), 3-es.DOI: 10.1145/1217299.1217302.

[4] Rubner, Y., Tomasi, C., & Guibas, L. J. (2000). The earth mover's distance as a metric for image retrieval. International journal of computer vision, 40, 99-121. DOI :https://doi.org/10.1023/A:1026543900054.

[5] Bonomi, L., Huang, Y., & Ohno-Machado, L. (2020). Privacy challenges and research opportunities for genomic data sharing. Nature genetics, 52(7), 646-654.DOI: 10.1038/s41588-020-0651-0 .

[6] Al-Jaroodi, J.; Mohamed, N. Blockchain in industries: A survey. IEEE Access 2019, 7, 36500–36515. DOI : 10.1109/ACCESS.2019.2903554

[7] H. Chai, S. Leng, Y. Chen, and K. Zhang, “A Hierarchical Blockchain-Enabled Federated Learning Algorithm for Knowledge Sharing in Internet of Vehicles,” IEEE Trans. Intell. Transport. Syst., vol. 22, no. 7, pp. 3975–3986, Jul. 2021, DOI: 10.1109/TITS.

[8] Yaga, D., Mell, P., Roby, N., & Scarfone, K. (2019). Blockchain technology overview. arXiv preprint arXiv:1906.11078. DOI: 10.48550/arXiv.1906.11078 .

[9] 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. DOI: 10.1109/COMST.2020.2969706

[10] S. King and S. Nadal. (Aug. 2012). PPCoin: Peer-to-peer crypto-currency with proof-of-stake. Self-Published Paper. [Online]. Available https://peercoin.net/assets/paper/peercoin-paper.pdf

[11] Castro, M., & Liskov, B. (1999, February). Practical Byzantine fault tolerance. In OsDI (Vol. 99, No. 1999, pp. 173-186).

[12] Liang, Y. C., & Liang, Y. C. (2020). Blockchain for dynamic spectrum management. Dynamic Spectrum Management: From Cognitive Radio to Blockchain and Artificial Intelligence, 121-146. DOI : 10.1007/978-981-15-0776-2_5.

[13] A. Banafa, "Iot and blockchain convergence: benefits and challenges," IEEE Internet of Things", 2017.

[14] V. Buterin, "Ethereum white paper: a next-generation smart contract and decentralized application platform.

[15] Brown, R. G. (2018). The corda platform: An introduction. Retrieved, 27, 2018.

[16] Valenta, Martin, and Philipp Sandner. "Comparison of ethereum, hyperledger fabric and corda." Frankfurt School Blockchain Center 8 (2017): 1-8.

[17] Zhou, Q., Huang, H., Zheng, Z., & Bian, J. (2020). Solutions to scalability of blockchain: A survey. Ieee Access, 8, 16440-16455. **DOI:** 10.1109/ACCESS.2020.2967218

[18] The Zilliqa Technical Whitepaper, Z. Team, Oakbrook Terrace, IL, USA,Sep. 2017, vol. 16, p. 2019.

[19] Lawlor, B., Chalk, S., Frey, J., Hayashi, K., Kochalko, D., Shute, R. & Sopek, M. (2025). Blockchain technology: driving change in the scientific research workflow. *Pure and Applied Chemistry*. DOI:10.1515/pac-2023-1204

[20] Liu, Y., Liu, J., Salles, M. A. V., Zhang, Z., Li, T., Hu, B., ... & Lu, R. (2022). Building blocks of sharding blockchain systems: Concepts, approaches, and open problems. Computer Science Review, 46, 100513. DOI: 10.1016/j.cosrev.2022.100513

[21] Reid, F., Harrigan, M.: An analysis of anonymity in the bitcoin system. In: Security and privacy in social networks, pp. 197–223. Springer (2013). DOI: 10.1109/PASSAT/SocialCom.2011.79.

