Blockchain in Personal Health Information Exchange

by

Xiaohu Zhou

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

School of Computing and Digital Technology

Faculty of Computing, Engineering and the Built Environment

February 2024

Abstract

The secure and efficient exchange of personal health information is a critical challenge in the healthcare sector. It is a social-technical issue, being concerned with the individual's right to data protection as well as the interoperability of existing health information management systems, such as electronic medical record systems. In particular, there is the need to legally, securely, and efficiently share personal health information between different organisations and entities within and across regions. The various entities in personal health information exchange have different requirements and responsibilities. This thesis focuses on two of these: (1) individuals as data subjects should have the opportunity to oversee the processing of their health information by others and to restrict the exchange of their health information, and (2) entities should be able to verify that data controllers are securely sharing personal health information as agreed and in compliance with regulations, laws and the preferences of data subjects.

To address these challenges, blockchain technology has been actively explored in the research community of health information exchange as a potential solution. This thesis is intended to contribute towards this global effort. Blockchain technology provides benefits on decentralisation, immutability, transparency and traceability of data transactions and public access of data by network users. As a distributed technology, the adoption of blockchain in health information exchange can support interoperability, security, and privacy protection. This thesis aims to explore the use of blockchain technology in personal health information exchange between stakeholders for privacy protection, confidentiality, non-repudiation, and auditability. The four main contributions of the thesis can be summarised as follows:

Firstly, the research identified the requirements of different roles involved in the cases of health information exchange and the current challenges of health information exchange in the sector by reviewing related work on personal health information exchange and

blockchain technology, and discussing existing blockchain-based applications in health information exchange. In summary, there are several challenges related to PHI exchange, including legal and regulatory barriers, privacy and security breaches, lack of interoperability between healthcare information systems, trust-building barriers, and low levels of patient engagement.

Secondly, to explore the use of blockchain technology in data exchange, the study designed a blockchain-based auditing framework for workflows involving different entities. This framework, called AudiWFlow, provides an audit trail for records verification on-the-fly and after the fact using smart contracts and personal receipts. In the context of data exchange in the health sector, the AudiWFlow framework makes data transactions auditable and builds trust between different entities located in the same jurisdiction. Workflow entities share required protected data with each other and use the blockchain to store proof of integrity about transaction records. The blockchain plays the role of an audit server in the framework and has a stable time delay compared to traditional servers.

Thirdly, to address challenges of secure cross-regional data exchange in health, particularly when combined with existing infrastructures in the health management system, this study developed a proper blockchain-based framework called BRUE that can help entities meet fit-for-purpose security requirements in the exchange of personal health information. The BRUE framework reconstructs the concepts of User-Managed Access protocol and uses personal data receipts and token-based records to achieve access control fulfilling the needs of privacy preservation, auditing, non-repudiation, and confidentiality.

Finally, to improve privacy preservation in the exchange of personal health information, the study developed a blockchain-based framework named BRESPE. This framework utilises sticky policy triggered by smart contracts to enforce access control, aligning with user preferences and data protection regulations during data transmission.

Contents

Advance Publications

1. Zhou, X., Nehme, A., Jesus, V., Wang, Y., Josephs, M. and Mahbub, K., 2019. Towards blockchain-based auditing of data exchanges. In Smart Blockchain: Second International Conference, SmartBlock 2019, Birmingham, UK, October 11–13, 2019, Proceedings 2 (pp. 43-52). Springer International Publishing.

2. Zhou, X., Jesus, V., Wang, Y. and Josephs, M., 2020, December. User-controlled, auditable, cross-jurisdiction sharing of healthcare data mediated by a public blockchain. In 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (pp. 87-96). IEEE.

3. Zhou, X., Nehme, A., Jesus, V., Wang, Y., Josephs, M., Mahbub, K. and Abdallah, A., 2022. AudiWFlow: Confidential, collusion-resistant auditing of distributed workflows. Blockchain: Research and Applications, 3(3), Article no. 100073.

List of Figures

List of Tables

Abbreviations

AS	authorisation server
DS	data subject
ETH	ether, currency of Ethereum
LAS	local authorisation server
LRS	local resource server
PDR	personal data receipt
PHI	personal health information
PT	permission token
RAS	remote authorisation server
RO	resource owner
RPT	requesting party token
RqP	requesting party
RRS	remote resource server
RS	resource servers
VIT	verified identity token

Chapter 1

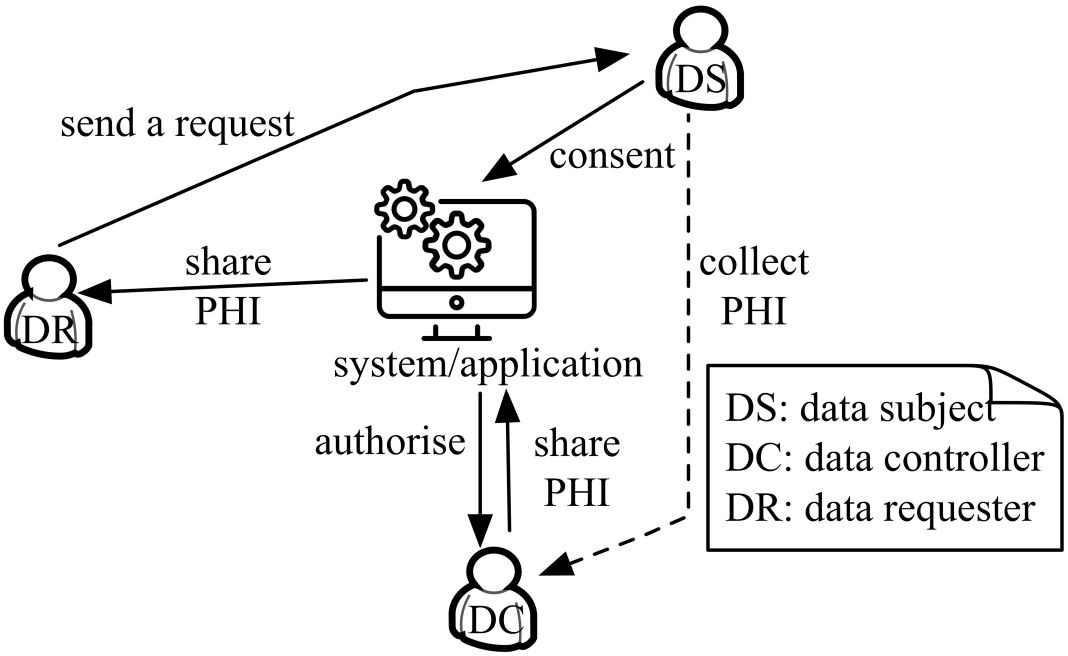
Introduction

The movement of populations worldwide has increased the need for distributed storage and management of personal health information (PHI). However, achieving data interoperability remains a challenge as PHI needs to be shared across different organisations. The COVID-19 pandemic has brought into sharp focus the importance of interoperability and the security challenges that come with it, particularly in sharing a large amount of PHI within and across organisations.

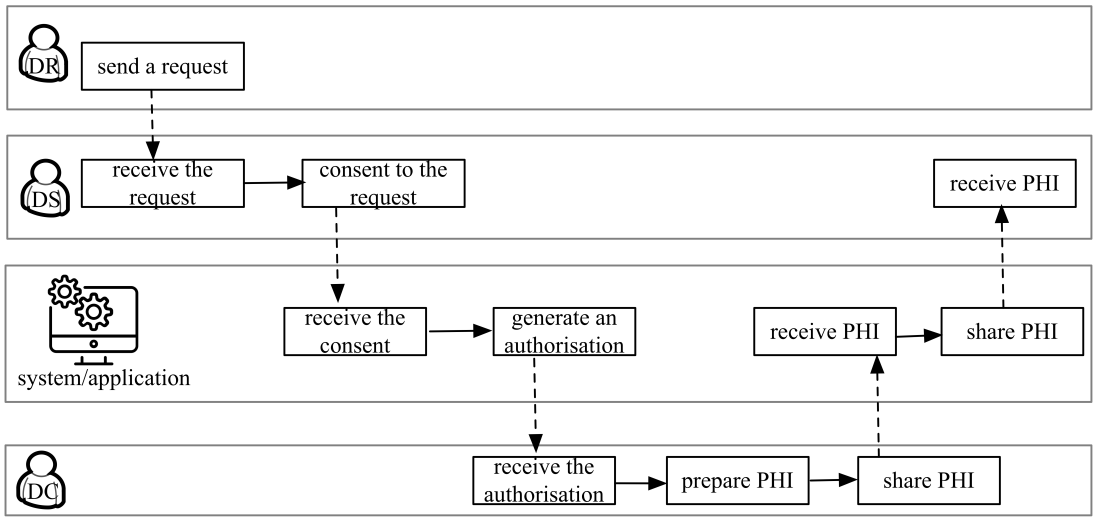
In response to the pandemic, different countries have adopted various methods and systems for exchanging personal data related to people's movements. For example, the British government recommended using the NHS mobile application to stay informed about the latest COVID-19 infection information in their locations and to receive timely notifications about potential exposure to the virus, whereas the Chinese government required people to use a regional barcode shown in WeChat (a widely-used mobile social application in China) to mark their vaccination status and track their physical movements. Both mobile applications collect PHI as proof of one's condition when interacting with other organisations, such as healthcare service providers or border officers. These applications aim to provide valuable information to individuals and health authorities, but raise concerns about the interoperability and privacy of PHI exchange.

To facilitate an understanding of the requirements for the exchange of PHI, we should consider the key data exchange transactions between the involved entities. Figure 1.1 provides a simplified representation of a PHI exchange case, which involves several main entities, including the *data subject*, *data controller*, *data requester*, and an intermediary system (application). We assume that the data controller has already collected PHI from

the data subject and this data flow is indicated by a dashed line in Figure 1.1a. Figure 1.1a presents the main data flow between entities in the scenario and Figure 1.1b provides an overview of the workflow involved in a PHI exchange case.



(a) A data flow representation



(b) A workflow representation

Figure 1.1: A simplified representation of a PHI exchange case

A data subject is an individual whose personal data is being collected and who has the right to authorise access and to know how the data controller processes his or her personal data [1]. For instance, individuals who travelled between countries were required to provide proof of vaccination in 2021. A data controller is an entity or organisation, such as the UK NHS or a data management centre in China, which collects, processes and shares

[1]GDPR Art. 12-23 Rights of the data subject: `https://gdpr.eu/tag/chapter-3/`

personal health data with the permission of the data subject [2]. As the executor of data exchange under the rules, the data controller is responsible for ensuring compliance with regulations and laws during the PHI exchange. A data requester is an entity (organisation) that requires access to the personal data of a data subject. It can be a customs officer or a security guard in a building who needs to view proof of vaccination. The system in Figure 1.1 serves as an intermediary entity that facilitates data transactions between entities, such as a mobile application like the NHS application or WeChat.

In the case presented in Figure 1.1, the data controller collects PHI from the data subject and then shares the data via the intermediary system. The data subject consents to the data controller sharing the data using the system. When the data requester requests to access data of the data subject, it initiates the workflow. However, an entity can play different roles in the workflow. For example, a healthcare organisation may be a data requester when requesting data from the data subject and be a data controller when responsible for data exchange. Compliance with local regulations is also essential when the data subject travels across jurisdictions. The mediation system in Figure 1.1 for PHI exchange can differ in different regions. In summary, to ensure efficient data flow regarding PHI between different entities, it is important that entities share PHI not only in ethical or lawful ways, with authorised parties but also available on an intrinsically as-needed basis.

The General Data Protection Regulation (GDPR) defines personal data as "any information related to an identified or identifiable living individual" (European Commission 2016). The GDPR acknowledges that personal data concerning health is considered sensitive data, which includes data related to "the past, current or future physical or mental health status of the data subject". In the context of the figures presented above, the data exchanged through the intermediary platform relates to personal sensitive data. When entities share PHI between organisations (entities), it is crucial to consider the challenges of data privacy and confidentiality in the data exchange, particularly when dealing with personal sensitive information.

In the health sector, full auditability and traceability of data exchange are crucial aspects of data sharing. A full audit trail and traceability of PHI exchange instil trust in data subjects that the data exchange behaviours of the data controller and data requester

[2]GDPR Art. 24-43 Controller and processor: `https://gdpr.eu/tag/chapter-4/`

comply with data protection regulations and user preferences.

To ensure data privacy and confidentiality in PHI exchange, entities involved in data exchange have several options that can be used. One option is implementing end-to-end encryption protocols to protect PHI from unauthorised access during transmission. Entities can also enforce data access controls to ensure only authorised personnel have access to PHI. Data minimisation practices can also be adopted, which would limit the amount of sensitive PHI exchanged to only that which is strictly necessary for a particular use case.

PHI exchange poses significant requirements for data privacy and confidentiality, which has prompted researchers to explore health technology solutions to address these challenges. Although health technology has attracted increasing interest across healthcare, applications in health information exchange have remained relatively limited to date. As a popular research domain, health technology increasingly handles massive amounts of PHI on a daily basis. With the development of computer networks and the rise of digitisation, healthcare providers have transitioned from paper-based to electronic-based systems, improving the quality of patient care (Schabetsberger et al. 2006, Hillestad et al. 2005) and the interoperability of data exchange. However, the large amounts of PHI stored in these systems and the involvement of sensitive data in data transmission increase the risk of data breaches and cyber attacks.

According to a study conducted by USA Department of Health and Human Services (2018) (HHS), healthcare is the most targeted industry for cyber attacks, accounting for 18.7 percent of all attacks. The HHS report highlighted five primary cybersecurity threats in the health industry that include attacks from email phishing, ransomware, equipment data loss or theft, insider, accidental or intentional data loss, and medical device disconnection (USA Department of Health and Human Services 2018). These cyber-attacks expose patients' sensitive information and pose a threat to patients' safety. The continuous rise of cyber security attacks from external malicious attackers or intentional actions from internal health providers has become a major concern in the health industry. In 2017, the UK NHS declared an investment of up to 20 million pounds to improve cyber security in its organisations (UK Digital 2017). The investment was aimed at enhancing on-site data security assessment, potential threat monitoring, and specialist support of security incidents. This gives a clear indication of the seriousness of the risk to cyber

security in the health industry. The 2019 long-term plan of the NHS emphasises the need for future enhancements in security implementation regarding patient records access and secure data management (NHS 2019).

Health technology solutions such as cloud systems and blockchains are being adopted to improve PHI exchange and enhance cybersecurity. Cloud systems facilitate timely access and sharing of health data between different healthcare organisations, while blockchain technology provides a decentralised and secure digital ledger for sharing health data. Blockchain applications can provide a secure and tamper-proof platform for patients to share their health data with healthcare providers.

Although innovation in health technology drives PHI exchange and enhances cyber security, trustworthiness remains a significant challenge in PHI exchange in terms of security and confidentiality. Health technology provides platforms and applications to support the interoperability of PHI exchange, while patients, as the data subject, need to have confidence that their PHI is being handled securely and ethically in the data transmission. Patients are concerned about the privacy protection of their sensitive information in the exchange of PHI. A lack of trust in the security requirement of PHI exchange can lead to patients withholding information from their healthcare providers, which can ultimately result in adverse health outcomes for patients. Therefore, it is crucial to build trust between patients and healthcare providers for the success of PHI exchange.

The distributed network concept proposed by Baran (1964) provides network solutions for the effective and secure sharing of PHI between different entities and organisations. Blockchain technology, with its intrinsic security, safety, traceability, and irreversibility properties, is a promising solution to address the challenges in PHI exchange. Blockchain consists of a dataset that includes a chain of data blocks and is extended by different new blocks, representing a complete ledger of historical transactions permanently (Wüst & Gervais 2018). As a developing technology in decentralised solutions, blockchain provides both permissionless and permissioned blockchains, catering to different requirements of access control policies.

Ethereum, as one of the popular permissionless blockchains, is available to allow any entity to perform a transaction without relying on any central trusted third authority. It is chosen as the blockchain network in this thesis. Applications of blockchain in personal data sharing are a necessity nowadays, especially for the trust-building in PHI exchange,

nevertheless, there are largely unsolved problems because of the sensitivity of personal data. The problem of user self-controlled operations in personal sensitive information exchange across organisational boundaries (domain) where no trust in any participating entity can be presumed is an open problem that needs to be addressed. It also occurs in the healthcare sector.

Auditing operations of all involved entities in multiple-entities data exchange, and over an arbitrary topology, is also a common requirement of user-controlled operation. The challenges range from compliance with data sharing regulations, trust-building between the data subject and data requester, collaboration work of participating entities, performed actions denied, and compatibility with existing infrastructure.

Regarding the requirements of auditability, traceability, data privacy and confidentiality in PHI exchange, this thesis mainly focuses on exploring the use of blockchain technology in PHI exchange to develop a blockchain-based framework for securely and efficiently sharing PHI between entities and organisations within and across domains. The proposed framework is designed to allow the following: the data subject has the opportunity to grant permission for PHI to be shared between the data controller and data requester; the data requester is allowed to access PHI of the data subject only as agreed, and both the data requester and data controller are prohibited from sharing any PHI of the data subject without explicit permission; the data subject has the right to know the details of its PHI exchange.

1.1 Research Domain and Questions

Privacy-preserving communication of data is a crucial issue in the healthcare environment. The sharing of PHI raises concerns about privacy violations and unauthorised access to data, which can result in a crisis of confidence in the data exchange. The entities involved in the exchange of PHI include independent entities such as patients, healthcare service providers, and third parties, who may be located in different regions. The number of entities participating in the data exchange and the nature of their relationships to facilitate data exchange can sometimes be unclear. It is, therefore, essential to adopt an approach that ensures privacy protection, prevents data disclosure, and monitors all data transactions during interactions between the involved entities.

Given the dynamic nature of PHI exchange requirements, this study aims to address the following three questions:

1. What are the essential requirements for the secure exchange of PHI?
2. How can blockchain technology be effectively employed to enhance data transmission in the healthcare sector?
3. How can a blockchain-based framework be designed and constructed to facilitate the exchange of PHI while ensuring data privacy protection and related authorisation measures?