[22] J.Bernal Bernabe, J. L. Canovas, J. L. Hernandez-Ramos, R. Torres Moreno and A. Skarmeta, 2019. "Privacy-Preserving Solutions for Blockchain: Review and Challenges ». IEEE Access, vol. 7, pp. 164908-164940. DOI: 10.1109/ACCESS.2019.2950872.

[23] Hiwale, M., Walambe, R., Potdar, V., & Kotecha, K. (2023). A systematic review of privacy-preserving methods deployed with blockchain and federated learning for the telemedicine. Healthcare Analytics, 100192. DOI: 10.1016/j.health.2023.100192

[24] Guo, L., Xie, H., & Li, Y. (2020). Data encryption based blockchain and privacy preserving mechanisms towards big data. Journal of Visual Communication and Image Representation, 70, 102741. DOI: 10.1016/j.jvcir.2019.102741

[25] Vasantha, R.; Prasad, R.S.; Guntur, A. Secured email data based on blowfish with blockchain technology. Sci. Technol. Dev. 2019, 8, 456–464.

[26] Ch, R., Srivastava, G., Gadekallu, T. R., Maddikunta, P. K. R., & Bhattacharya, S. (2020). Security and privacy of UAV data using blockchain technology. Journal of Information security and Applications, 55, 102670.DOI: 10.1016/j.jisa.2020.102670.

[27] Sharma, P., Moparthi, N. R., Namasudra, S., Shanmuganathan, V., & Hsu, C. H. (2022). Blockchain-based IoT architecture to secure healthcare system using identity-based encryption. Expert Systems, 39(10), e12915.DOI: 10.1111/exsy.12915

[28] Zuo, Y., Kang, Z., Xu, J., & Chen, Z. (2021). BCAS: A blockchain-based ciphertext-policy attribute-based encryption scheme for cloud data security sharing. International journal of distributed sensor networks, 17(3), 1550147721999616. DOI: 10.1177/15501477219996

[29] Verma, G., & Kanrar, S. (2024). Secure document sharing model based on blockchain technology and attribute-based encryption. Multimedia Tools and Applications, 83(6), 16377-16394. DOI : 10.1007/s11042-023-16186-z .

[30] P. Wang, J. Huang, Z. Cui, L. Xie, and J. Chen, “A gaussian error correction multi-objective positioning model with nsga-ii,” Concurrency and Computation: Practice and Experience, vol. 32, no. 5, p. e5464,2020.DOI: 10.1002/cpe.5464 .

[31] Qashlan, A., Nanda, P., & Mohanty, M. (2024). Differential privacy model for blockchain based smart home architecture. Future Generation Computer Systems, 150, 49-63.DOI: 10.1016/j.future.2023.08.010.

[32] Gai, K., Wu, Y., Zhu, L., Zhang, Z., & Qiu, M. (2019). Differential privacy-based blockchain for industrial internet-of-things. IEEE Transactions on Industrial Informatics, 16(6), 4156-4165. **DOI:** 10.1109/TII.2019.2948094.

[33] Qiu, Y., Liu, Y., Li, X., & Chen, J. (2020). A novel location privacy-preserving approach based on blockchain. Sensors, 20(12), 3519.DOI: 10.3390/s20123519 .

[34] Sowmiya, B., & Poovammal, E. (2022). A heuristic K-anonymity based privacy preserving for student management hyperledger fabric blockchain. Wireless Personal Communications, 127(2), 1359-1376. DOI: 10.1007/s11277-021-08582-1 .

[35] Ahamed, N. N., & Vignesh, R. (2022). Smart agriculture and food industry with blockchain and artificial intelligence. *J. Comput. Sci*, *18*(1), 1-17.DOI: 10.3844/jcssp.2022.1.17

[36] Khadke, S., Gupta, P., Rachakunta, S., Mahata, C., Dawn, S., Sharma, M., ... & Dalapati, G. K. (2021). Efficient plastic recycling and remolding circular economy using the technology of trust–blockchain. *Sustainability*, *13*(16), 9142.DOI: 10.3390/su13169142.

[37] Philip, A. O., & Saravanaguru, R. K. (2020). Secure incident & evidence management framework (SIEMF) for internet of vehicles using deep learning and blockchain. *Open Computer Science*, *10*(1), 408-421. DOI: 10.1515/comp-2019-0022.

[38] Zhao, Y., Zhao, J., Jiang, L., Tan, R., Niyato, D., Li, Z., ... & Liu, Y. (2020). Privacy-preserving blockchain-based federated learning for IoT devices. IEEE Internet of Things Journal, 8(3), 1817-1829.DOI:10.1109/JIOT.2020.3017377.

[39] Singh, S., Rathore, S., Alfarraj, O., Tolba, A., & Yoon, B. (2022). A framework for privacy-preservation of IoT healthcare data using federated Learning and blockchain technology. Future Generation Computer Systems, 129, 380-388.DOI: 10.1016/j.future.2021.11.028.

[40] Wang, N., Yang, W., Wang, X., Wu, L., Guan, Z., Du, X., & Guizani, M. (2022). A blockchain based privacy-preserving federated learning scheme for Internet of Vehicles. Digital Communications and Networks.DOI: 10.1016/j.dcan.2022.05.020.

[41] Truong, N. B., Sun, K., Lee, G. M., and Guo, Y, 2019. Gdpr-compliant personal data management: A blockchain-based solution. IEEE Transactions on Information Forensics and Security, 15, 1746-1761. DOI: 10.1109/TIFS.2019.2948287.

[42] Liang, X., Zhao, J., Shetty, S., and Li, D, 2017.Towards data assurance and resilience in IoT using blockchain. IEEE Military Communications Conference (MILCOM) (pp. 261-266). DOI: 10.1109/MILCOM.2017.8170858.

[43] Chunxia Yu, Luping Zhang, Wenfan Zhao & Sicheng Zhang (2019): A blockchain-based service composition architecture in cloud manufacturing, International Journal of Computer Integrated Manufacturing.DOI: 10.1080/0951192X.2019.1571234.

[44] Radmanesh, S.-A., Haji, A., & Fatahi Valilai, O. (2021). Blockchain-based cloud manufacturing platforms: A novel idea for service composition in XaaS paradigm. PeerJ Computer Science, 7, e743.DOI: 10.7717/peerj-cs.743

[45] Viriyasitavat, W., Da Xu, L., Bi, Z., & Sapsomboon,A. (2018). Blockchain-based business process management (BPM) framework for service composition in industry 4.0. Journal of Intelligent Manufacturing, 31(7), 1737–1748. DOI: 10.1007/s10845-018-1422-y.

[46] M. barhamgi, C. Perera, C. -M. Yu, D. Benslimane,D. Camacho and C. Bonnet, "Privacy in Data Service Composition," in IEEE Transactions on Services Computing, vol. 13, no. 4, pp. 639-652, 1 July-Aug.2020. DOI**:** 10.1109/TSC.2019.2963309

[47] G. P. Tiwary, E. Stroulia and A. Srivastava,"Improving Privacy in Data Service Composition," in IEEE Access, vol. 9, pp. 95716-95729, 2021. **DOI:** 10.1109/ACCESS.2021.3094188.

[48] Tbahriti, S.-E., Ghedira, C., Medjahed, B., & Mrissa,M. (2014). Privacy-Enhanced Web Service Composition. IEEE Transactions on Services Computing, 7(2), 210–222. DOI**:** 10.1109/TSC.2013.18.

[49] Kalapaaking, A. P., Khalil, I., & Yi, X. (2023). Blockchain-based Federated Learning with SMPC Model Verification Against Poisoning Attack for Healthcare Systems. IEEE Transactions on Emerging Topics in Computing. **DOI:** 10.1109/TETC.2023.3268186.

[50] Qi, Y., Hossain, M. S., Nie, J., & Li, X. (2021). Privacy-preserving blockchain-based federated learning for traffic flow prediction. Future Generation Computer Systems, 117, 328-337.DOI: 10.1016/j.future.2020.12.003.