To address these questions, this research will conduct a literature review of PHI exchange, examining current problems related to health data transmission. It also explores the research landscape of blockchain technology and its applications in health information exchange. The proposed blockchain-based framework will enable effective management of data access authorisation and privacy protection measures.

This study's contribution lies in the development of secure and effective mechanisms for personal health information exchange. This can foster trust and confidence among healthcare sector stakeholders. The research domain of this study encompasses data sharing, healthcare, and blockchain, providing an interdisciplinary approach to address the issues at hand.

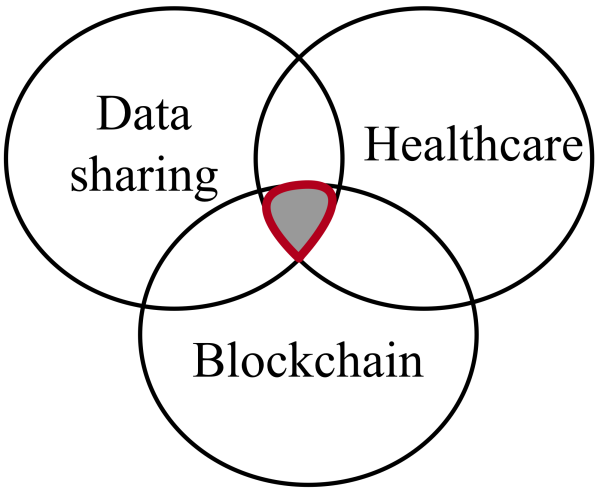


Figure 1.2: Research domain

Figure 1.2 illustrates the research domain of this study, which consists of three interconnected circles representing different research areas. The circle on the left pertains to

the domain of data exchange, which intersects with the healthcare domain in the context of research on healthcare data exchange. Since health information involves sensitive data, it is essential to consider data privacy protection and confidentiality during the data exchange process.

The circle at the bottom represents research on blockchain technology, which intersects with the data exchange domain to examine the use of blockchain technology in facilitating secure data exchange. Blockchain technology has been applied in various industries, and the intersection of the healthcare and blockchain domains examines its applications in the healthcare sector.

The intersection of all three domains is depicted in the central circle marked in red. This study focuses on the application of blockchain technology in health information exchange, which falls within this red area. The terms health data exchange and health information exchange are used interchangeably, and they refer to personal health data or information (PHI). PHI represents personal electronic (digital) data or information that can be used to convey an individual's health status and history online.

1.2 Aim and Objectives

The primary aim of this research is to develop a blockchain-based framework that can facilitate effective and secure PHI exchange between different entities within a single jurisdiction and across jurisdictions while mitigating the risks of unauthorised data access and malicious data processing. It also enhances data transparency and trust in cross-jurisdiction workflows. To achieve this goal, the study delves into the intricacies of PHI exchange and explores the potential of blockchain technology in enhancing data privacy protection and confidentiality measures. This study also explores the practical implementation of the proposed framework and discusses its potential applications in the healthcare sector. The following shows details of the aim of this study.

- Develops a lightweight and distributed framework for secure data exchange that utilises smart contracts on a public blockchain. It aims to achieve auditability and confidentiality for data access control between entities in workflows. In the context of PHI exchange, this framework is used to exchange PHI between different entities located in the same jurisdiction.

- Proposes a novel framework for secure PHI exchange using blockchain technology aiming to achieve effective transmission of PHI while minimising the exchange of such data between entities in cross-region cases.

- Proposes a third framework that exchanges PHI for privacy preservation using smart contracts and other advanced technologies.

The focus of this research is to investigate the current requirements of PHI exchange between different entities within a single region and across regions and to develop a framework that can enable successful data exchange collaboration while ensuring data privacy and security. The research provides a theoretical basis for secure personal health information exchange. To achieve the aim of this research, the research objectives are primarily to identify the requirements, design a framework that satisfies the requirements, and implement the proof-of-concept system of the proposed framework for possibility demonstration. The following objectives have been identified with details:

1. Examines current key requirements of PHI exchange. It will involve a comprehensive review of the literature on health information exchange.

2. Explores the blockchain technology and employment of blockchain in PHI exchange. It reviews literature concerning on auditing, security, and privacy associated with blockchain-enabled implementation.

3. Designs and constructs a blockchain-enabled framework for PHI exchange. According to different concerns about requirements of security, privacy and auditing, there are required three frameworks in total shown as below.

    - Establishes a secure mechanism for data exchange between different entities against malicious data processing. This work provides an initial framework of PHI exchange using blockchain and explores the potential of blockchain technology in this domain. This framework is mainly designed to exchange PHI between entities located in a single jurisdiction ensuring auditability and confidentiality.

    - Develops a specific blockchain-based framework for cross-jurisdictional PHI exchange ensuring confidentiality, traceability, non-repudiation and compati-

bility. It requires implementing a user-friendly interface in the prototype system. Additionally, the effectiveness and efficiency of the proposed framework will be evaluated through experiments on a public testnet and local network. This aims to provide a practical solution for secure cross-jurisdictional personal health information exchange using blockchain technology.

- Enhances data privacy protection in the health environment by combining advanced technologies and simplifying the workflow for sharing PHI to prevent unauthorised data access. As a part of the research objectives, a web application with a user interface will be developed and implemented in the prototype system to demonstrate the feasibility of this third proposed framework.

4. Implements proof-of-concept systems of proposed frameworks and evaluates their effectiveness and feasibility.

5. Concludes and makes recommendations.

1.3 Research Contributions

The proposed framework in this thesis offers several benefits, including self-management of data for the data subject, high-level security, and authorisation management. Through this research, we aim to contribute to the application study of blockchain technology in the health sector by presenting a novel framework for PHI exchange linked to blockchain technology.

The research has achieved its aims and objectives by designing and implementing a prototype framework using a public blockchain for PHI exchange. The proposed framework meets the security requirements for data privacy protection, confidentiality, traceability, non-repudiation, and compatibility with existing infrastructure. The data subject can view and manage authorisation of data access if PHI is exchanged. The data controller can exchange the health data of the data subject based on the protocol without any operational denials. The data requester can effectively access the required health data with permission within and across domains.

The main contributions of the research are as follows:

- Reviewing the literature on PHI exchange and blockchain technology and its ap-

plication in health data exchange. This stage of the research explored the current research situation in the relevant area and defined security requirements.

- Building a new portable blockchain-based auditing framework for data exchange between different entities called AudiWFlow. The AudiWFlow framework demonstrated the use of blockchain technology to meet the requirements of non-repudiation and auditability in the data exchange workflow. The AudiWFlow framework makes data exchange between different entities located in a single jurisdiction auditable when it comes to PHI exchange. In AudiWFlow, the role of the blockchain is an auditing server compared with traditional data exchange frameworks. Experiment results reflect a stable relationship between the cost and size of exchanged records, the average response time of data transactions, and the number of exchanged records.

- Designing and implementing a secure distributed mechanism for PHI exchange across jurisdictions with minimal sensitive personal data and authorisation management. The BRUE framework has been designed for effective and secure PHI exchange in cases of cross-regional exchanges. The framework processes all pre-phases before the actual PHI exchange to be compatible with existing infrastructure, such as a health information management system. It employed smart contracts to build, exchange, and store user permission for data access. A lightweight token-based information exchange was implemented to process authorisation information in the transmission network. All exchanged information in the transmission network about the service request and authorisation management goes through the blockchain network and involves minimal personal sensitive data.

- Exploring the possibility of dynamic privacy-preserving PHI exchange by combining blockchain technology, Personal Data Receipt (PDR), and sticky policy. The BRESPE framework is designed to enhance data privacy protection in PHI exchange, ignoring the content and structure of the required exchanged health data. In the implementation of BRESPE, it used smart contracts to generate, save, and share policy information of data exchange based on user preferences and regulations. The policy record sticks with data transactions in the blockchain network rather than following the exchanged data in the client to prevent information redundancy for clients.

1.4 The Organisation of Thesis

This study reviews the existing literature on the exchange of PHI and blockchain technology, with a focus on proposing secure mechanisms using blockchain technology for PHI exchange. Three frameworks for secure data exchange by smart contracts using a blockchain are proposed, with improvements made to ensure the secure exchange of personal data in the health sector. The proposed frameworks are implemented in three blockchain-enabled prototype systems to demonstrate their feasibility. The following outlines the organisation of the rest of this thesis.

Chapter 2 outlines the methodology employed in the whole thesis including methods about data collection, framework design, prototype implementation and evaluation. This chapter supports the construction of this research.

Chapter 3 provides a literature review of PHI exchange and summarises relevant regulations and rules of PHI exchange to discuss the current security requirements in the health sector. This chapter aims to address research question 1 and archive objective 1.

Chapter 4 introduces blockchain technology, particularly Ethereum and smart contracts, which are used to build secure mechanisms for data transmission meeting security requirements. The chapter provides an overview of current research situation related to blockchain-based PHI exchange about auditing and blockchain-enabled implementation. This chapter addresses research question 2 and targets objective 2.

Chapter 5 presents the proposed framework, which consists of three blockchain-based frameworks in total designed to meet security requirements in different scenarios. The AudiWFlow framework in Section 5.2 is an initial approach designed for PHI exchange to meet the requirements of confidentiality and auditing using Ethereum and smart contracts for different entities positioned in the same jurisdiction. An auditing trail is designed to support data record verification on-the-fly and after the fact. The BRUE framework in Section 5.3 is built for PHI exchange across jurisdictions and uses tokens to exchange minimally sensitive PHI while providing an effective and simple method for authorisation management. The BRESPE framework in Section 5.4 is an extension work that enhances

privacy protection in PHI exchange between the data subject, data controller, and data requester, and employs sticky policies to standardise rules of data exchange and user preferences. This chapter also analyses the design details of the proposed frameworks and addresses the required security requirements. It answers research question 3 and meets objective 3. Lastly, the chapter compared three proposed frameworks with four selected existing frameworks.

Chapter 6 describes the implementation of prototype systems for the three proposed frameworks and evaluates the performance of their prototype implementations in terms of cost and response time. The prototype implementation of the AudiWFlow framework includes an auditing server and workflow, while the implementations of the BRUE and BRESPE frameworks are web applications with client interfaces. Both demo experiments of their prototype implementations are conducted on the default local network and the Goerli network, respectively, to obtain the expected results. This chapter discusses the prototype implementations of the three frameworks in terms of research objective 4 and mainly evaluates the experimental results of the BRUE and BRESPE frameworks. Finally, the work performed the research method of comparative analysis to discuss performance between specific four prototype system and three proposed systems.

Chapter 7 summarises the work of the thesis and introduces future research in framework improvements. It achieves research objective 5.

Chapter 2

Methodology

This chapter briefly introduces the methodology used in the research of this thesis, following by description of a methodological approach. Next, it outlines the details of which research method is selected in every research progress including data collections, framework design, proof-of-concept implementation and evaluation.

2.1 Introduction

The methodology employed in this research is crafted to systematically address the key research questions and objectives, which revolve around understanding the essential requirements for secure personal health information exchange, effectively employing blockchain technology for data transmission in the healthcare sector, and designing a blockchain-based framework that ensures data privacy protection and authorisation measures.

Research questions in this thesis include:

1. Essential requirements for PHI exchange.

    - Objective 1: To investigate the current requirements of PHI exchange between different entities within a single region and across regions.
    - Rationale: This research question aims to comprehensively identify the diverse requirements associated with the exchange of personal health information. By understanding these requirements, the study seeks to inform the development of a framework that caters to the specific needs of secure PHI exchange.

2. Effective employment of blockchain in personal health information exchange.

- Objective 2: To explore how blockchain technology can be effectively employed to enhance data transmission in the healthcare sector.
- Rationale: This question addresses the integration of blockchain as a solution for secure data transmission. The objective is to identify the ways in which blockchain technology can be optimally utilised to address the challenges and inefficiencies in current healthcare data exchange practices.

3. Design and construction of a blockchain-based framework for PHI exchange.

    - Objective 3 and 4: To design a framework that satisfies the identified requirements and implement a proof-of-concept system for demonstration.
    - Rationale: This research question aims to bridge the theoretical understanding of requirements and blockchain technology with practical implementation. These objectives are to propose a viable solution by designing and constructing a blockchain-based framework that not only meets the identified requirements but also serves as a proof of concept for its feasibility.

The research design for this study is carefully crafted to address the multifaceted nature of the research questions, encompassing both qualitative and quantitative aspects. The overarching aim is to holistically investigate the requirements for PHI exchange, explore the utilisation of blockchain technology, and implement blockchain-enabled frameworks. The methodology for this thesis involves a multifaceted approach that encompasses several key components. The following structures the details of research design based on key components of the methodology.

1. Exploratory descriptive study for requirements identification: utilising literature review to identify the essential requirements for secure PHI exchange. An exploratory descriptive study is well-suited for understanding the complexities and variations in PHI exchange requirements.

2. Technology assessment for blockchain technology overview: conducting a comprehensive review of existing blockchain technologies and their implementations to PHI exchange. A comprehensive assessment of existing blockchain technologies, including their strengths and weaknesses, will be conducted. This also involves litera-

ture reviews to understand the practical implications of integrating blockchain into healthcare systems.

3. Design science research for framework design: it is chosen as the overarching approach for developing a novel solution. This involves iterative design cycles, conceptualisation, and architectural planning. The design process will be informed by requirements identified in the exploratory and technology assessment phases.

4. Proof-of-concept implementation and evaluation: implementing proof-of-concept systems based on the designed frameworks to demonstrate the feasibility. The designs will be separately translated into a practical solution through the implementations of proof-of-concept systems. This involves coding the proposed blockchain-based frameworks. Then, evaluating the frameworks through experimental simulation and testing. The evaluation will be carried out in a controlled test environment to demonstrate the feasibility of the proposed solutions.

Through this methodological approach, the research aims to contribute to the theoretical understanding of secure PHI exchange requirements and provide a practical solution in the form of a blockchain-based framework, fostering enhanced data privacy and auditing in PHI exchange.

2.2 Data Collection

The data collection strategy for the research in this thesis is designed to gather comprehensive and diverse information to address the research questions and achieve the stated objectives. Given the multifaceted nature of the study, a combination of qualitative and quantitative data collection methods will be employed:

1. Exploratory descriptive study. Associated with objective 1, there involves data collection methods as below:

    - Document analysis: Existing documents, rules, and standards related to PHI exchange will be analysed to complement the findings. Chapter 3 gives an overview of related regulations and laws about PHI and its exchange between entities. It provides special requirements about regulatory compliance for PHI exchange.

- Literature review: A comprehensive review of academic literature and industry reports on the PHI exchange will be conducted. Chapter 3 explores related work about personal health information exchange including summary of features of personal health information, overview of selected literature about PHI exchange, and discussion on challenges and requirements of PHI exchange.

2. Technology assessment. To meet objective 2, the thesis selects literature review to collected required information for exploring blockchain technology.

   - Literature review: A comprehensive review of academic literature and industry reports on the application of blockchain in auditing and PHI exchange will be conducted. The thesis applies the method of literature review to determine the current state of blockchain in PHI exchange, identify research gaps and limitations, and provide a theoretical and empirical foundation for further research. The indexed keywords include four combined phrases based on 'personal health information exchange', 'blockchain', and 'auditing' that were selected to review literature. Chapter 4 reviews background and current state of blockchain technology, specially for Ethereum and smart contract. It also reviews related work on blockchain for auditing, blockchain-enabled designs for PHI exchange, and blockchain-enabled implementations for PHI exchange in auditing, security and privacy.

3. Design science research. It selects the method of prototyping to support framework design that is employed to meet objective 3.

   - Prototyping: The design process will involve the creation of prototypes and models, which will be refined iteratively based on the requirements. Chapter 5 shows the details that designs three prototypes based on the requirements and findings from the previous method conduction. Besides, two selected software architectures, user-managed access (UMA) and personal data receipt (PDR), are also employed in supporting prototype design. The details of selected software architectures are described in the next section.

4. Proof-of-concept implementation and evaluation. The research of this thesis runs functional test to collect test results for implementations that involves objective 4. It

also conducts experiments for evaluation. Performance metrics will be collected from experiments to assess the efficiency and effectiveness of the proposed blockchain-based frameworks. The following shows the details of related selected methods.

- Functional test: Proof-of-concept systems of three proposed frameworks apply functional tests to valid the system functionality following by their data flow. Chapter 6 gives the details of the implementations.
- Quantitative metrics: Chapter 6 shows and analyses proof-of-concept system performances about response time and cost in a controlled test environment that runs on a Ethereum local network and a simulated test network. The implementations of all proposed framework employ Ganache-CLI to run as blockchain nodes. Besides, both BRUE and BRESPE implementations conduct extra experiments in the Goerli test network to collect more accurate test data. Sections 6.3 and 6.4 shows the details.
- Comparative analysis: a comparative analysis with existing solutions will be conducted to highlight the advantages of the proposed framework. Also, to compare three proposed blockchain-based frameworks with each other to show difference between them. Section 6.6 employs this method to achieve objective 4.

Through these varied data collection methods, the research of this thesis aims to ensure a robust and holistic understanding of the research questions, facilitate the design and implementation of the blockchain-based framework, and provide insights for potential improvements and iterations.

2.3 Selected Software Architectures Used in Framework Design

As an initial blockchain-enabled framework for PHI exchange, AudiWFlow framework is designed to exchange health data between entities in a single organisation or jurisdiction. Its' architecture chooses a blockchain as an audit server without involving a specific software architectures. The other of proposed frameworks in the research refers to two existing

software architectures, UMA and PDR. User-managed Access (UMA) is an OAuth-based protocol. BRUE and BREPSE frameworks reused definitions and concepts from UMA to construct roles and data domain in the data workflow. Personal data receipt (PDR) is an electronic kind of similar supermarket shopping receipt. BRUE and BRESPE frameworks applied PDRs to record interaction information as evidence proof for auditing in the client side. This section introduces UMA and PDR respectively as below.