[51] Singh, S., Rathore, S., Alfarraj, O., Tolba, A., & Yoon, B. (2022). A framework for privacy-preservation of IoT healthcare data using federated Learning and blockchain

technology. Future Generation Computer Systems, 129, 380-388. DOI: 10.1016/j.future.2021.11.028

[52] Xu, J., Lin, J., Liang, W., & Li, K. C. (2022). Privacy preserving personalized blockchain reliability prediction via federated learning in IoT environments. Cluster Computing, 25(4), 2515-2526. DOI: 10.1007/s10586-021-03399-w

[53] Yuan, S., Cao, B., Sun, Y., Wan, Z., & Peng, M. (2024). Secure and efficient federated learning through layering and sharding blockchain. IEEE Transactions on Network Science and Engineering. DOI: 10.48550/arXiv.2104.13130

[54] Wang, Q., Ji, T., Guo, Y., Yu, L., Chen, X., and Li, P, 2020. TrafficChain: A Blockchain-Based Secure and Privacy-Preserving Traffic Map. IEEE Access, 8,60598-60612.

[55] EuropeanUnion, "General data protection regulation (GDPR)," Official Journal of the European Union, vol. L119, pp. 1–88, 2016.

[56] Yang Xiao, Ning Zhang, Wenjing Lou, and Y Thomas Hou. A survey of distributed consensus protocols for blockchain networks. IEEE Communications Surveys & Tutorials, 22(2):1432–1465, 2020. **DOI:** 10.1109/COMST.2020.2969706.

[57] Laskey, K. B., & Laskey, K. (2009). Service oriented architecture. Wiley Interdisciplinary Reviews: Computational Statistics, 1(1), 101-105. DOI: 10.1002/wics.8

[58] Doynikova E.and Kotenko I,2016. Countermeasure Selection Based on the Attack and Service Dependency Graphs for Security Incident Management. In:Lambrinoudakis C., Gabillon A. (eds) Risks and Security of Internet and Systems. CRiSIS 2015. Lecture Notes in Computer Science, vol 9572. Springer, Cham. Doi : 10.1007/978-3-319-31811-0-7.

[59] Shameli-Sendi, Alireza,Dagenais, Michel and Wang, Lingyu,2017. Realtime Intrusion Risk Assessment Model based on Attack and Service Dependency Graphs. Computer Communications. 116. DOI: 10.1016/j.comcom.2017.12.003.

[60] Belabed, A., Aimeur, E., Chikh, M. A., and Hadjila, F,2017. A Privacy-Preserving Approach for Composite Web Service Selection. TRANSACTIONS ON DATA PRIVACY, 10(2), 83-115.

[61] Casas-Roma, J., Herrera-Joancomartí, J.and Torra, V. A,2017. survey of graph-modification techniques for privacy-preserving on networks. Artif Intell Rev 47,341–366. 10.1007/s10462-016-9484-8.

[62] Laskey, K. B., & Laskey, K. (2009). Service oriented architecture. WIREs Computational Statistics, 1(1), 101–105. Portico. 10.1002/wics.8

[63] S Sridevi S., Karpagam G. R., Vinoth Kumar B., & Uma Maheswari J. (2021). Investigation on Blockchain Technology for Web Service Composition. International Journal of Web Services Research, 18(1), 70–93. DOI: 10.4018/IJWSR.20210101.oa1.

[64] J. Konecn´y, H. B. McMahan, F. X. Yu, P. Richt´arik, A. T. Suresh, and D. Bacon,"Federated learning: Strategies for improving communication efficiency," CoRR, vol.

abs/1610.05492, 2016.DOI: 10.48550/arXiv.1610.05492

[65] Khemaissia, R., Derdour, M., Ferrag, M. A., & Bouhamed, M. M. (2023). Prschain: A blockchain-based privacy preserving approach for data service composition. Informatica, 47(9). DOI://doi.org/10.31449/inf.v47i9.5081.