2.3.1 User-Managed Access

Kantara Initiative introduced UMA, it defines different roles and key concepts associated with the OAuth protocol to support access control for personal data and information resource by individuals (Machulak & Richer 2016). Appendix C shows more details of UMA architecture. UMA delineates roles of *resource owner*, *requesting party*, *resource server*, *authorisation server* and *client*, and concepts of *requesting party token*, *permission*, *permission ticket*, *authorisation process*, *claim*, *token*, *claim token*, *persisted claims token*, *protection API access token*, *authorisation API token* and *saved consent token*. These roles and concepts define the scope of data flow in the workflow shown in Figure C.1. *Resource owner* manages access control for protected information saved in *resource server* and controls *authorisation server* to authorise access. *Authorisation server* authorises *requesting party* to access resources from *resource server* and protects resources in the *resource server* by authorisation with tokens.

BRUE framework uses roles of *resource owner*, *authorisation server*, *resource server* and *requesting party* associated with UMA specification. Besides, BRUE expands the scope of *resource server* and *authorisation server* to meet the requirements of cross-jurisdiction data exchange. In the different jurisdiction compared to *resource owner*, *resource server* and *authorisation server* have local and remote servers. Section 5.3 describes the construction details of BRUE framework.

BRESPE framework selects roles of *data subject*, *data requester* and *data controller* to define the scope of data flow. These roles are constructed on the UMA roles of *resource owner*, *requesting party* and *resource server*.

2.3.2 Personal Data Receipt

A PDR is a suitable method to enhance the completeness of an evidence trail. It is like a conventional paper receipt (e.g., a shopping receipt) that is a digital storable artefact that both the user and service provider store in order to possess evidence satisfying non-repudiation of who, what and how was agreed. The work done at Kantara Initiative (Consent & Information Sharing Work Group 2018) was perhaps the first step towards maturing this concept. The existing ISO 29184 and the upcoming ISO 27560 [1] are expected to bring them into the mainstream. Jesus (2020) outlines the use of fair-exchange protocols to demonstrate a cryptographic receipt of acceptance that can be used to prove the consent and elicit non-repudiation. Jesus argues for a 'web of receipts' based on accountable receipts for consent with a demonstration of the concept in a web browser. Adopting PDR is a good idea because of its inherent simplicity and familiarity as a mechanism for the average user. To target the traceability of data transactions in PHI exchange, a complete audit trail of all data transactions is one of the essential options. Two proposed frameworks, BRUE and BRESPE apply PDRs to record data transaction achieving auditing. PDRs record every data transmission between two entities.

2.4 Proof-of-Concept System Implementation and Evaluation

The implementation of the blockchain component is a pivotal aspect of this research, as it directly addresses the research objectives related to the effective utilisation of blockchain technology for PHI secure exchange. Ethereum is chosen as the blockchain network in three proof-of-concept systems. JAVA and JavaScript are selected as the primary coding languages for the implementations.

2.4.1 Selection of Blockchain Network: Ethereum

Ethereum is selected as the underlying blockchain network due to its widespread adoption, smart contract functionality, and well-established developer community. The Ethereum network is known for its flexibility and suitability for developing decentralised applications

[1] ISO 27560: `https://www.iso.org/standard/80392.html`

(dApps), making it a pertinent choice for a PHI exchange framework. Implementations of AudiWFlow, BRUE and BRESPE frameworks run on the Ethereum.

2.4.2 Coding Languages: JAVA and JavaScript

The implementations of the blockchain-based frameworks will leverage JAVA for back-end development and JavaScript for frontend development. This choice is made for its compatibility with Ethereum's development tools and libraries, facilitating seamless integration with the chosen blockchain network. AudiWFlow uses Java to implement the prototype without an user client. BRUE and BRESPE use JavaScript to implement the user interface and JAVA for data process.

2.4.3 Configuration of Blockchains for Prototype Implementation

For the prototype implementations, a local test environment and Goerli test network are selected to facilitate efficient development, testing, and debugging without the complexities of a live blockchain network. This local environment will run an Ethereum client locally, allowing for rapid iteration and code validation. To ensure the universality and accuracy of experimental results, both the BRUE and BRESPE implementations are also tested on the Goerli network. Table 6.1 shows the experimental configuration environment for the implementations evaluation presented in chapter 6. The following introduces test network and coding framework chosen in the implementations of proof-of-concept systems.

- Ganache-cli: Ganache is employed as the local blockchain network. It provides a controlled Ethereum network for development purposes, allowing for quick and easy testing of smart contracts and interactions with the blockchain.

- Goerli: Goerli is a public testnet of Ethereum with the proof-of-stake consensus mechanism. It provides a simulated Ethereum environment for test with experiments results similar as a real Ethereum network.

- Truffle suite [2]: The Truffle framework is used for the development, testing, and deployment of smart contracts on the Ethereum network. Truffle streamlines the

[2]Truffle: a library to use with web applications. `https://trufflesuite.com/`. Notes: Truffle suite and Ganache were sunset since 2023.

development process and provides a set of tools for managing the entire lifecycle of smart contracts. AudiWFlow uses Truffle suite to develop and deploy blockchain-enabled codes. Besides, BRUE uses a webpack-based truffle box and BREPSE uses React-based truffle box for blockchain-enabled codes in prototypes implementations.

2.4.4 Smart Contracts Development

Smart contracts play a crucial role in three proposed blockchain-based frameworks. These self-executing contracts, written in Solidity (Ethereum's smart contract language), will govern the rules and logic of the PHI exchange on the blockchain.

- Smart contracts design: Smart contracts are designed to handle the secure exchange of audit records and required data, incorporating encryption and authorisation measures in the prototypes implementations.
- Smart contracts test: Rigorous tests of smart contracts are conducted and deployed in the local and Goerli test network to ensure their correctness and feasibility.
- User interface development: A user interface (UI) will be developed using JavaScript for frontend interactions with the blockchain in the BRUE and BRESPE implementations. The UI will provide a user-friendly experience for users involved in the PHI exchange process.

2.5 Summary

The thesis employed a methodological approach to conduct the research including exploratory descriptive study, technology assessment, design science research, proof-of-concept implementation and evaluation. The key components of this methodology selects literature review, document analysis, prototyping, functional test, quantitative metrics and comparative analysis to collect and analyse data, and also demonstrate the possibility of a idea (a blockchain-enabled framework). Chapter 3 identifies requirements of PHI exchange by exploratory descriptive study. Chapter 4 selects technology assessment method to overview blockchain technology and its applications in PHI exchange. Chapter 5 designs the frameworks with design science research method and chapter 6 applies the proof of concept implementation and evaluation. The design of proof of concept systems

refers to software architectures of UMA and PDR. Both BRUE and BRESPE frameworks uses some definitions and concepts of UMA to define some relevant roles and its data flow. With PDR, both proposed frameworks have an appropriate method to record data transaction during the data flow. UMA and PDR enhance the privacy protection of data exchange in the proposed frameworks.

Chapter 3

Personal Health Information Exchange

The exchange of PHI is a crucial aspect of healthcare research. This chapter aims to provide an academic overview of PHI exchange by describing the background of PHI and its evolution based on the development of health technology. It further elucidates the key features of PHI, including its high research value, data-intensive nature, sensitivity, and fragmentation, rich data format, organisation-centred data management, and regulation compliance. The chapter then delves into the relevant regulations and rules governing PHI exchange, including the Health Insurance Portability and Accountability Act (HIPAA) and the EU GDPR. Here describes these regulations how they ensure the security and privacy of personal health information transmission. Furthermore, this chapter discusses the challenges of PHI exchange, which mainly include data exchange incentives, regulation compliance, patient consent, interoperability, and data security. These challenges can impede the efficient exchange of PHI among different healthcare entities and stakeholders. By identifying these challenges, this chapter lays the design foundation for the proposed framework to address these issues and support the secure exchange of PHI. The details see as below.

3.1 Introduction

GDPR provides a comprehensive definition of personal data as any information that pertains to an identified or identifiable living individual (European Commission 2016). This

encompasses a broad range of information, such as passports, driving licenses, electronic health records, and other sources. Regarding personal health information, the GDPR recognises that personal data pertaining to an individual's health is especially sensitive. For example, this includes data related to an individual's physical or mental health condition. This research focuses on the exchange of personal health information, which refers to the electronic transmission of information related to personal health data in the health sector. The sensitivity and importance of this data require stringent regulations and safeguards to protect the privacy and security of individuals. The following outlines the development history of health technology that supports health information exchange to meet security requirements.

The evolution of health technology has a strong relationship with PHI exchange. Health technology supports the development of health information systems that store and process most personal health information. These systems have different types, such as electronic medical records (EMR) and hospital information systems (HIS), which have replaced the traditional paper-based method of managing PHI in healthcare organisations. The paper-based communication of medical documents between care providers was often slow and error-prone (Schabetsberger et al. 2006). The hospital information system was introduced to improve the effectiveness of hospital information management and reduce healthcare costs (Haried et al. 2019). Healthcare organisations use these systems to share information related to healthcare within the organisation.

EMR was developed to manage medical and clinical information related to patients for data querying, storage, and retrieval in the HIS (Esposito et al. 2018). It is a centralised information system that enables healthcare service providers to access patients' medical information directly, while this system is mainly designed for healthcare service providers. Electronic health record (EHR) (Häyrinen et al. 2008) was designed to be more accessible by multiple healthcare providers and patients. It contains a richer data structure than EMR. Both EMR and EHR systems improve the quality of patient care (Hillestad et al. 2005).

With the development of smart wearable devices, more personal information is collected from patients' smart devices. For example, patients use a mobile application to monitor their health state and then collect their daily healthcare data. Personal health record (PHR) was developed to manage data collection between different healthcare

providers and smart devices and store data from multiple sources with a rich structure compared to EMR and EHR. However, the centralisation of data storage in these systems mentioned above has led to problems with real-time data sharing and compatibility with users' different devices. PHI is stored in different healthcare services organisations and third parties databases (collected by smart devices). That leads to issues with system scalability and data exchange between organisations.

To address these issues, a cloud-based system was developed to provide seamless data exchange between EMRs and healthcare organisations (Esposito et al. 2018). However, there are ongoing debates about the problems of health data sharing in a cloud platform (Casola et al. 2016), such as malicious access. Blockchain technology has been proposed as a solution for distributed healthcare data exchange. Some trailblazers have already applied blockchains to healthcare data as a new stage of health document communication. Leveraging the blockchain, 16 percent of healthcare organisations are trailblazers that plan to commercialise blockchain at scale in 2017 and even seem to have a lead in the financial industry (IBM Institution for Business Value 2016).

Requirements of PHI exchange drive innovation in health technology. HISs store all relevant information together for health information management. EMR and EHR systems show specific and professional medical data of patients. PHR system records the health information about an individual's lifetime. Cloud-based and blockchain-based systems solve the problem of decentralised storage and exchange of data and improve system scalability.

This chapter mainly explores the requirements of PHI exchange by reviewing related literature and discusses its challenges in common circumstances as below. It also introduces the development state of PHI and its features, and relevant regulations and rules about data protection.

3.2 Personal Health Information

Patients will normally interact with many healthcare service providers during their lifetime, such as primary care, physicians, and clinicians (Abbas & Khan 2014). Personal health information (PHI) is commonly collected and managed by different healthcare service providers. It concerns the healthcare life cycle of a people, which is scattered stored

in different organisations and regions, and can easily move from one to another care provider due to patients' life events (Azaria et al. 2016). In other words, PHI refers to any information about an individual's health, medical conditions, treatments, and other related information that can be used to identify that person. This type of information can include an individual's name, address, date of birth, medical history, laboratory test results, and other sensitive health information.

Compared with general data, PHI contains information that can identify an individual and is related to their physical or mental health. All relevant information is health-related and continuously updated based on personal health status. PHI has several key characteristics, including research value, data intensity, sensitivity, fragmentation, rich data format and structure, organisation-central management, and regulation. The following lists the details:

- Research value. The research value of PHI is significant for healthcare research. The Office of the National Coordinator for Health Information Technology (ONC 2015) mentioned in its report that biomedical and public health researchers need to analyse information from many sources to identify public health risks, develop new treatments, and enable precision medicine. The research value of PHI also supports healthcare service providers to determine their treatment plans (Wang et al. 2017).

- Intensity. The daily healthcare services involve a large amount of personal data (Esposito et al. 2018). The large population, the more electronic records of patients are created, stored, disseminated, and accessed by healthcare service providers. For instance, an universal healthcare system in China stores a huge amount of personal health data covering all citizens to support data access and management (Zhang, Wang, Li, Zhao & Zhan 2018). Therefore, it is necessary to consider the large size of PHI when applying new techniques in healthcare.

- Sensitivity. PHI is considered highly sensitive information. A report mentioned that individuals are significantly concerned about the privacy of their PHI when data is exchanged between different healthcare service providers (Abdelhamid et al. 2017). There are some regulations and laws (European Commission 2016, National People's Congress Standing Committee of China 2021) mentioning that personal health data involves sensitive data, which is restricted to share between entities

(organisations). The sensitivity of PHI makes it essential to maintain confidentiality against unauthorised operations.

- Fragmentation. PHI is fragmented since it is collected from various healthcare service providers. A PwC report mentioned that although half of the world's population uses the Internet and there are 150,000 health applications for people to use worldwide, very few of them have downloaded more than 5,000 times (PwC Health Research Institution 2018). Health services providers use different healthcare applications to collect and process PHI. Different healthcare applications apply various standards and procedures to collect and store health data, which has resulted in the fragmentation of health information.

- Rich data format and structure. Healthcare service providers collect and store PHI in a rich data format and structure. For example, radiology data is recorded by images and medicine data is recorded by text in the EMR system. Besides, the text records can be presented by structural or natural (nonstructural) language. For instance, a GP's prescription is presented in the structural language and diagnosis information is described in both languages. The diversification of PHI formats and structures increases the difficulty of PHI exchange between healthcare service providers.

- Organisation-centric data management. PHI is organisation-centric that is managed by a specific organisation and stored in a centralised storage. Although patients visit different care providers during their lifetime, most healthcare service providers store the collected data in their centralised systems or platforms. Besides, the data collected from smart devices is also stored in the centralised systems of device suppliers.

- Regulated. PHI is protected and regulated by various standards and regulations, such as HIPAA and Fast Healthcare Interoperability Resources (FHIR). These regulations and standards aim to protect individuals' right to privacy protection and control over their personal health information, as well as to ensure that this information is used in a secure and responsible manner. Therefore, it is essential for healthcare providers to adhere to these regulations and standards when collecting,

storing, and using PHI.

3.3 Relevant Regulations and Laws

In the realm of personal information exchange, various regulations and laws exist across different countries and domains to ensure data privacy preservation and security. The GDPR (European Commission 2016) is an example of such regulation. It was established by the EU to reshape how data can be handled across different sectors in various industries. EU introduced the Directive on Network and Information System (NIS Directive [1]) regulation in 2016 to boost the overall level of security in the cyber and physical resilience of network and information systems within the EU. It is the first EU-wide rule on cybersecurity. In 2023, this Directive was replaced by the EU Directive 2022/2555 (known as NIS2 [2]). In the healthcare domain, the UK NHS developed a Data Security and Protection (DSP) Toolkit to enable organisations to measure and disclose their performance against the National Data Guardian's ten data security standards (UK Digital 2018). The Japanese Act on Protection of Personal Information (APPI) aims to protect individuals' rights and interests and ensure the proper handling of the use of personal data (Japan Personal Information Protection Commission 2017). The cybersecurity law (National People's Congress Standing Committee of China 2016) of China was enacted in 2016 to safeguard information and network space security by providing details of legal measures about data processing in the network. Besides, the Chinese government published a personal information protection law (National People's Congress Standing Committee of China 2021) that defines the conception of personal information and makes rules for handling (sensitive) personal information, including rules for cross-border data exchange. These national laws regulate the collection, storage and processing of personal data during transmission.

There are some well-known standards and rules that standardise data exchange and protection in the healthcare sector. Health Level Seven (HL7) is a standards development organisation that provides a framework for the exchange, integration, and retrieval of electronic health information (Health Level Seven International 2011). HL7 safeguards

[1] EU NIS 2016/1148: `https://eur-lex.europa.eu/eli/dir/2016/1148/oj`

[2] NIS2: `https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32022L2555&from=EN`

sensitive information in the exchange of PHI that needs to be protected about data privacy. It stipulates standards for the secure exchange of PHI between healthcare service providers and organisations. In the data transmission, HL7 requires encrypting and protecting all PHI using authentication mechanisms. Moreover, HL7 has developed Fast Healthcare Interoperability Resources (FHIR), a scalable framework that facilitates interoperability of healthcare data exchange among different organisations leveraging the latest web standards (Health Level Seven International 2011). FHIR is a standard intended to electronically exchange healthcare information between health systems such as EHRs and clinical health systems. It provides specific resources and data elements for exchanging various types of healthcare information, including demographics, clinical observations, medications, and procedures.

Another well-known regulation in America regarding PHI exchange is the Health Insurance Portability and Accountability Act (HIPAA) [3] [4]. HIPAA sets standards for the privacy and security of PHI and limits its use and disclosure (U.S. Department of Health and Human Services 2002). It also requires the confidentiality and protection of individually identifiable health information that is transferred, received, handled, or shared by health professionals and organisations. HIPAA mandates that only minimal health information is necessary to conduct business that can be used or shared.

3.4 Related Work on Personal Health Information Exchange

Personal health information exchange has existed for several decades and has a strong relationship with health technology. PHI exchange and health technology have a symbiotic relationship. The changed requirements of PHI exchange drive the development of health technology and new health technology ensures the secure and efficient exchange of PHI between different entities and organisations. Moreover, health technology provides the infrastructure for PHI exchange to allow users to collect, use, access and share data. In terms of the development history of health technology, there have been numerous research efforts related to the exchange of PHI. The following provides an overview of the related

[3]HIPAA 1996: `https://aspe.hhs.gov/reports/health-insurance-portability-accountability-act-1996`

[4]HIPAA Privacy Rule: `https://www.hhs.gov/hipaa/for-professionals/privacy/index.html`

work on PHI exchange.

3.4.1 Health Technology for PHI Exchange

The advancement of health technology has resulted in a shift from paper-based systems to electronic-based systems, leading to the interoperability of PHI exchange (Schabetsberger et al. 2006). EMR is a typical example of such a system to store and share PHI within authorised departments of healthcare service providers (Steward 2005). Since the computer networks evolution, healthcare organisations use EHR (PHR) to store medical information that can be accessed and shared between different healthcare organisations and patients. While EMR and EHR store information in centralised databases, these remain challenges regarding real-time data sharing between different care providers and system scalability, particularly in meeting users' mobility needs.

Along with the development of the mobile Internet, health technology is required to meet the need for sharing PHI between mobile-based applications and computer-based systems. That gives birth to cloud-based systems in PHI exchange. Different users from different locations and organisations can access and share patients' history records using a cloud-based application that provides a centralised exchange option for PHI stored in decentralised locations (Esposito et al. 2018). The widespread adoption of cloud-based applications led to allowing timely PHI exchange in remote and virtual circumstances. However, centralisation management renders cloud systems vulnerable to external and internal attacks, posing a significant issue to healthcare providers (Casola et al. 2016). To address the security challenges associated with PHI exchange in cloud-based systems, healthcare providers have resorted to the adoption of new technologies such as blockchain. The use of blockchain technology offers a new option in the distributed data exchange of PHI with auditing, transparency, anonymity, and privacy protection.

3.4.2 Interoperability

Interoperability is crucial for the effective communication and exchange of information between different healthcare systems. Health information systems improve the interoperability of data exchange between different healthcare organisations. Casola et al. (2016) recommend the use of cloud technology for healthcare data processing and storage. The

paper presents requirements of healthcare providers that must consider when using cloud technology, including a decentralised and distributed design, synchronous interaction, flexible data and service integration, and a privacy-preserving secure mechanism. Li et al. (2013) propose a system for secure and efficient sharing of PHR in cloud computing environments. The system employs attribute-based encryption to provide fine-grained access control and privacy protection for sensitive health information based on the attributes of users and data objects. It is designed to be scaled, allowing it to handle large amounts of data and users. Fabian, Ermakova & Junghanns (2015) present a novel collaborative architecture and its implementation for inter-organisational PHI exchange, ensuring security and privacy protection in semi-trusted cloud computing environments. The paper also analyses the difficulty of healthcare data exchange scattered across different cloud providers.

3.4.3 Privacy Protection and Security

However, the interoperability of health systems can also bring about problems of privacy and security in the exchange of PHI. A significant number of researches focus on the challenges of privacy and security in the exchange of PHI because of the sensitive nature of health information. Matthews, Harel & Aseltine (2016) highlight the need to balance the public health benefits of data analysis with the need to protect privacy. Kuperman (2011) analyses the challenges associated with the implementation history of health information exchange, including legal and regulatory barriers, privacy and security concerns, and the lack of interoperability among different health information systems. The paper suggests that the success of health information exchange will depend on the adoption of standardised protocols, the development of secure and reliable data-sharing agreements, and the involvement of stakeholders in the health information exchange process.

Vest & Gamm (2010) discuss the challenges of health information exchange, including technical, organisational, financial, and legal barriers. The paper concludes that new strategies are needed to address the ongoing challenges of health information exchange, including the development history of sustainable business models, the establishment of effective governance structures, and the adoption of common data standards and interoperability frameworks. Kaelber & Bates (2007) introduce the potential impact of health

information exchange on patient safety that highlights the benefits of health information exchange, including improved care coordination, reduced medical errors, and decreased healthcare costs. The paper lists several strategies to maximise the benefits of health information exchange while minimising its risks, including establishing clear policies and guidelines for data exchange, ensuring appropriate levels of data access, and using data analytics to identify and address potential safety issues. It also emphasises the importance of involving patients in the design and implementation of health information exchange systems to ensure that patients' privacy concerns are addressed and patients' perspectives are taken into account.

There is also a requirement for security and privacy from data protection regulations and rules for the exchange of PHI. Mello et al. (2018) discuss the legal challenges that hinder the progress of health information exchange in the US by analysing the state and federal laws that regulate health information exchange, including privacy laws, security laws, and laws related to liability and malpractice. They argue that although these laws serve to protect patient confidentiality and security, they also pose significant barriers to the efficient sharing of patient health data. The paper also highlights that the development of a more unified legal framework that balances privacy and security concerns with the need for data sharing is necessary to facilitate the growth of health information exchange.

3.4.4 Trust Build and Entity Incentive

Another important area of research has built trust between patients and healthcare service providers in the exchange of PHI. The willingness of patients to exchange PHI is important. Esmaeilzadeh (2019) investigates the role of perceived transparency of privacy policies and trust in healthcare providers in building trust among patients for health information exchange by a study collecting data from 505 individuals in the US. The results of the study suggest that perceived transparency of privacy policies and trust in healthcare providers are important factors for building trust in health information exchange and trust in healthcare providers has a greater impact on trust in health information exchange compared to perceived transparency of privacy policies. Hence, the paper summarises that improving trust in healthcare providers through better communication and transparency can help build trust in health information exchange and promote

the sharing of health information for improved healthcare outcomes.

Jones et al. (2022) conduct an anonymous online survey in the UK to assess public opinion on sharing data from health services for clinical and research purposes without explicit consent. The authors found that a majority of the respondents were supportive of such data sharing but with certain conditions such as the need for clear information about the purpose of data sharing and data security measures. Participants were less supportive of sharing data for commercial purposes. The study highlights the importance of transparency, trust, and adequate safeguards in promoting public acceptance of data sharing without explicit consent in the healthcare sector. Milne et al. (2021) study the public's views on collecting and sharing genomic data across 22 countries. They found that the majority of participants were willing to share their genomic data for medical research purposes, but were concerned about the security of their data and the potential for discrimination. The paper concludes that trust-building measures and transparent policies are necessary to promote public willingness to share genomic data.

Overall, many studies are concerned about the benefits and limitations of health technology and healthcare systems in the exchange of PHI. Researchers study different health technology and develop different healthcare systems to support PHI exchange. Healthcare systems of PHI exchange include two types, which are centralised and decentralised. To compare both types of healthcare systems in terms of throughput, latency, data integrity, trusted third party, storage, privacy, and system fault tolerance, Table 3.1 (Lee et al. 2022) summarises the difference between the two types of systems in PHI exchange. Decentralised healthcare systems have benefits on latency, data integrity, privacy preservation, system fault tolerance and without mediation, while centralised healthcare systems have high throughput and proper data integrity.

Table 3.1: Comparison: Decentralised healthcare system versus centralised healthcare system (Lee et al. 2022)

Characteristics	Decentralised system	centralised system
throughput	low	high
latency	high	low
data integrity	high	medium
trusted third party	no	yes
storage	distributed ledger	centralised database
privacy protection	strong	weak
system fault tolerance	strong	weak

3.5 Summary

PHI is considered highly sensitive and confidential and subject to a number of legal protections. Relevant laws and regulations govern the privacy and security of PHI and protect its confidentiality by guidelines that must be followed by related healthcare organisations. Additionally, these laws and standards that govern the use and exchange of PHI prioritise privacy and security. To summarise challenges associated with PHI exchange from a review of data protection rules and regulations and related literature, they include legal and regulatory barriers, privacy and security breaches, lack of interoperability among different healthcare information systems, trust-building barriers and low patient engagement. Each of these challenges will be discussed in turn.

Firstly, trust is a critical factor when considering health information exchange. We need concern that building a secure connection without mediators for health information exchange is crucial. In the absence of a data-sharing agreement, exchanging PHI between related entities can be difficult, particularly in emergencies. The presence of a malicious node as a mediator increases the potential risk for secure data sharing.

Secondly, interoperability is also a major challenge. HIPAA defines interoperability as the ability of one computer system to exchange data with another (Lumpkin et al. 2000). The ability to achieve interoperability is classified into three levels: basic, functional, and semantic. Different healthcare organisations achieve varying levels of interoperability that cause difficulty in smoothly exchanging PHI between healthcare organisations. The requirements for health information exchange range from enabling data exchange without interpretation to sharing both the structure and meaning of data without extra interpretation.

Privacy and security breaches are also major concerns, with numerous healthcare incidents such as data breaches and leaks occurring worldwide due to cybersecurity issues. A 2017 report by Verizon (2017) showed that healthcare is the second-most targeted industry for cyber-attacks, with 68 percent of threat actors originating internally. Patients are worried about their data security and privacy during sharing, and potential privacy concerns can influence their intention to agree to health information exchange. They are typically willing to share PHI if the potential issue of privacy is solved (Hersh et al. 2015).

Fourthly, efficiency and scalability are crucial for health information exchange. The

related systems and applications are required to quickly respond to user operations about data access, transmission, and storage. For instance, remote medical systems require the timely sharing of health data. Systems and applications scalability is necessary to be concerned due to the daily exchange of a large amount of health data between different entities.

Fifthly, regulatory compliance is essential, given the numerous regulations by governments, industries, and committees. Rigorous and unified standards are necessary for health information exchange interoperability when different parties are involved (Peter et al. 2007).

Finally, entity incentive is also a critical factor. Healthcare service providers, stakeholders, relevant government departments, patients, and pharmacies are all involved in data sharing, each of them with different requirements for the exchange. Meeting these requirements is a key factor in their willingness to participate in data exchange.

In conclusion, there are several challenges associated with PHI exchange, with legal and regulatory barriers, privacy and security breaches, lack of interoperability among healthcare information systems, trust-building barriers, and low patient engagement being the most significant. Addressing these challenges will require collaboration between healthcare organisations, regulatory bodies, and other stakeholders, with a focus on establishing trust, improving interoperability, ensuring security and privacy, enhancing efficiency and scalability, ensuring regulatory compliance, and offering appropriate incentives to involved entities.

Chapter 4

Blockchain and its Application to Personal Health Information Exchange

Blockchain technology is a relatively new and continually developing technology. It was initially introduced to address the "double-spending problem" in digital currency and later recognised as a solution to the more general "two generals" problem. Since then, blockchain technology has progressed from its first popular application, Bitcoin, to supporting generic scripting in the form of smart contracts, as is the case with Ethereum. Whether used for storing information in a distributed ledger or for algorithmic methods and execution state, such as in Ethereum, blockchains possess the disruptive property of immutability. Once data is stored, it is subject to cryptographic operations that are almost impossible to reverse without abundant computing resources. The longer the time that passes and the more blocks (holding information) that are added, the harder it becomes to modify or destroy a record (i.e., enough computing effort is spent), making it a useful tool for auditing potential (Abreu et al. 2018).

Objective 2 of this work focuses on exploring the blockchain technology and its employment on exchange of PHI. This chapter provides an introduction to the background of blockchain technology, its types, key characteristics, and consensus algorithms. The next section provides an individual introduction to Ethereum and smart contracts considering the design and implementations in chapters 5 and 6. This chapter then reviews the literature on blockchain technology for auditing, security and applications implementation

about information exchange in the healthcare sector. Finally, the chapter summarises the challenges of blockchain-enabled implementation in PHI exchange based on previous discussion, including trust-building barrier, interoperability, security, auditing, and low patient engagement.

4.1 Blockchain

The term "blockchain" was first introduced in 2008 in a paper proposed by the pseudonym Satoshi Nakamoto, regarding the Bitcoin cryptocurrency system (Nakamoto 2008). In 2014, blockchain gained prominence, but the industry attempted to distance the technology from Bitcoin tokens due to its strong association with illegal activities, such as drug trading (Chuen & Deng 2017). Blockchain technology has since been extensively used in various fields, including finance, trade, credit, the Internet of Things, and supply chain management. Many companies, including IBM and Microsoft, have invested heavily in blockchain technology across various industries. In the survey report (The World Economic Forum's Global Agenda Council on the Future of Software & Society 2015) conducted by the World Economic Forum, 58 percent of all respondents expected that 10 percent of the global gross domestic product would use blockchain technology by the year 2025. The survey respondents' high expectations reflect their confidence in the potential for the use of blockchain technology.

Blockchain is a public distributed ledger that contains an ordered list of records linked through blocks generated on a chain (Zhang & Lin 2018, Esposito et al. 2018). Once a block is confirmed and added to the chain, it cannot be deleted or modified, and the chain can only continue to grow. These blocks in the blockchain network permanently record transactions to ensure consistency. The information in a block consists of two parts: the block header and body, as shown in Figure 4.1. The block header records data about the block version, hash of the previous block, timestamp, nonce, difficulty, and Merkle root hash. The block body includes transactions (shown with TX) and the transaction counter. To provide a comprehensive understanding of blockchain technology, the rest of this section outlines the types of blockchains based on entities, delves into the key characteristics of blockchains, and explains three main consensus mechanisms.

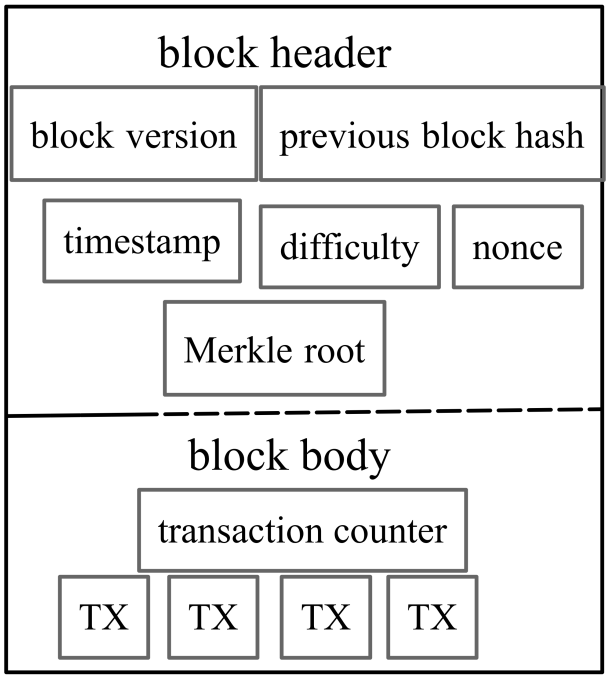


Figure 4.1: Structure of a block in blockchains

4.1.1 Permissionless and Permissioned Blockchains

Blockchain-based applications have become increasingly popular and are being used in various domains such as government processes (Antipova 2018), enterprise business (Ahmad et al. 2018), healthcare data exchange (Castaldo & Cinque 2018, Anderson 2018), and building trust between sectors without intermediaries. These applications operate on different types of blockchains. Blockchains are generally classified into two types based on the entities: permissioned and permissionless. Permissionless blockchains, also known as public blockchains, allow anyone to take part in the network, while permissioned blockchains restrict access to specific individuals or organisations. However, permissionless blockchains may not be suitable for government audit systems due to the difficulty of verifying user identities and enforcing strict data governance (Antipova 2018). In government applications, permissioned blockchains may be more appropriate for ensuring accountability and transparency. Unlike permissionless blockchains, permissioned blockchains restrict access to specific individuals or organisations, making it easier to enforce data governance and verify identities.

Bitcoin and Ethereum are examples of permissionless blockchains. Bitcoin [1], one of the most popular blockchain applications, was first introduced by Nakamoto (2008). It is a cryptocurrency system that combines real-time timestamps with data when generating

[1]Bitcoin: `https://bitcoin.org/en/`

a block, thereby permanently and immutably recording all transactions (IBM Institution for Business Value 2016). Ethereum [2] is an open-source public blockchain platform that features smart contract capabilities and powers thousands of decentralised applications.

A permissionless blockchain is a blockchain that allows all entities to process data transactions, and all transactions in this network are fully transparent to all entities. Although it has lower efficiency (Nasir et al. 2022), this network allows full decentralised unknown entities to interact in transactions consensus validations with security. There is no central authority involved in processing consensus validations in the network, and no one can remove or modify existing transactions within the permissionless blockchain network.

A permissioned blockchain, also known as a private blockchain, allows authorised users to access data in the blockchain network or only permits trusted entities to process transactions. One popular instance of permissioned blockchain platforms is Hyperledger Fabric [3]. This platform establishes the decentralised trust of known entities in the network and provides a modular and pluggable architecture for developing enterprise-grade applications.

Unlike permissionless blockchains, permissioned blockchains restrict access to specific entities in transaction consensus validation. Because entities in consensus validation are limited, this network provides partial decentralisation and transparent data transaction interactions but faster processing. Compared to permissionless blockchains, permissioned blockchains can provide customisable and flexible rules for different use cases, and transaction verification and consensus validation are controlled. In summary, permissioned blockchains offer a more controlled and regulated environment suitable for applications that require enhanced privacy, security, and data governance, while permissionless blockchains are more appropriate for scenarios that require complete decentralisation and openness. Permissionless blockchains can also well support data transactions between untrusted sectors meeting security needs.

[2]Ethereum: `https://ethereum.org/en/`

[3]Hyperledger Fabric: `https://www.ibm.com/topics/hyperledger?mhsrc=ibmsearch_a&mhq=hyperledger`

4.1.2 Key Characteristics of Blockchain

The unique block structure of the chain drives the distinct features of blockchain technology, including transparency, immutability, decentralisation, auditability, and anonymity. Among different types of blockchains, permissionless blockchains provide true transparency and immutability compared with permissioned blockchains (Nasir et al. 2022). Transparency is a crucial aspect of blockchain technology. Data stored in blocks on permissionless blockchains are open to the public for access, whereas in permissioned blockchains, blocks containing data are only open to authenticated users with specific conditions, resulting in limited transparency. Immutability is another key characteristic of blockchains. Once data is stored on a blockchain, it becomes challenging to modify or delete, making it a highly secure and reliable way to store information. Decentralisation is a critical aspect of blockchains that supports data transactions without intermediaries or third parties, which means there is no central authority controlling the transaction network. Instead, the network is maintained by a distributed network of nodes, making it more resistant to attacks and failures.