[66] Hua, G., Zhu, L., Wu, J., Shen, C., Zhou, L., & Lin, Q. (2020). Blockchain-based federated learning for intelligent control in heavy haul railway. IEEE Access, 8, 176830-176839. DOI :10.1109/ACCESS.2020.3021253.

[67] Zhu, H., Goh, R. S. M., & Ng, W. K. (2020). Privacy-preserving weighted federated learning within the secret sharing framework. IEEE Access, 8, 198275-198284.DOI:10.1109/ACCESS.2020.3034602

[68] Jia, B., Zhang, X., Liu, J., Zhang, Y., Huang, K., & Liang, Y. (2021). Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in IIoT. IEEE Transactions on Industrial Informatics, 18(6), 4049-4058. **DOI:** 10.1109/TII.2021.3085960

[69] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne,"Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," CoRR, vol. abs/1711.10677, 2017.DOI: 10.48550/arXiv.1711.10677.

[70] Reina, G.A., Gruzdev, A., Foley, P., Perepelkina, O.S., Sharma, M., Davidyuk, I., Trushkin, I., Radionov, M., Mokrov, A., Agapov, D., Martin, J., Edwards, B., Sheller, M.J., Pati, S., Moorthy, P.N., Wang, S., Shah, P., & Bakas, S. (2021). OpenFL: the open federated learning library. Physics in Medicine and Biology, 67. **DOI** 10.1088/1361-6560/ac97d9

[71] Ren S, Kim E, Lee C (2024) A scalable blockchain-enabled federated learning architecture for edge computing. PLOS ONE 19(8): e0308991. DOI: 10.1371/journal.pone.0308991

[72] Natarajan, H., Krause, S., & Gradstein, H. (2017). Distributed ledger technology and blockchain.DOI:10.48550/arXiv.1906.11078

[73] Wattenhofer, R. (2016). The science of the blockchain. CreateSpace Independent Publishing Platform.

[74] Casati, F., Ilnicki, S., Jin, L., Krishnamoorthy, V., Shan, MC. (2013). Adaptive and Dynamic Service Composition in eFlow . In: Bubenko, J., Krogstie, J., Pastor, O., Pernici, B., Rolland, C., Sølvberg, A. (eds) Seminal Contributions to Information Systems Engineering. Springer, Berlin, Heidelberg. DOI: 10.1007/978-3-642-36926-1_17

[75] Iyengar, V. S. (2002, July). Transforming data to satisfy privacy constraints. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 279-288). DOI: 10.1145/775047.7750.

[76] Clarke, R. (2006, July). What's' Privacy'. In Australian law reform commission workshop (Vol. 28).

[77] Xiao, X., Yi, K., & Tao, Y. (2010, March). The hardness and approximation algorithms for l-diversity. In Proceedings of the 13th International Conference on Extending Database Technology (pp. 135-146). DOI: 10.1145/1739041.1739060.

[78] Li, N., Li, T., & Venkatasubramanian, S. (2006, April). t-closeness: Privacy beyond k-anonymity and l-diversity. In 2007 IEEE 23rd international conference on data engineering (pp. 106-115). IEEE. DOI: 10.1109/ICDE.2007.367856.

[79] Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3 (pp. 265-284). Springer Berlin Heidelberg. DOI : 10.1007/11681878_14.

[80] Erlingsson, Ú., Pihur, V., & Korolova, A. (2014, November). Rappor: Randomized aggregatable privacy-preserving ordinal response. In Proceedings of the 2014 ACM SIGSAC conference on computer and communications security (pp. 1054-1067). DOI: 10.1145/2660267.2660348.

[81] Zyskind, G., and Nathan, O, 2015. Decentralizing privacy: Using blockchain to protect personal data. IEEE Security and Privacy Workshops (pp. 180-184). DOI :10.1109/SPW.2015.27.

[82] Merkle, R. C. (1980, April). Protocols for public key cryptosystems. In 1980 IEEE symposium on security and privacy (pp. 122-122). IEEE.