Consensus algorithms, which are introduced in the next section below, validate data transactions within the blockchain network. The involved data in these transactions are stored, accessed, and managed at multiple locations. Blockchain also provides auditability for data transactions using immutable blocks. All transactions using a blockchain have been recorded by blocks, with a timestamp on each block to record the current transaction time. This feature provides strong support for data auditing in supply chain management. Anonymity is also a noteworthy feature of blockchains that ensures data privacy. Entities can use anonymous identities in blockchains to share data while maintaining privacy preservation.

In summary, blockchain technology has various unique features that make it a highly secure, reliable, and transparent way to store and share data. Its decentralisation, immutability, and transparency make it an excellent fit for use cases where data security, reliability, and transparency are paramount. Additionally, its auditability and anonymity further expand its potential applications in areas such as supply chain management and data exchange.

4.1.3 Consensus Mechanisms

Consensus mechanisms are essential to ensure that nodes on a blockchain network agree on the current state of the blockchain, which guarantees the correctness of information and transaction validation. Blockchains use consensus algorithms to ensure the consistency of data transactions by entities without central authorities or mediators. There are different types of consensus mechanisms used for transaction verification in blockchains. This section primarily introduces three commonly used consensus mechanisms in blockchains: Proof-of-work (PoW), Proof-of-stake (PoS), and Practical Byzantine Fault Tolerance (PBFT). PoW involves nodes competing to solve a cryptographic puzzle to add the next block to the chain. PoS involves nodes staking their cryptocurrency to validate transactions and create new blocks. PBFT is designed to handle faults and can tolerate up to one-third of nodes in the network being Byzantine faulty. Each consensus mechanism has its strengths and weaknesses, and the choice of mechanism will depend on the specific use case of the blockchain. The following presents details of these consensus mechanisms.

Proof-of-work mechanism was first introduced by Back (1997) in his *HashCash* proposal to solve issues of email spam and DoS attacks. Until Bitcoin, PoW was mostly used in systems with decentralised digital currency. Bitcoin uses "proof-of-work" to achieve the consensus of the shared state and order of transactions by incentivizing miners and discouraging rogue actors (Zhang, White, Schmidt, Lenz & Rosenbloom 2018). PoW is a decentralised consensus protocol that requires network entities to invest effort in resolving an arbitrary mathematical conundrum to prevent system gaming. A sufficient number of network nodes need to execute continuous hash computation to achieve a consensus (Ding et al. 2022). This consensus protocol consumes significant amounts of computing power to validate transactions and mine new blocks, causing massive energy consumption while it secures the network. In the Bitcoin network, network nodes are required to have at least 51 percent of all the computational power to reach a consensus. For scalability, PoW allows numerous network nodes to take part but has a high latency (Wang et al. 2019).

Proof-of-stake is another consensus mechanism and an alternative option to PoW in blockchains that depends on the amount of coins (stake) and the duration of coins (stake) held. It prevents wasting computing power in block mining compared to PoW. For security, the network with the PoS protocol requires consensus from nodes holding

at least 51 percent of stakes. PoS improves a node's chances of success in creating new digital tokens proportional to the number of digital tokens the node already owns (Chuen & Deng 2017). It is continuously explored in the applications to secure the network achieving the consensus. King & Nadal (2012) implemented PoS protocol in Peercoin that uses coin-age to calculate the weight of nodes for block mining. Nextcoin (Nxt Community 2014) uses a stake to displace coin-age based on PoS protocol. The proof-of-activity mechanism (Bentov et al. 2014) combines PoS with Nakamoto consensus to only allow online nodes to receive revenues and transaction fees from block mining. Delegated PoS (DPoS) is evolved from a standard PoS consensus mechanism that requires network nodes to vote and elect a smaller group of delegates for the validators in transactions validation and new block generation. DPoS was first introduced by Larimer (2014) and deployed in BitShares [4] in 2015. It is faster to reach a consensus because a particular number of delegates make an agreement in the network. With DPoS, individuals have a chance to secure the network even if they do not have a significant number of stakes.

Byzantine Fault Tolerance (BFT) was designed to work with Byzantine faults but was expensive to implement and only applied in critical real-time systems such as aircraft systems (Chuen & Deng 2017). Practical Byzantine Fault Tolerance was proposed to provide another option for solving Byzantine faults in conventional cases. PBFT protocol was first mentioned by Castro & Liskov (1999). In this protocol, network nodes are authenticated and allowed to send transactions to the validators of the network (Chuen & Deng 2017). These validators are selected as the *primary* or *backup*s to process transactions. A validator selected as a *primary* can broadcast messages to the *backup*s, and the *backup*s return the result messages to the user based on the messages from the *primary*. A network node confirms whether the transaction is completed, which depends on whether it receives the same response from 33 percent of all *backup*s. The role of the *primary* changes if the transaction cannot reach a consensus. With (P)BFT consensus mechanism, the network has a low latency compared to PoW, but it suffers from scalability issues (Wang et al. 2019). There are some applications with (P)BFT protocol, such as Byzcoin (Kokoris Kogias et al. 2016) and Algorand (Gilad et al. 2017), which respectively combine Nakamoto consensus or PoS mechanism with the BFT protocol to achieve a consensus in transactions. Mazieres (2015) uses BFT protocol to take the place

[4]BitShares: `https://bitshares.eu/`

of Nakamoto consensus totally. RIPPB framework improves PBFT protocol by involving reputation calculation for the voting weight and selecting network nodes with the highest reputation as the *primary* (Ding et al. 2022). The calculation of the reputation value of users is dynamically updated based on a logistic regression model.

4.2 Ethereum and Smart Contracts

As a public blockchain platform, Ethereum provides complete decentralisation and security for data exchange. Different distributed applications use smart contracts run on Ethereum to secure and share data in different sectors. Smart contracts are designed to automatically execute predefined protocols supporting business logic. This thesis selected smart contracts run on Ethereum to build and implement the main components of the proposed frameworks. This section gives details of the background of Ethereum and smart contracts.

4.2.1 Ethereum

Ethereum is a decentralised blockchain platform that supports peer-to-peer transactions through smart contracts (Wohrer & Zdun 2018). Ether, also known as ETH, is the dedicated cryptocurrency of Ethereum. Buterin first mentioned the term Ethereum in 2013 and published this white paper in 2014 (Buterin 2014). The first project of Ethereum launched in 2015 and Ether went on sale in 2014. The first implementation of Ethereum, called Frontier, was designed for technical users. Frontier is a bare-bone implementation project of Ethereum in 2015. Homestead is the second implementation of Ethereum that was released in March 2016 and supported the development of smart contracts.

In Ethereum, all network users must register an account to participate in transactions, and each account is defined by a pair of public and private keys. An account address is indexed by the last 20 bytes of its public key, and it includes a nonce, current balance, contract code, and storage. A network user can use this account to transfer value and information with other users, such as signing a transaction with the account's private key (Zheng et al. 2017). There are two types of accounts in Ethereum (Buterin 2014): externally owned accounts, which are controlled by users' private keys and can sign transactions to send messages to others, and contract accounts, which are controlled by their

contract codes and are activated to read and write information to the accounts' storage when they receive messages. An externally owned account has no contract codes.

Ethereum network users can utilise accounts to send messages to other accounts of users or between their own accounts. A complete message contains a STARTGAS value, an optional data field, the amount of ether for message transfer, and the message receiver and sender (Buterin 2014). The message production and transmission can activate the code of a contract to run. A message is packaged inside a signed data package called a transaction, which includes the signature of the message sender, the message receiver, an amount of ether for the transaction, an optional data field, a STARTGAS value, and a GASPRICE value.

A complete transaction process in Ethereum involves initialising a transaction as an instance, signing it with the sender's private key, and submitting it to the local chain node for verification. An instance is a JSON file that includes information about the recipient's address, the amount of ether about this transaction, and the required gas price per limit. The local chain node verifies the signature and adds this instance to the TX pool. Nodes in Ethereum then verify and confirm the transaction before broadcasting it to other nodes.

4.2.2 Smart Contracts

Solidity is Ethereum's Turing-complete programming language that simplifies the implementation of business logic through smart contracts. Smart contracts are computer programs that execute agreements automatically without the need for mediators or third parties. Developers can use Solidity to build codes of smart contracts. Ethereum is a popular example to apply smart contracts in different areas. Szabo began refining the idea of smart contracts since the early 1990s (Szabo 1994) and introduced more details about smart contracts in 1997 (Szabo 1997*b*). There is no a standard definition of smart contracts as well as the terminology of blockchains currently. Szabo defines smart contracts that combine protocols with user interfaces to formalise and secure relationships over computer networks (Szabo 1997*a*). He et al. (2021) describe that a smart contract is a computer protocol designed to disseminate, validate or enforce contracts in an informational manner.

Smart contracts have significant differences from traditional contracts (e.g., tenancy contracts). First, contracts are written in digital form using codes. Once a contract is confirmed, specific inputs of contracts about execution are irreversible. Second, automatic execution is a significant benefit of smart contracts. Codes in contracts are encapsulated with predefined requirements. Smart contracts are automatically activated and then executed without mediators or third parties when predefined requirements are met. There is no need to involve people during this period. That weakens the need to build trust between the entities involved. Besides, smart contracts also provide a predictable outcome from execution. We run codes based on the protocols defined in the contracts to produce the predefined result. Fourth, the execution of smart contracts provides public records that support auditing and traceability. Ethereum is a public blockchain that runs smart contracts to process transactions. The execution information of smart contracts in Ethereum is transparent to the blockchain users. Also, network users can check and view the requirements of smart contracts activation before their executions for transactions. Finally, smart contracts protect privacy. Although the execution of smart contracts is available to network users, all transactions in Ethereum are linked with a cryptographic address during the execution of contracts rather than directly disclosing users' identities.

4.3 Related Work

There are many blockchain-based applications used in diverse areas, such as government settings, resident identification, tax and social framework, and supply chain management (McGhin et al. 2019). The prior literature (Vazirani et al. 2020, Hölbl et al. 2018, Angraal et al. 2017, Zhang et al. 2017, McGhin et al. 2019, Mackey et al. 2019, Yeung 2021, O'Donoghue et al. 2019, Mazlan et al. 2020) has reviewed the current state of blockchain applications in the healthcare sector, outlining the benefits and challenges of such implementations. Some papers propose novel blockchain-based frameworks for processing and sharing personal health information, including the use of a public, private, or consortium blockchain to store the entire health records, only hashes of records or metadata. Some combined blockchain technology with existing infrastructure, such as a database or cloud system, to improve privacy but without compromising security. There are some benefits to using blockchain for data processing, such as auditing, traceability, and distributed

storage. However, potential barriers and challenges remain a concern, i.e., data disclosure (Zhou et al. 2020), regulatory restriction (Angraal et al. 2017, O'Donoghue et al. 2019), and technical issues with data storage (Esposito et al. 2018) for PHI.

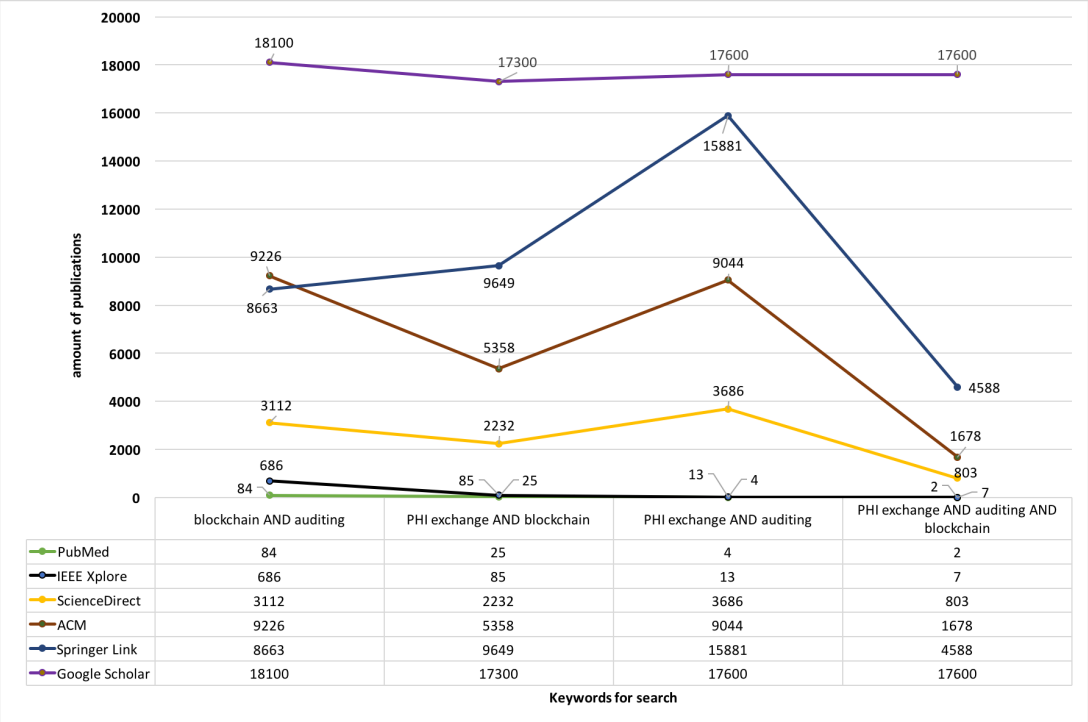


Figure 4.2: Keywords search results in different online databases (2019.1-2023.12)

To better understand the current research state of blockchain technology in auditing and in PHI exchange with auditing before literature discussion, here shows a summary of exploring online relevant publications within five years period in the above figure. Figure 4.2 refers to a keywords search result of electronic publications shown between January 2019 and December 2023 from five common online digital databases and one popular search engine which are PubMed, IEEE Xplore, ScienceDirect, ACM, Springer Link, and Google Scholar. The result in Figure 4.2 is indexed by combined phrases based on keywords 'personal health information exchange', 'blockchain' and 'auditing'. Prior researchers are much more attention on exploring auditing in the PHI field and its development with blockchain technology. That is proved by the search result from Google scholar shown in the purple line that the total of publications from four search keywords are close to a similar point. As a hot index term, exploring auditing in the PHI exchange has significant attracted eye contract. Besides, there are less digital publications about exploring auditing in the PHI exchange with blockchain technology when compared with other index keywords. To meet objective 2 mentioned in chapter 1, the following discusses

the details of relevant literature based on previous keywords phrases and the published period of literature ranges from year 2016 to 2023.

4.3.1 Blockchain for Auditing

The discussion of this section is related to review literature about the use of blockchain for auditing. Prior work has conducted a literature review of blockchain technology applications about auditing (Antipova 2018, Abreu et al. 2018), providing theoretical support for the potential benefits of using blockchain technology (Antipova 2018). By providing an automated mechanism for trust without the need for a central authority (Antipova 2018), blockchain technology has the potential to minimise fraud, optimise existing procedures, and reduce workloads for auditors (Abreu et al. 2018). However, previous research does not present how to integrate blockchain technology with existing auditing processes.

Several previous studies have focused on developing proof-of-concept designs for blockchain-based auditing systems. Ahmad et. al. (Ahmad et al. 2018) propose a system that records distributed and immutable logs in the Hyperledger blockchain to audit transactions in the data exchange workflow, which can withstand external and internal attacks. However, the transparent logs are publicly available and may not be suitable for credential authorities or institutions that require secrecy. Pourmajidi & Miranskyy (2018) propose an approach based on the super-blockchain and circled blockchain to record and receive logs, which can access through APIs on an immutable hierarchical ledger. However, this approach may increase the time required to retrieve logs due to the multiple-hierarchical structure of block storage. Further evaluation requires to determine the impact of this proposal on system performance. Suzuki & Murai (2017) design a prototype system based on the test environment of Bitcoin, which uses blockchain technology to construct audit logs for strictly access-controlled client-server communication channels. However, the issues of high energy consumption and latency associated with the mining process have not been resolved, although this is somewhat compensated through coin returns. Overall, further research is required to address the challenges and limitations of integrating blockchain technology with existing auditing processes.

Many blockchain-based schemes and frameworks are assumed that the audit records generation is trusted. Cucurull & Puiggalí (2016) challenge the storage entity with check-

points published on a Bitcoin blockchain reflecting the integrity of the logs prior to the time each checkpoint is recorded; however, tampering with logs is possible between the checkpoint intervals. Putz et al. Putz et al. (2019) target this limitation by enabling the verification of the integrity of each log entry through hashes published on a permissioned blockchain. They verify that individual log records, collected from different organisations, have not been modified since generation. They also replicate their audit data to ensure its availability, but trust the entity storing their logs with the confidentiality and privacy of their audit records. Tian (2017) uses blockchain with distributed databases to track a food supply chain process. Each participant in the supply chain generates and maintains audit records of its part of the process, and submits a proof of authenticity of the records they have to the blockchain. The discussed mentioned above approaches trust the audit records and use blockchain to verify these evidences if required. However, Tapas et al. (Tapas et al. 2019) do not assume trust during their generation of logs, and rely on mutual challenges between two parties to verify the authenticity of evidence reported to cover interactions between them. They follow a blockchain-based approach that supports the verification of basic operations between a client and a cloud service provider storing data, and do not consider workflows including multiple administrative domains. A lightweight and confidentiality friendly approach has not been covered in the literature to verify the integrity of audit trails in a workflow combining different domains.

4.3.2 Blockchain-Enabled Designs for PHI Exchange

In terms of related work on PHI exchange, some papers (Hölbl et al. 2018, Angraal et al. 2017, Zhang et al. 2017, McGhin et al. 2019, Mackey et al. 2019, Yeung 2021, O'Donoghue et al. 2019, Mazlan et al. 2020) aim to discuss the benefits and challenges of the use of blockchain technology in healthcare by providing a comprehensive overview of the existing literature, identifying key limitations and technical issues that must be addressed to ensure successful implementation. Hölbl et al. (2018) presented the need for further research on the implementation and scalability of blockchain-based healthcare solutions. Angraal, Krumholz & Schulz (2017) evaluated some proposed existing solutions that could increase health data transparency and operating efficiency. The authors suggested that further research is needed to address limitations for large-scale production deployment,

including system scalability, security, and cost-effectiveness. Zhang et al. (2017) defined a set of evaluation metrics for blockchain-based healthcare decentralised applications to build the development of blockchain applications in the healthcare domain, which include cost-effectiveness, patient-centred care model, system scalability, interoperability, user identification, Turing-complete operations and compliance with HIPAA. But, these metrics are only concerned with the regulatory requirements of HIPAA.

McGhin et al. (2019) assessed nine types of existing blockchain-based applications in healthcare and pointed out limitations and technical issues of blockchain technology, such as scalability, mining incentives, standardisation, and key management. They noted that while blockchain technology has potential benefits for the healthcare sector, its limitations must be addressed to ensure successful implementation. Mackey et al. (2019) recommended a 'fit-for-purpose' health blockchain design framework as a guiding principle of application design. Yeung (2021) identified technical and practical challenges that must be addressed in order to fully realise the potential of blockchain technology in healthcare. These challenges include ensuring organisational commitment and interoperability, establishing internal governance and standardisation protocols, maintaining data security and integrity, ensuring quality and safety, and preserving truth and immutability. The author concluded that blockchain technology is unlikely to bring about a complete revolution in healthcare in the immediate future due to the numerous significant, multidimensional, and complex challenges that must be overcome for its adoption within healthcare applications.

O'Donoghue et al. (2019) discussed and analysed the design choices and trade-offs of the use of blockchain technologies in EMRs, which include trade-offs from the architecture factors of technology, data, application, and business. They recommended implementing an NHS consortium blockchain using a scripting language and PBFT consensus that has an adaptive block size and an adaptive number of confirmation blocks based on circumstances. Vazirani et al. (2020) also explored how blockchains manage EMRs, and how to build a more efficient and interoperable infrastructure to manage records that leads to improved healthcare outcomes, while retaining patient data ownership but without compromising privacy or security of sensitive data. Mazlan et al. (2020) explored and summarised the various scalability challenges in the healthcare blockchain system by 184 articles, which include block size, high volume of data, transactions, number of nodes, and

network protocol. Regarding issues about block size and high volume of data, the solution is to optimise storage. To consider challenges about transactions, number of nodes, and protocol, this paper mentioned redesigning the architecture of blockchains.

4.3.3 Blockchain-Enabled Implementations for PHI Exchange in Auditing, Security and Privacy

In terms of implementation of blockchain-based applications in healthcare, some literature focuses on how to improve system functionality and interoperability, data validation and auditing, regulatory compliance, privacy preservation, and so on. One proposed example is an attribute-based signature blockchain-based scheme with multiple authorities to encapsulate EMRs in the blockchain (Guo et al. 2018). In this scheme, patients endorse a message based on the attribute for each EMR, without disclosing any additional information except for the evidence they have certified. Multiple authorities generate and distribute patients' private and public keys for access authorisation. However, this scheme stores a large number of relational medical data in the blockchain, which may cause a problem of data retrieval (Fan et al. 2018). Another example (Zhang, Wang, Li, Zhao & Zhan 2018) is a secure and privacy-preserving health information exchange scheme for diagnosis improvement, which applies a private blockchain to store personal health information and a consortium blockchain to record the secure indexes of the PHI. Although both examples have the benefits of access control, data privacy, and immutability due to storing all data in the blockchain, they may be in violation of the EU's GDPR, which gives individuals the right to erase personal information and revoke access authentication.

Lee et al. (2022) designed a blockchain-based EMR-sharing system to effectively manage and share EMRs between different medical organisations. It uses a consortium blockchain to store and share the hashes value of EMR records and uses an InterPlanetary File System (IPFS) to store and share EMR records that balance the system throughput, latency, privacy preservation, and scalability. The use of re-encryption-based data encryption resolves the issue of data breaches from malicious users and key disclosure. The use of the private blockchain for EMR sharing in this proposed system causes a drawback to the consensus mechanism. Alrebdi et al. (2022) presented a searched and verifiable blockchain-based EMR system that enables to search, verify and store protected EMRs.

This system uses an IPFS and cloud to save patients' data and files and runs a smart contract for transaction verification by any outside related entity through a decentralised application. Although it improves the latency, further research needs to concern the cost, particularly in the cost of the function of adding new patients.

Monga & Singh (2022) proposed a decentralised MRBSChain EMR framework for medical data management between patients, administrators, and doctors based on Binance Smart Chain (BSC), ensuring security, privacy, confidentiality, scalability, interoperability, authentication, cost efficiency, and unified trusted record by implementing a security saving model and authentication role mapping. The deployment of the MRBSChain system in BSC achieves less transaction cost, deployment cost and average block time than in Ethereum. Mohey Eldin et al. (2023) introduced a federated blockchain system (FBS) to solve challenges of patients' unique identity, EMR security, and interoperability among different healthcare organisations by implementing components of authority, master and cache. Patients' EHRs and their hashes respectively store on the cloud and blockchain. Although FBS provides a physical card for patients to sign EHRs for confidentiality, it raises a problem of data disclosure if the physical card is lost.

Esposito et al. (2018) proposed an on-chain/off-chain scheme that stores health data in a conventional or distributed database and hashes values of these data in an on-chain way. However, this scheme has not been practically implemented (Fan et al. 2018). Zhang, Schmidt, White & Lenz (2018) implemented a use case called "Decentralised Application for Smart Health" to address issues of system evolvability, storage requirements, privacy, and scalability. Although they provided details on the application of the abstract factory, flyweight, proxy, and publisher-subscriber design patterns, they did not provide any experimental data to verify their proposal. They also presented a hybrid on-chain/off-chain framework named FHIRchain (Zhang, White, Schmidt, Lenz & Rosenbloom 2018) to improve security and scalability in clinical data sharing, which is designed to meet the regulatory requirements of the ONC. This hybrid framework is only compatible with systems that support FHIR.

Fan et al. (2018) proposed a blockchain-based information management system called "MedBlock" to address issues of large-scale data retrieval and sharing without additional costs and network congestion. However, they do not describe execution incentives regarding the entities that consent to the data exchange. Xia et al. (2017) developed the

"MeDShare" framework to improve security and data authentication in medical data sharing. However, the high level of security has resulted in additional latency in cloud-based services during high traffic times (McGhin et al. 2019). Azaria et al. (2016) built a "MedRec" system upon existing medical management databases that supports data sharing and access between multiple healthcare providers and enables authentication. They attempted to address mining incentive (McGhin et al. 2019) and improve the scalability of the system (O'Donoghue et al. 2019), but did not consider the issue of content attacks on existing databases and the potential difficulty of data auditing in EMRs.

Overall, when implementing applications with blockchain technology, it is also important to consider and assess the costs associated with deployment (O'Donoghue et al. 2019), business process redesign, hardware, and potential system replacement (Angraal et al. 2017, Esposito et al. 2018). The cost-effectiveness of the blockchain application must be assessed to ensure that it can efficiently process large amounts of data. However, there may be a bottleneck in system scalability as all blocks are stored on each node of the blockchain network (Angraal et al. 2017). Besides, the significant number of involved network entities in the blockchain to process data transactions increases the computational requirements of the infrastructure and consumes overhead resources (McGhin et al. 2019). The throughput of the blockchain network may also be limited by the maximum rate of transaction validation, which is dependent on the increasing computational devices in the network. Additionally, in the blockchain network, all data are encrypted and stored in the distributed ledger. The keys of network users are generated to allow access to encrypted data in the blockchain, but the current principles of key management may not be suitable for the blockchain (McGhin et al. 2019). If there is only one key to encrypt all blocks, it is unsafe if the key is lost or leaked. On the other hand, if there is a key generated for each block, it is not practical due to the high cost of storing and recovering the involved keys. If the key is lost or leaked, the consequences of data leakage can be significant. Therefore, careful consideration must be given to key management in blockchain applications.

Many companies have heavily invested in the use of blockchain technology, while these existing blockchain-based applications commonly work with their own standards and principles. That is an increasing potential issue of cross-interaction between different applications without standardisation. Some vulnerabilities are also considered in the use of blockchain technology, such as block withholding attacks, 51 percent attacks, double

spending attacks, selfish mining attacks, and block discarding attacks (McGhin et al. 2019). Additionally, the cost and energy consumption of blockchain-based applications is of concern. The PoW consensus algorithm used in some blockchain networks requires a significant amount of computing power and electricity to validate transactions and add new blocks to the chain. This can lead to a significant carbon footprint and energy expenditure, which may not be sustainable in the long term. There are alternative consensus algorithms, such as PoS, which are less energy-intensive, but they also have their own limitations and potential security concerns. It is important to consider the environmental impact of the use of blockchain technology and explore alternative solutions to reduce energy consumption.

4.4 Summary

Chapter 3 summarised the requirements of PHI exchange that includes a secure connection without mediation between transaction participants, interoperability, privacy, security, regulatory compliance, and patient engagement. To satisfy these requirements, there requires technologies to support data transactions between involved participants collaboration. As we mentioned in above section, blockchain technology has been identified as a disruptive innovation that exhibits unique characteristics such as transparency, anonymity, auditing, authentication, decentralisation, and security. These features make blockchain technology as a suitable platform supporting the exchange of PHI and addressing key challenges within the healthcare sector. Prior literature discussed above introduce the current research situation of how blockchain technology support PHI exchange in different aspects. The following summarises the details of how blockchain technology address challenges and satisfy requirements of PHI exchange, including trust-building barrier, interoperability, security, auditing, and low patient engagement.

- Trust-building barrier. To build a secure connection for data transactions between multiple involved participants is important for PHI exchange. Blockchain technology provides transparent and unchanged data transaction channel, and its approaches mentioned in above literature generates automatically audit records for verification about data transactions if required. Besides, it makes data transactions during exchange period between participants without mediators. Smart contracts run on

blockchains take roles of mediators involved.

- Interoperability. It is defined as the ability of one computer system to exchange data with another, with three levels of increasing complexity: basic, functional, and semantic (Lumpkin et al. 2000). Basic-level interoperability refers to data exchange without interpretation ability, while functional interoperability involves data exchange in a defined format. Semantic-level interoperability requires interpreting and understanding the meanings of the exchanged data. Different healthcare organisations have their own specifications for collecting, querying and storing data in the system. Without a pre-defined data structure and format for exchanging data, health providers may have difficulty interpreting or misinterpreting the shared information. Therefore, it is necessary to establish clear data structures and define formats to facilitate the interoperability of high-quality sharing. To achieve this, a high level of standardisation across diverse organisations is necessary to enable data interaction and exchange across different infrastructures and applications (McGhin et al. 2019). Blockchain-enabled implementations mentioned in above literature relies on smart contracts to guide data storage and exchange in a pre-defined data structure or format. Also, all participants can share required decentralised data through blockchain networks with anonymity but can be verified if required.

- Security and auditing. Kish & Topol (2015) highlighted in their work that as data sets grow larger, they become increasingly attractive targets for cyber attacks. This is a serious concern in the healthcare sector, where incidents of cybersecurity breaches have led to data breaches and network security issues on a global scale. According to a report by Verizon, the healthcare sector is second only to the finance industry in terms of the number of cyber-attacks annually, with 68 percent of threat actors from internal sources (Verizon 2017). The average total cost of a healthcare data breach was found to be 3.62 million US dollars per incident in 2017 (Snell 2017). At least half of all healthcare breaches have been caused by hackers in the breach incident that was reported during the first half of 2017 (Caban 2017), and over 100 million medical records were publicly disclosed during the first half year of 2015 in America (Collins 2015). Wanna Decryptor malware affected the British NHS in 2017 that has resulted in the cancellation of approximately 19,000 appointments (BBC

NEWS 2017). These incidents demonstrate the critical importance of implementing effective cybersecurity measures in the healthcare sector to protect sensitive PHI from unauthorised access, theft, or loss. Blockchain-enabled approaches discussed in above literature improve data security during PHI exchange with data verification. If participants share data with others through a blockchain network, an evidence with data transaction information is saved in the networks without tampered. Also, these implementations run smart contracts to generate a special audit trail as an evidence or verify the existing evidence records. A blockchain can be adopted in PHI exchange only as an audit server to verify audit records or as a non-human mediator to share, generate, save and audit data.

- Low patient engagement. In 2009, Clancy, Anderson & White (2009) emphasised the urgent need for health information technology investments to promote healthcare information exchange. However, the success of PHI exchange depends on patients' willingness to share their PHI, which can be influenced by privacy concerns. Patients are typically willing to share their PHI (Hersh et al. 2015), if potential privacy concerns are addressed (Abdelhamid et al. 2017). On the other side, unsolved privacy issues may significantly affect patients' willingness that consents to share their data. Resolving privacy concerns is paramount to ensuring patients' willingness to participate in the exchange of PHI and unlocking its full potential in improving healthcare outcomes. Prior literature mentioned above use blockchain-enabled applications to improve security and data privacy, or involve system automatic operations to reduce human operation errors so that patients can trust to share their data.

The adoption of blockchain technology in PHI exchange has significant potential to improve security and privacy, build trust, and support interoperability, although there remains room for improvement. Previous literature has introduced limitations and benefits in addressing these critical requirements of PHI exchange. For all involved participants in a PHI exchange workflow including patients, health service providers, third parities and so on, these discussed mentioned above are crucial considerations, encompassing not only a secure channel for sharing health data but also the need for explicit consent for data exchange. As such, a blockchain-based framework that satisfies these requirements of PHI exchange is essential. The three proposed blockchain-enabled approaches intro-

duced on chapter 5 are designed for auditing, security, and privacy preservation of PHI exchange between multiple participants within a single organisation or across multiple organisations.

Chapter 5

Frameworks for PHI Exchange Using Blockchain

This chapter focuses on the design of secure data exchange frameworks in the healthcare sector using blockchain technology. The goal of these frameworks is to meet the requirements of auditability, non-repudiation, privacy preservation, security, compatibility, integrity, and confidentiality when sharing PHI between involved entities within and across boundaries. In the context of PHI exchange, the AudiWFlow framework builds an audit trail for data exchange between different entities involved in a single jurisdiction; the BRUE framework focuses on cross-jurisdiction PHI exchange; and the BRESPE framework is designed for privacy-preserving PHI exchange.

5.1 Overview

Controlling the sharing of sensitive data is an open problem. Beyond the impact of the loss of data itself, data breaches also bring a sharp negative impact on the public's trust and discourage them from engaging with electronic systems to share their data (Reddick & Anthopoulos 2014). Auditing the workflow is thus essential when handling data flows, especially when dealing with healthcare data sharing. In addition, PHI involves sensitive personal information that requires strict access control. Individuals or organisations may need to access partial or complete personal health information under certain circumstances. That has motivated researchers to explore solutions that balance the privacy protection of healthcare information with the convenience of information access. In

a distributed workflow, the requirements of non-repudiation and accountability are also essential to maintain the connectivity of healthcare information.

Consider a health insurance scenario shown in Figure 5.1, its vertices are the collaborating entities, including a customer (Alice), an insurer (Bob), and doctors (Cathy, Dippy, and Eva). The directed edges in the figure are the sequence of actions and deliverables. In contrast with Bob the Insurer, doctors are registered and employed in the NHS. The act of data exchange between doctors is presented in the figure as a sequence of numbers from 5 to 8. The data exchange interactions between these doctors take place within the NHS Spine [1]. The interactions between Bob and Alice take place outside the Spine.

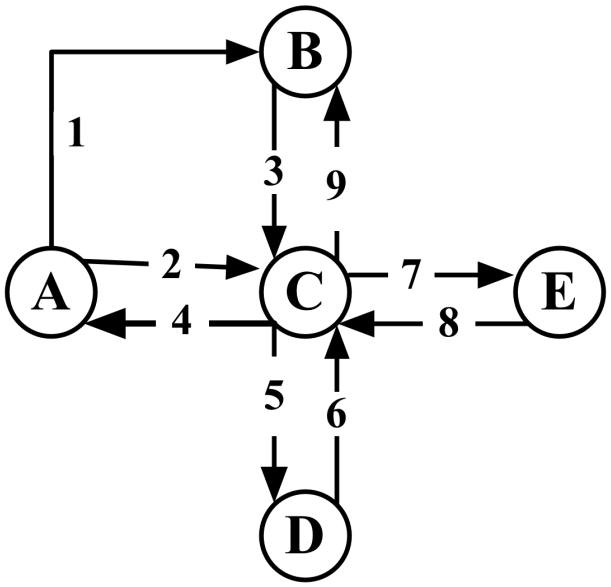


Figure 5.1: The example scenario of health insurance (Zhou et al. 2022)

Alice (A) wants health insurance from the insurer Bob (B). Alice has to provide her medical history to Bob and allow Bob to contact her doctor, Dr. Cathy (C). Insurer Bob also needs Alice's family medical history, but this has to go through Dr. Cathy who, following a confidentiality friendly approach, will provide an overall report after she contacts the family doctors Dr. Dippy (D) and Dr. Eva (E). To save on insurance costs, Alice asks her doctor, Dr. Cathy, to provide an untruthful medical record. Bob thus obtains Alice's medical history from Dr. Cathy (which includes reports from Dr. Dippy and Dr. Eva). Happy with the outcomes, insurer Bob offered a deal with which Alice was satisfied. After two years, Alice claimed compensation after a medical incident. Referring to Alice's insurance claims, Bob investigated all documents of Alice and found that Alice's

[1]NHS Spine: it allows information to be shared securely through national services such as the Electronic Prescription Service, the Personal Demographics Service, the Summary Care Record and the e-Referral Service. `https://digital.nhs.uk/services/spine`

medical records provided by Dr. Cathy contradict records in the hospital. Insurer Bob rejects Alice's claim. Alice counter-argued that the insurance company was responsible for collecting her medical history and she is thus entitled to compensation. Insurer Bob then tries to obtain the contacts of Dr. Dippy and Dr. Eva, which Dr. Cathy refuses to provide on the grounds of confidentiality.