[83] Bitcoin white paper.

[84] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang. "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends". In: 2017 IEEE International Congress on Big Data (BigData Congress). 2017, pp. 557–564. DOI: 10.1109/ BigDataCongress.2017.85.

[85] Chen, L., Xu, L., Shah, N., Gao, Z., Lu, Y., & Shi, W. (2017). On security analysis of proof-of-elapsed-time (poet). In Stabilization, Safety, and Security of Distributed Systems: 19th International Symposium, SSS 2017, Boston, MA, USA, November 5–8, 2017, Proceedings 19 (pp. 282-297). Springer International Publishing. DOI : 10.1007/978-3-319-69084-1_19.

[86] Min. Y, Shibin. Z, Yang. Z, Dynamic negotiation of user behavior via blockchain technology in federated system. International Journal of Computational Science and Engineering (2020).

[87] Ongaro, D., & Ousterhout, J. (2014). In search of an understandable consensus algorithm. In 2014 USENIX annual technical conference (USENIX ATC 14) (pp. 305-319).

[88] Dabbagh, M., Kakavand, M., Tahir, M., & Amphawan, A. (2020, September). Performance analysis of blockchain platforms: Empirical evaluation of hyperledger fabric and ethereum. In 2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAIET) (pp. 1-6). IEEE. DOI: 10.1109/IICAIET49801.2020.9257811

[89] Liu, Y., Wang, Y., and Jin, Y. Research on the improvement of mongodb auto-sharding in cloud environment. In 2012 7th International Conference on Computer Science & Education (ICCSE) (2012), IEEE,pp. 851–854. **DOI:** 10.1109/ICCSE.2012.6295203.

[90] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena,‘‘A secure sharding protocol for open blockchains,’’ in Proc. ACM SIGSAC Conf. Comput. Commun. Secur.-CCS, 2016, pp. 17–30. DOI: 10.1145/2976749.297838.

[91] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford,‘‘OmniLedger: A secure, scale-out, decentralized ledger via sharding,’’ in Proc. IEEE Symp. Secur. Privacy (SP), May 2018, pp. 583–598. **DOI:** 10.1109/SP.2018.000-5

[92] Zamani, M., Movahedi, M., & Raykova, M. (2018, October). Rapidchain: Scaling blockchain via full sharding. In Proceedings of the 2018 ACM SIGSAC conference on computer and communications security (pp. 931-948). DOI: 10.1145/3243734.324385.

[93] J. Wang and H. Wang, ‘‘Monoxide: Scale out blockchains with asynchronous consensus zones,’’ in Proc. 16th USENIX Symp. Netw. Syst.Design Implement. (NSDI), 2019, pp. 95–112.

[94] Danezis George and Sarah Meiklejohn. 2016. Centrally Banked Cryptocurrencies.In Network and Distributed System Security Symposium (NDSS). DOI: 10.48550/arXiv.1505.06895

[95] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin,and Beng Chin Ooi. 2019. Towards Scaling Blockchain Systems via Sharding. In SIGMOD Int. Conf. on Management of Data. ACM.DOI: 10.1145/3299869.3319889.

[96]Faisal Nawab and Mohammad Sadoghi. 2019. Blockplane: A global-scale byzantizing middleware. In 2019 IEEE 35th International Conference on Data Engineering (ICDE). IEEE, 124–135. DOI: 10.1109/ICDE.2019.00020.

[97] Amiri, M. J., Agrawal, D., & El Abbadi, A. (2021, June). Sharper: Sharding permissioned blockchains over network clusters. In Proceedings of the 2021 international conference on management of data (pp. 76-88). DOI: 10.1145/3448016.3452807

[98] Douceur, J. R. The sybil attack. In International workshop on peer-to-peer systems (2002), Springer, pp. 251–260. DOI: 10.1007/3-540-45748-8_24

[99] Kumar, N.M.; Mallick, P.K. Blockchain technology for security issues and challenges in IoT. Procedia Comput. Sci. 2018, 132, 1815–1823. DOI: 10.1016/j.procs.2018.05.140.