As we see, some entities are involved which are independent and, until then, individually unaccountable to any other entity. All have to collaborate toward the end result, which is health insurance for Alice. A further aspect is that some entities are unknown to other entities: B does not know who are D and E and, in fact, only knows that there are other entities beyond C. In the absence of perfectly shared information, there is no way for any entity to verify the accuracy of the information provided. For example, insurer Bob is in a position where he cannot prove that Alice is at fault. This is because Alice can claim that insurer Bob colluded with doctors from the hospital and modified her original documents to close a sale. Insurer Bob also cannot find any traces of a collusion attack between Dr. Cathy and Alice. Finally, Dr. Cathy denies ever signing the records that insurer Bob holds and raises the suspicion that her signature was forged. This scenario illustrates the challenges of obtaining evidence in the case of a distributed multiple-entities workflow within and across domains. It also highlights that having a robust audit system with an immutable audit trail is vital to ensure non-repudiation and assign accountability for malpractice (Nehme et al. 2019).

Regarding data exchange in the distributed workflow, as we discussed in chapter 4, blockchain technology cannot be overlooked. The inherent properties of immutability and distribution of this technology have made it useful in both financial and non-financial domains (Crosby et al. 2016, Nofer et al. 2017), such as government public management (Nofer et al. 2017), healthcare sector (Guo et al. 2018, Fan et al. 2018), and privacy preservation in data sharing workflows (Gai, Wu, Zhu, Qiu & Shen 2019, Gai, Wu, Zhu, Xu & Zhang 2019). Blockchain also enables peer-to-peer transactions without intermediaries or trust relationship agreements between entities. To explore blockchain technology in facilitating the exchange of PHI while ensuring confidentiality, privacy preservation, auditability, and compatibility, this chapter proposes three frameworks which are AudiWFlow, BRUE, and BRESPE. The first framework is primarily focused on PHI exchange between different entities located in a single jurisdiction. It explores how blockchain

technology supports data sharing compared to traditional centralised server frameworks.

A simple auditable blockchain-based framework has been proposed to handle the requirements for auditing in data transmission operations (Zhou et al. 2019). The proposed scheme relies on public key cryptography (PKI), a group of signatures, record verification, and Shamir's secret sharing scheme (see Appendix B.1) to create an auditing trail that ensures confidentiality, integrity, and accountability for all actions of entities involved in the data transmission process, irrespective of the generic topology and data flow. The use of PKI encrypts the exchanged messages, which enhances the confidentiality of the workflow. Shamir's secret sharing scheme improves the security of encrypted data. Theoretically, using a partial key to decrypt audit records is infeasible (Shamir 1979). Using a group of signatures ensures data integrity for each transaction. Audit records verification is an essential component, enabling entities to check the correctness of audit records equivalent to a received transaction. In this scheme, the blockchain plays the role of an audit server to share and save audit records and credential keys. A certificate authority is also selected to manage workflow credential keys. Entities are required to trust this third party which can securely generate and distribute workflow key pairs as agreed. As an initial model, it is well structured while it still involves a third party, except for the workflow entities involved in the data transmission. The involvement of a third party in the process raises the risk of key disclosure.

By modifying the above auditable blockchain-based model, AudiWFlow improves the efficacy of auditing data transactions and entities' operations (Zhou et al. 2022). The AudiWFlow framework provides a solution for PHI exchange between different entities located in the same jurisdiction, such as data exchange between healthcare service providers (e.g., GP and registered dentist in the NHS). It is designed to be confidentiality-friendly and collusion-resistant which significantly reduces the risk of third-party involvement in malicious activities. The key management in AudiWFlow involves the selection of one of the entities to serve as the key generator and distributor. Entities share the necessary data with another entity that includes an audit record linking with a previous data transaction. This process creates a local complete audit trail between entities. Blockchain saves the audit record after the data transaction by the data sender. All entities can then verify the transaction on-the-fly and after the fact based on the local audit trail and online audit records.

This chapter also focuses on addressing the specific requirements of PHI exchange, particularly in improving privacy preservation and legality (user-informed consent), while ensuring operational feasibility. The feasibility of PHI exchange is contingent upon various scenarios, such as sharing between entities across jurisdictions. The BRUE and BRESPE frameworks are proposed and designed.

BRUE applies blockchain technology in user authorisation management in the distributed workflow (Zhou et al. 2020). It is designed for secure cross-jurisdiction exchange of PHI with the consent of the data subject. BRUE requires entities to share permission information through the blockchain while minimising the required personal data sharing in the workflow without authorisation. To improve the feasibility of data exchange across jurisdictions, BRUE selects relevant local and cross-regional organisations as the role of authority servers to process the service request. Personal receipts are returned to the data sender as integrity proof after the data transaction.

To enhance privacy preservation in peer-to-peer communication, the BRESPE framework is proposed. This framework manages user-informed consent and preferences in the exchange of PHI. It aims to explore the secure exchange of PHI on a blockchain with "sticky policies" that govern the use of data in a user-friendly way, given the varying needs of different entities involved in information sharing. BRESPE uses cryptographic protocols for exchanging information and manages credential keys and personal receipts in the same way as BRUE. Additionally, it builds data exchange policies based on user preferences and data protection regulations to stick to the exchanged data in a digest format to prevent redundant information from being stored in local storage.

The following elaborates three sections, and each presents a unique framework designed to address specific requirements of secure data exchange using blockchain. Section 5.2 introduces a fundamental framework called AudiWFlow, which is designed to provide a complete audit trail to ensure data authenticity and entity behaviours. It can be used to audit data transactions of PHI exchange between entities located in the same jurisdiction. Section 5.3 presents the proposed framework BRUE, which aims to meet the specific requirements of PHI exchange across jurisdictions. Section 5.4 shows the development work of the BRESPE framework, which emphasises privacy preservation and user consent management in PHI exchange using blockchain technology. The final section discusses how the three frameworks meet the security requirements.

5.2 AudiWFlow: Blockchain-based Auditing of Data Exchange in Distributed Data Workflow

Auditing operations in multiple-entities data exchange and over an arbitrary topology are common requirements yet still an open problem, especially in the case where no trust in any participating entity can be presumed. Challenges range from the storage of the audit trail to the tampering and collusion of participating entities. We provide a solution to the accountability problem that arises when entities collaborate towards a common result, such as companies in a supply chain or government departments working together. In the absence of a fully trusted central point, obtaining a trusted audit trail for a workflow can be difficult when each entity is unaccountable to the others.

To address this issue, we propose AudiWFlow, an auditing architecture that makes entities accountable for their contributions in a distributed workflow. AudiWFlow provides confidentiality, collusion detection, and evidence availability after the workflow is terminated. The framework is built on Shamir's secret sharing scheme and real-time peer-to-peer verification of records and supports multiple levels of assurance to balance evidence availability and auditing overhead. Additionally, AudiWFlow uses smart contracts running on a public blockchain, removing the need for any central point (third party) of control. This section presents the design of AudiWFlow and describes how it meets the security requirements.

5.2.1 Introduction

Distributed workflows involving multiple organisations cooperating toward a certain outcome are a common way to leverage the potential of the Internet. This is common in many domains including governments, digital health, education, engineering, supply chains, goods distribution, etc. Collaboration is enabled by interoperable applications through which each organisation contributes to a workflow. A key enabler is trust: organisations need to trust each other in that each will perform their part as contracted. When a problem occurs, the workflow needs to be audited to determine what failed. With massive digitisation, nearly every domain has similar needs – supply chains (Tian 2017), inter-department business processes (Weber et al. 2016), e-government services (Hart-

mann & Steup 2015, Pappel et al. 2017), etc. Note that the problem becomes trivial if a central entity is able to coordinate and gather evidence; however, trusting a central party is a difficult problem in itself, especially in a distributed workflow where parties may not even know each other beyond their adjacency. Furthermore, a simple log record is not enough as any valid evidence cannot be open to manipulation (Guan et al. 2019). A further problem is a collusion between a central entity managing the workflow or between two adjacent entities in the workflow topology to tamper with digital evidence. Even if the orchestration of the workflow is managed in the cloud, a privileged insider can tamper with the logging process. Finally, confidentiality requirements should be noted (Zawoad et al. 2016). In a pure distributed workflow, organisations may want to only deliver the expected outcome and not disclose any other information.

To illustrate the problem of data sharing between multi entities, consider using a linear topology - see Figure 5.2. Entities (A - E) represent the involved organisations or individuals that share data in the workflow. The arrow represents the direction of data flow. The processes of data flow and the related entities are pre-established, which means the interaction between workflow entities is pre-defined. A is the information sender, who wants to send information to B. A knows the receiver is B and B knows the sender is A. If an outside attacker plants forged data instead of the payload that B sent to C, we need to ensure that the honest entity C can detect this action. If B colludes with D in that they tamper with the existing audit information and repudiate performed actions to avoid incrimination during the inspection, there should be enough evidence to make honest entities spot the incorrect data. If confidential data is exfiltrated, it is necessary to ensure that the data is encrypted and only minimal information is exposed.

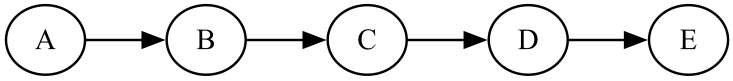


Figure 5.2: The representation of data exchange workflow

To address these challenges, we propose AudiWFlow, a blockchain-based smart auditable check scheme that generates evidence in real time, guaranteeing the integrity, availability, and unforgeability of all audit records. AudiWFlow replaces the need for a third party to record and share audit trails, using a public blockchain with smart contracts. The double-lock mechanism, using key pairs of the workflow and entities within the workflow, allows for encrypted message exchange and encrypted audit records be-

tween entities. AudiWFlow provides confidentiality, integrity, and availability assurances for audit records, reflecting the contributions of each entity in the workflow. This section defines the problem and presents the architecture design of AudiWFlow.

5.2.2 Problem Statement

The proposed scheme is designed to address the following security requirements in the threat model:

1. Confidentiality and integrity. All workflow entities cannot forge or tamper with existing information after the fact. Only the data owner can generate correct encrypted audit logs. The blockchain nodes and workflow entities cannot forge or tamper with the audit logs, even if they are dishonest individuals or collude with others. Besides, the audit logs are only stored and verified in cipher form. They cannot be intentionally exposed in plaintext form. In other words, they cannot be viewed or modified in an undetected or unauthorised way. The audit server only stores the related encrypted audit logs and public keys.

2. Availability. No entity should be able to destroy evidence at any time after release. Entities cannot escape the audit processes when they require a service. All encrypted audit logs are tamper-resistant and stored in the blockchain. The honest entity can access the audit trail to verify the received data.

3. Collusion detection. If two or more entities collude, this can be detected quickly.

4. Non-repudiation. No entity should be able to dispute the recorded evidence.

The above security aspects help to achieve accountability assurance that is enabled by having reliable evidence. The security model renders the proposed approach suitable for applications where the confidentiality of digital evidence is required. It also aims to assure the availability and integrity of audit trails.

The proposed approach targets the key requirements including *accountability*, *non-repudiation*, *confidentiality*, *availability*, and *collusion detection*. Figure 5.3 shows a linear workflow with two tiers. The audit server includes the code of a smart contract run on the blockchain that is trusted to perform the protocol, which stores audit records and

conducts the verification triggered by the workflow entities. In the simple topology of Figure 5.3, entity A starts by requesting work from B; B then requests work from C in order to complete the request from A. Then it continues to D. When D performs the expected action based on the request from C, the workflow terminates. The evidence generated while the workflow progresses is composed of individual audit records. Should a dispute arise at a point in the future, this evidence must hold all entities accountable for their contributions.

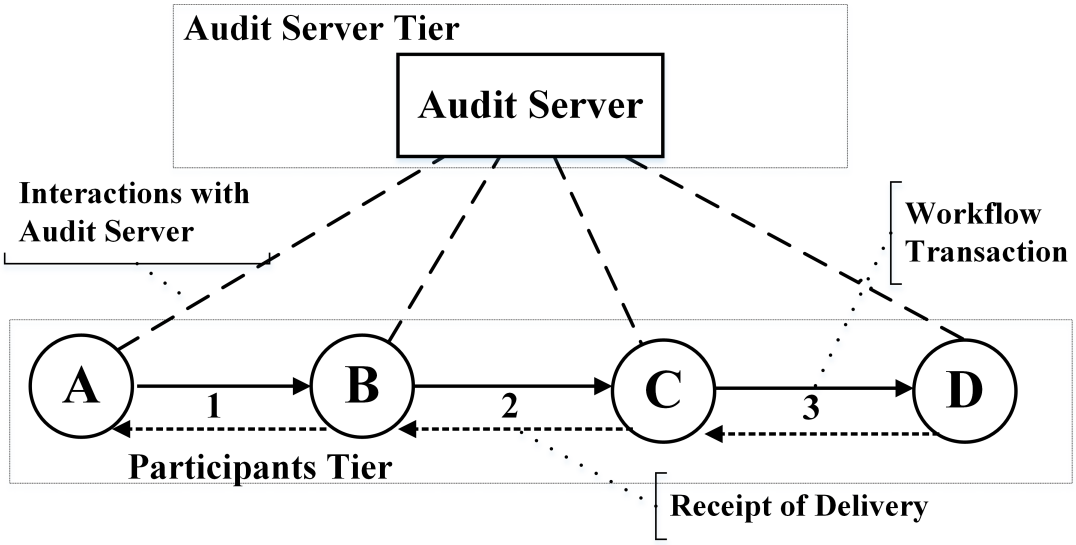


Figure 5.3: The representation of an auditing architecture with two tiers (Zhou et al. 2022)

In the workflow of Figure 5.3, entities are trusted but some of them may collude with others to intentionally deny their mischievous actions or modify existing information in storage after the fact. The outside attackers can eavesdrop on a message from the transmission channel and plant a forged message instead of the true one in the workflow. Any entities in the workflow can collude with others to repudiate the performed actions. Therefore, the proposed framework is constructed based on the following assumptions:

1. The blockchain is deemed to be trusted to immutably store data.

2. The workflow entities do not intentionally expose their private keys.

3. There is at least one honest entity in every workflow.

5.2.3 System Architecture

The proposed framework employs a public blockchain to execute smart contracts, which exchange a valid digest of an audit record for an attestation of the integrity of the verified record. That minimises the risk of third-party involvement and malicious acts. The audit record is shared, alongside the exchanged message, with the next entity. Both the sender and recipient store the audit record to compare it with a digest of the message in the blockchain. We encrypt all messages with workflow or entity keys. For any distributed workflow, the approach provides a robust and confidentiality-friendly way to record and verify audit records at any desired granularity, while giving auditing capability to entities. The subsection describes the notation and terminology used in the framework, shows the structure of the architecture, and outlines the protocol of the proposed framework.

Notation

The proposed framework uses the following notation:

- a workflow $\mathcal{W}$ executes over a directed graph $G = (V, E)$ and is associated with audit evidence $\mathcal{A}^{\mathcal{W}}$ produced during its execution.
- $V = \{A, B, C, ...\}$ is the set of entities involved in a workflow.
- $E = \{1, 2, 3, ...\}$ is a set of the sequence number of entities' actions in a workflow.
- $pk_{\mathcal{W}}$ and $sk_{\mathcal{W}}$ are, respectively, the public and private keys of a workflow $\mathcal{W}$.
- pk_i and sk_i are, respectively, public and private keys of entity $i = 1, 2, ..., N$ with $|V| = N$ entities.
- k_j is the j-th share of a threshold key, in the sense of secret sharing, and $j = 1, 2, ..., N$. It is derived from $sk_{\mathcal{W}}$ and any threshold $K \leq N$ members can recover the key. Appendix B.1 presents details of Shamir's secret sharing.
- $M_1 \| M_2$ denotes the concatenation of messages M_1 and M_2.
- M_{ij} is a message sent from entity i to j.
- $\text{sign}_i(M)$ is a message M signed by entity i.

- $\text{enc}_i(M)$ is message M encrypted with pk_i.
- $\text{enc}_{\mathcal{W}}(M)$ is message M encrypted with $pk_{\mathcal{W}}$.
- $\text{hash}(M)$ is a digest of message M using a one-way collision-resistant function (a "hash").

Architecture

Figure 5.4 shows a view of the system architecture of the proposed scheme, which includes two main components: entities and audit server. Entities refer to individuals or organisations involved in the workflow, such as authorities and stakeholders. Each entity collaborates and exchanges information in a workflow. A public blockchain plays the role of an audit server. All audit records are encrypted and then stored in the blockchain for access by entities. The blockchain acts as a trusted messaging channel such as halting a workflow and notifying all entities. Noted that audit records also need to be stored somewhere outside the blockchain. An entity can choose this location for record storage. The hash of each record stored on the blockchain ensures the authenticity and integrity of the data. Each entity can immediately access audit records and store them locally. If an entity fails to retrieve a specific audit record or receive a response from the chosen data storage, it can broadcast an alert on the blockchain to warn other entities about this failure. In the workflow, we assume that all entities have a unique identification that can be used across different workflows. Each workflow has a single pair of keys that can only be used in this specific workflow. To identify all relevant entities, all of them have their unique pair of keys for transactions. Every public key is stored in the blockchain. Entities save their private keys and use them to approve transactions in the workflow.