[100] Ron, D., Shamir, A.: Quantitative analysis of the full bitcoin transaction graph. In: International Conference on Financial Cryptography and Data Security. pp. 6–24. Springer (2013). DOI: 10.1007/978-3-642-39884-1_2.

[101] Koshy, P., Koshy, D., McDaniel, P.: An analysis of anonymity in bitcoin using p2p network traffic. In: International Conference on Financial Cryptography and Data Security. pp. 469–485. Springer (2014). DOI: 10.1007/978-3-662-45472-5_30

[102] A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016, 2016, pp.839–858. DOI: 10.1109/SP.2016.55.

[103] Panwar, G., Misra, S., & Vishwanathan, R. (2019, March). Blanc: Blockchain-based anonymous and decentralized credit networks. In Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (pp. 339-350). DOI: 10.1145/3292006.3300034

[104] Liang, X., Shetty, S. S., Tosh, D., Njilla, L., Kamhoua, C. A., & Kwiat, K. (2019). ProvChain: Blockchain-based Cloud Data Provenance. Blockchain for Distributed Systems Security, 69, 67-94. DOI: 10.1109/CCGRID.2017.8.

[105] Al Omar, A., Rahman, M. S., Basu, A., and Kiyomoto, S,2017. Medibchain: Ablockchain based privacy preserving platform for healthcare data. In International conference on security, privacy and anonymity in computation, communication and storage (pp. 534-543). Springer, Cham. DOI: 10.1007/978-3-319-72395-2_49

[106] A. Shamir, Identity-based cryptosystems and signature schemes, in: CRYPTO,1984.

[107] Truong, H. T. T., Almeida, M., Karame, G., & Soriente, C. (2019, July). Towards secure and decentralized sharing of IoT data. In 2019 IEEE International Conference on Blockchain (Blockchain) (pp. 176-183). IEEE. DOI: 10.1109/Blockchain.2019.00031

[108] Sahai, A. and Waters, B. (2005). Fuzzy identity-based encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 457–473. Springer. DOI: 10.1007/11426639_27.

[109] Qiao, Z., Liang, S., Davis, S., & Jiang, H. (2014, June). Survey of attribute-based encryption. In 15th IEEE/ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD) (pp. 1-6). IEEE. DOI: 10.1109/SNPD.2014.6888687

[110] W, Huang, A., Kandula, A., & Wang, X. (2021, March). A differential-privacy-based blockchain architecture to secure and store electronic health records. In 2021 The 3rd

International conference on blockchain technology (pp. 189-194).DOI: 10.1145/3460537.3460555.

[111] Bao, X., Su, C., Xiong, Y., Huang, W., & Hu, Y. (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. DOI: 10.1109/BIGCOM.2019.00030.

[112] Mikula, T.,and Jacobsen, R. H,2018. Identity and access management with blockchain in electronic healthcare records. IEEE In 2018 21st Euromicro conference on digital system design (DSD) (pp. 699-706). **DOI:** 10.1109/DSD.2018.00008.

[113] Ridhawi, I. A., Aloqaily, M., Boukerche, A., & Jaraweh, Y. (2020). A Blockchain-Based Decentralized Composition Solution for IoT Services. ICC 2020 - 2020 IEEE International Conference on Communications (ICC). DOI: 10.1109/ICC40277.2020.9149031.

[114] Madill, E., Nguyen, B., Leung, C. K., & Rouhani, S. (2022, May). ScaleSFL: a sharding solution for blockchain-based federated learning. In Proceedings of the Fourth ACM International Symposium on Blockchain and Secure Critical Infrastructure (pp. 95-106). DOI: 10.48550/arXiv.2204.01202

[115] Honari, K., Zhou, X., Rouhani, S., Dick, S., Liang, H., Li, Y., & Miller, J. (2022). A scalable blockchain-based smart contract model for decentralized voltage stability using sharding technique. In IEEE International Conference on Blockchain (Blockchain) (pp. 124-131).DOI: 10.48550/arXiv.2206.13776

[116] Ron, D., & Shamir, A. (2013). Quantitative analysis of the full bitcoin transaction graph. In Financial Cryptography and Data Security: 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers 17 (pp. 6-24). Springer Berlin Heidelberg. DOI : 10.1007/978-3-642-39884-1_2.

[117] Gao, A., Yang, D., Tang, S.,and Zhang, M,2006. QoS-driven web service composition with inter service conflicts. In Asia-Pacific Web Conference (pp. 121-132). Springer, Berlin, Heidelberg. DOI: 10.1007/11610113_12.

[118] Srivastava, U., Munagala, K., Widom, J., & Motwani,R. (2006, September). Query optimization over web services. In Proceedings of the 32nd international conference on Very large data bases (pp. 355-366).

[119] A. Hard, K. Partridge, R. Mathews, and S. Augenstein, “Jointly learning from decentralized (federated) and centralized data to mitigate distribution shift,” in Proceedings of NeurIPS Workshop on Distribution Shifts, 2021.DOI: 10.48550/arXiv.2111.12150.

[120] Khemaissia, R., Derdour, M., Djeddai, A., & Ferrag, M. A. (2021). SDGchain: When Service Dependency Graph Meets Blockchain to Enhance Privacy. Proceedings of the 2021 ACM Workshop on Security and Privacy Analytics.DOI: 10.1145/3445970.3451157.

[121] Djeddai, A. (2021, November). KGChain: A Blockchain-Based Approach to Secure the Knowledge Graph Completion. In International Conference on Mining Intelligence and Knowledge Exploration (pp. 218-224). Cham: Springer International Publishing. 10.1007/978-3-031-21517-9_22.

[122] Khemaissia, R., Derdour, M., Ferrag, M. A., & Djeddai, A. (2021). Network Countermeasure Selection Under Blockchain Based Privacy Preserving. 2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI). DOI:10.1109/ICRAMI52622.2021.9585916.

[123] Awan, S., Li, F., Luo, B., & Liu, M. (2019, November). Poster: A reliable and accountable privacy-preserving federated learning framework using the blockchain. In Proceedings of the 2019 ACM SIGSAC conference on computer and communications security (pp. 2561-2563). DOI: 10.1145/3319535.336325

[124] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In Artificial intelligence and statistics, pages 1273–1282. PMLR, 2017.DOI: 10.48550/arXiv.1602.05629

[125] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):22782324, 1998. **DOI:** 10.1109/5.726791

[126] SimonKemp. Digital 2023: Global Overview Report. DataReportal, Global Digital Insights. Available online: https://datareportal.com/reports/digital-2023-global-overview-report (accessed on 08 November 2023).

[127] Ethan Frey and Christopher Goes. [n. d.]. Cosmos Inter-Blockchain Communication (IBC) Protocol.

[128] Wang, Q., Li, R., Wang, Q., & Chen, S. (2021). Non-fungible token (NFT): Overview, evaluation, opportunities and challenges. arXiv preprint arXiv:2105.07447.DOI: 10.48550/arXiv.2105.07447.

[129] Youngblood, C. (2005). An introduction to identity-based cryptography. CSEP 590TU, 1-7.

[130] A. Ekblaw. (2017). Medrec: Blockchain for Medical Data Access, Permission Management and Trend Analysis.

[131] Beata Strack, Jonathan P. DeShazo, Chris Gennings,Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, John N. Clore, "Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records", BioMed Research International, vol. 2014, Article ID 781670, 11 pages, 2014. https://archive.ics.uci.edu/ml/datasets/Diabetes 130-US hospitals for years 1999-2008. DOI: 10.1155/2014/781670

[132] Juan Benet. Ipfs-content addressed, versioned, p2p file system. arXiv preprint arXiv:1407.3561, 2014.