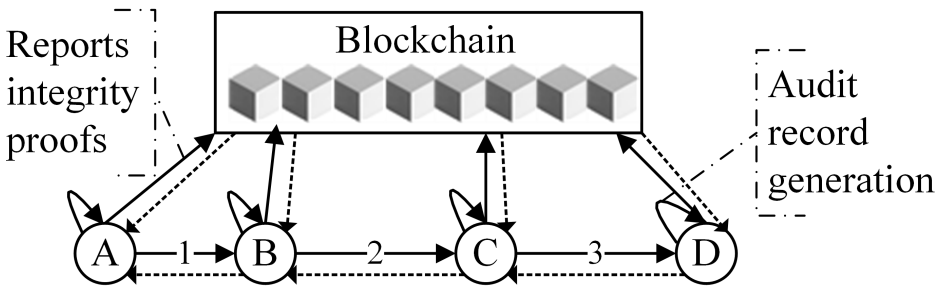


Figure 5.4: The system architecture of the AudiWFlow framework (Zhou et al. 2022)

Key Management

The scheme proposed for securing the data exchange in the workflow involves the use of two types of keys, namely entity keys and workflow keys. These keys need to be securely generated, distributed, and validated.

- Entity keys. We assume that credentials pk_i and sk_i for every entity have been managed before the workflow starts. For example, conventional certificates can be verified. It should be noted that storing entities' public keys for a specific workflow can potentially reveal the number and identity of entities. We assume that this is an acceptable relaxation and leave it for future work.

- Workflow keys. We need to generate and distribute the keys $pk_{\mathcal{W}}$ and $sk_{\mathcal{W}}$ for each workflow $\mathcal{W}$. The threshold keys k_j are derived from $sk_{\mathcal{W}}$: it uses Shamir's secret sharing scheme (Shamir 1979) to create N shares, where K shares are enough to reconstruct the secret. Workflow key distribution can either be done through direct messages to each entity over a secure channel or by encrypting each share of the key with the corresponding entity's private key and posting them to the blockchain. We use a pragmatic approach and task the participant with the least incentive to corrupt the audit trail to generate and distribute the key shares. This is generally the first or last entity depending on the workflow: a first entity in a workflow can be a gift shop salesman required to keep track of orders for customers, and the last entity in another workflow can be a supermarket manager that needs to keep track of where products ordered by customers are from.

Protocol

The protocol used to implement the AudiWFlow framework consists of the *initialisation*, *data exchange*, and *records verification and distribution* phases. Key generation and entity registration take place in the initialisation phase. In this phase, all involved workflow entities are required to register on the blockchain network before any actions can be taken. Entities can push and pull records to or from the blockchain. We need to generate a key pair for the workflow: $pk_{\mathcal{W}}$ and $sk_{\mathcal{W}}$. The workflow needs a trust anchor (maybe the first entity of the workflow) to bootstrap the process for key generation. We once

more delegate the choice of the entity that coordinates the distribution to the specific use case, such as to the entity with the least incentive to be malicious. Whereas $pk_{\mathcal{W}}$ is stored in the blockchain for public access, the k_i split shares of $sk_{\mathcal{W}}$ are distributed to each entity. We assume that each entity has a cryptographic key pair previously generated and all have stored and made available their public keys in the blockchain. When an entity wants to get a public key from the blockchain, a smart contract is called in order to find the respective key.

Figure 5.5 presents a message sequence chart of the proposed protocol, which includes the main phases of *data exchange* and *records verification and distribution*. The following message formats are used in this message sequence chart.

- authenticity

$$P_{ij} = \text{enc}_j(\text{sign}_i(M_{ij}))$$

  The entity i sends a plaintext output M_{ij} to j and, to assure authenticity in a future audit, it signs the message. For confidentiality, the entity i encrypts the result with j's public key.

- receipt

$$R_{ij} = \text{sign}_j(\text{enc}_{\mathcal{W}}(\text{sign}_i(M_{ij}))$$

  After i sends an output to j, j is returning a receipt of delivery to i.

- audit record

$$A_{ij} = \text{sign}_i(\text{enc}_{\mathcal{W}}(\text{sign}_i(M_{ij}))$$

  This message generates an audit record which, for confidentiality, is encrypted with the public key of the workflow. The resulting object is then signed again with i's private key as an entity will need a verification of the record without the need of inspecting its contents. The complete audit trail of workflow $\mathcal{W}$ is $\mathcal{A}^{\mathcal{W}} = \{A_{12}, A_{23}, ..., A_{ij}, ...\}$ with indexes that match the graph path of the workflow.

- integrity proof

$$I_{ij} = \text{hash}(A_{ij})$$

This message simply extracts a digest of an audit record by the sender.

- integrity proof for verification

$$I'_{ij} = \text{hash}(A_{ij})$$

This message extracts a digest of an audit record by the recipient.

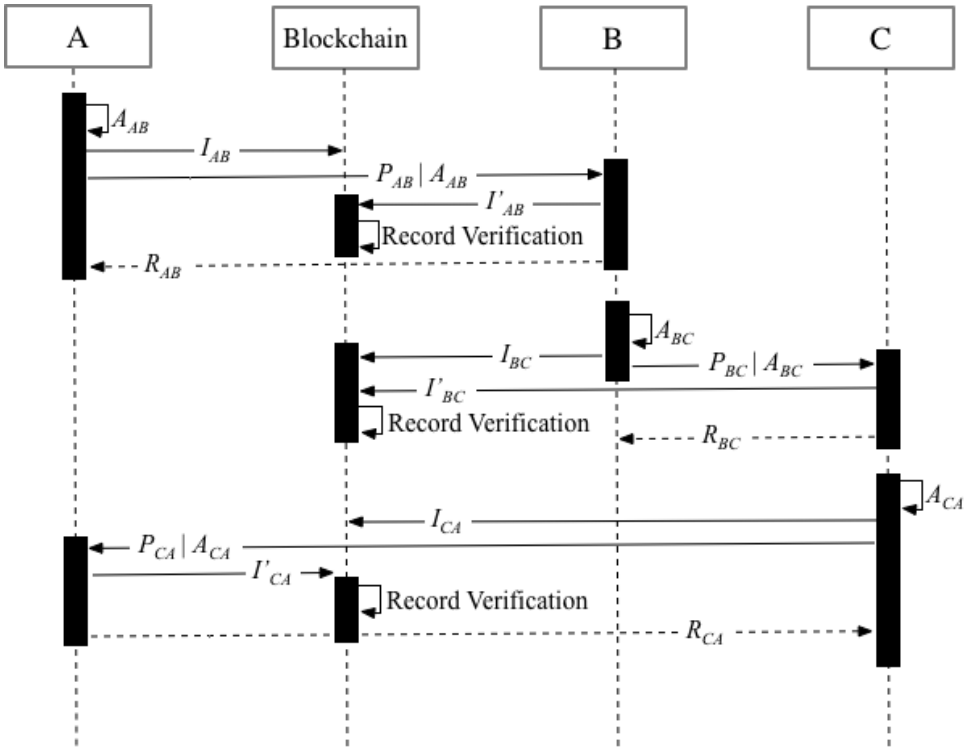


Figure 5.5: A message sequence chart of the AudiWFlow framework (Zhou et al. 2022)

During the data exchange phase, entities sign and encrypt the exchanged message to ensure confidentiality and accountability. The message sender packages the required data based on the requirements and then shares it with the receiver. The exchanged data package includes an encrypted message with a signature and an audit record. The sender also saves this audit record locally for data validation in the future. The recipient receives this exchanged data package. As shown in Figure 5.5, entity A starts the data flow with a message M_{AB} and generates a payload P_{AB} followed by its audit record A_{AB}, which is then sent to entity B. Entity A signs message M_{AB} and encrypts it with a public key pk_B of B to construct the payload P_{AB}, which is represented by $P_{AB} = \text{enc}_B(\text{sign}_A(M_{AB}))$. The audit record A_{AB} is evidence proof that records the required exchanged message M_{AB}. It is encrypted with a workflow public key $pk_{\mathcal{W}}$ by entity A with a group of signatures. The

representation of A_{AB} is $\text{sign}_A(\text{enc}_{\mathcal{W}}(\text{sign}_A(M_{AB}))$. Peer-to-peer transactions between B and C, C and A are similar to the mentioned transaction between A and B.

Entities will cycle through a record verification and distribution phase. The sender generates and pushes an audit record to the audit server (blockchain) as integrity proof. The recipient receives the data package from the sender and then generates its audit record to push into the blockchain for record verification. The record verification compares two audit records respectively from the sender and the receiver. After verification, the receiver responds with a receipt to the sender. Algorithm 1 shows the pseudo-code of a smart contract for record verification. This algorithm performs a string comparison operation to compare the audit record with the integrity proof (the digest value of an audit record) stored on the blockchain. In Figure 5.5, entity A needs to generate an integrity proof I_{AB} and push it to the blockchain. The proof I_{AB} is represented by hash(A_{AB}) that extracts the hash digest of the audit record A_{AB}. Entity B receives the data package $P_{AB}|A_{AB}$ and then calculates the hash digest of the audit record I'_{AB} from the data package to send it to the blockchain for verification. This comparison allows B to check whether the data has been intentionally modified. Then, B returns a receipt R_{AB} to A to end the data transmission between A and B. This receipt R_{AB} is signed by B with the protected message M_{AB} and a signature from A. The data flow ends there.

Algorithm 1 Smart Contract on Record Verification

Input: *OwnerAddress* ▷ unique address
Integrity_Rec ▷ integrity proof reported by the recipient
Output: Boolean indicating if a record is verified successfully

```
1:  function compareLogs(Integrity_Rec, OwnerAddress)
2:      Hash_Rec[] ← retrieveLog(OwnerAddress)
3:      i ← 0
4:      while i <Hash_Rec[].length do
5:          if Hash_Rec[i]==Integrity_Rec then
6:              return True
7:          else
8:              i ← i + 1
9:          end if
10:     end while
11:     return False
12: end function
```

5.3 BRUE: User-Controlled, Cross-Jurisdiction, Auditable Sharing of Healthcare Data Mediated by a Public Blockchain

Due to the sensitive nature of the information, different regional data protection regulations and standards impose severe limits on what can be exchanged, even in case of emergencies. Furthermore, systems in different jurisdictions do not communicate. To address this challenge, this section tackles the problem of sharing PHI across jurisdictions. It proposes the BRUE framework allows PHI to be securely exchanged, with the data subject always in the position of mediator. BRUE is designed to ensure auditability, confidentiality, and decentralisation in cross-jurisdiction PHI exchange.

The main contributions of BRUE are: firstly, to minimise sensitive information exchanged in the sharing network. It only verifies identity information in the local jurisdiction and avoids the need to check and share identity information across jurisdictions. Secondly, the framework creates lightweight and short-lived authorisation tokens that are shared between entities to access PHI in different regions. For example, a token with identity information, which represents the verified identity without disclosing any sensitive information, is produced by the authorisation server. All entities share and verify these tokens to identify each other. Tokens are shared through the blockchain by means of smart contracts, which are system-agnostic for the existing infrastructure. That improves system scalability. Since tokens can be revoked at any time with immediate effect, it promotes compliance with virtually all regulations. Thirdly, BRUE reuses the concept of Personal Data Receipts (PDRs) from the Data Protection Research Community (Jesus 2020). BRUE provides acknowledgement receipts for all operations of the involved entities. Receipts not only meet accountability requirements (e.g., for data controllers) but also provide a means for data subjects to trace past access. Finally, cryptographic access is required for any exchange between two entities. For this matter, BRUE applies the well-known Diffie-Hellman key exchange scheme (Diffie & Hellman 1976) to be used in a blockchain, providing the useful result of proving, beyond any doubt, that the two entities were engaged. It further provides forward secrecy and confidentiality. The following subsections explain the design details of the BRUE framework.

5.3.1 Introduction

The increasing globalisation of people's travel has led to a situation where individuals may seek medical treatment from different healthcare providers across different jurisdictions. eHealth data of individuals is managed by diverse health service providers and stored in different locations. Although there are many agreements between different jurisdictions (such as countries), in general, they do not allow health information to be shared externally. Often, sharing is not even allowed within the same country (e.g., in China) between different healthcare providers. There are several reasons but, as a general theme, it is due to a lack of trust or data disclosure considerations stemming from compliance and regulations. There is a consensus that PHI involves sensitive personal information that must be well protected. To tackle the problem of sharing PHI across different jurisdictions, BRUE is proposed. The main challenges stem from three fundamental problems. First, there needs to be full accountability (e.g., non-repudiation) when sharing data; accountability also refers to the possibility of someone sharing PHI without the authorisation of the patient. Second, since there is no global infrastructure to discover the locations of health information, it must be a truly decentralised scheme. Third, before sharing PHI, explicit consent must be obtained from the data subject. The health information custodian (e.g., healthcare service provider) needs to be able to demonstrate that appropriate measures had been taken to address these problems if there is an audit or breach in the future.

In the design of BRUE, the framework tackles the problem by, first, centring all the information exchange and control on the data subject. This user mediates all steps by the resource owner and is effectively the (cryptographical) trusted communication channel between all entities. Second, it combines a set of technologies and standards, each addressing a particular requirement. For authorisation and management of access to records, BRUE uses and extends User-Managed Access (UMA) (Machulak & Richer 2016). For a trusted and confidential communication channel, distributed discoverability, and overall accountability, BRUE uses a public blockchain able to run smart contracts. To handle the requirement of demonstration of valid consent, BRUE uses PDRs. This combination of technologies motivates the name of BRUE: Blockchain, Receipts, and UMA for eHealth data exchange (BRUE). The remainder of the section defines the problem and then presents the design details of the framework.

5.3.2 Problem Statement

To illustrate the problem of cross-jurisdictional exchange of PHI, we informally analyse a simple working scenario of international PHI exchange. Alice (A) is a French citizen with a history of heart trouble and originally registered with a local General Practitioner FGP in France. She currently resides in the UK and is registered with a local GP (BGP). When Alice travelled to Canada at 20 years of age, she fell sick and visited a GP (CGP) in Canada to receive temporary treatment. After that trip, she returned to the UK and visited her UK GP. She wants to share with her UK GP her previous medical data stored with healthcare service providers in Canada and France. However, she does not want to simply share her personal credential with her UK GP. Instead, she wants to authorise her UK GP to access her medical data using her GP's credential.

The scenario involves four entities: the data subject A, the requesting party BGP, and two data controllers which are healthcare service providers FGP and CGP. Note that BGP, CGP, and FGP are independent entities located in different jurisdictions. They are not known to each other and their only point of contact is their relationship with A. Furthermore, as is overwhelmingly the case, assume that there is no global system in place that allows all entities to directly communicate, find each other, or self-certify. In other words, BGP needs to access A's medical data from FGP and CGP but FGP and CGP do not recognise BGP so A needs to mediate the request and grant access. A, in turn, needs to authenticate against FGP and CGP. To address the challenges of cross-jurisdictional PHI exchange, a distributed architecture is needed that ensures auditability, non-repudiation, confidentiality, and compatibility. To tackle these requirements, Figure 5.6 provides a simple example of data exchange between two entities.

All data flow of transactions from entities go through the blockchain network (for provenance and accountability). A receipt of each transaction is also produced following the data flow. In the sketch, entities N and M prepare R-wallets to store receipts[2]; N then requests services from M by invoking smart contracts running on the blockchain. M processes the request from N pulled from the blockchain. When M returns the expected outcome and pushes the result to the blockchain, N then obtains the outcome from the blockchain. The fact is that the communication channel is the blockchain itself to

[2]The framework reuses the familiar term of "wallet" as the (digital) container of a receipt

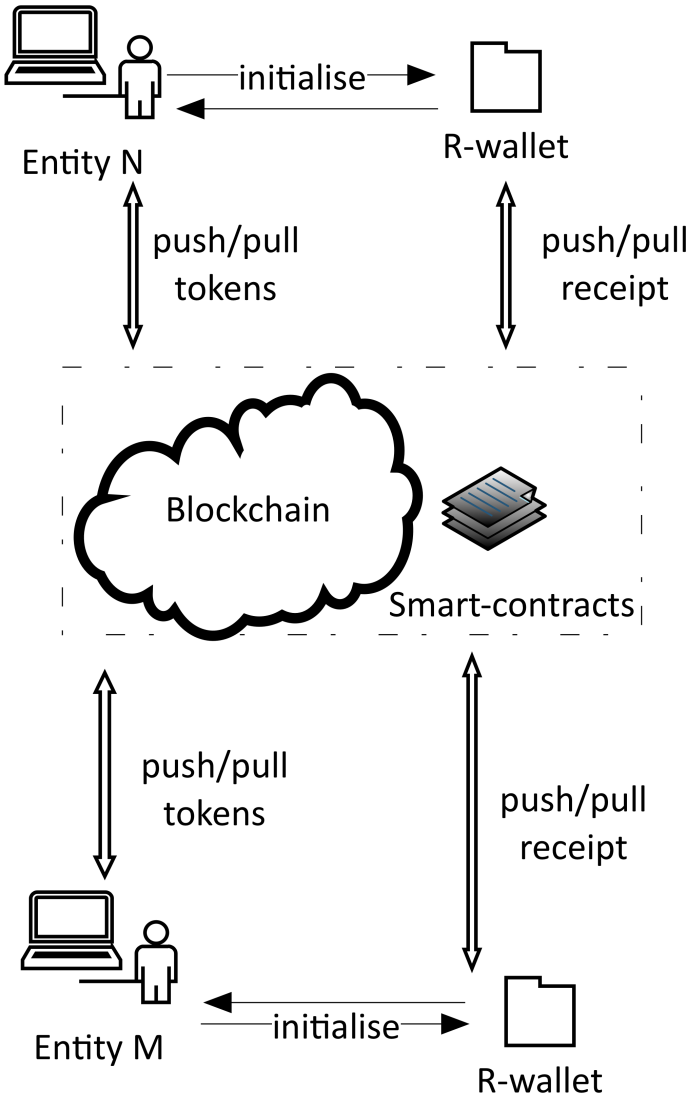


Figure 5.6: Data exchange between two entities (Zhou et al. 2020)

guarantee traceability. The receipt as transaction proof is generated and follows the data flow in both directions. The peer-to-peer data exchange between N and M terminates at this point. The receipt generated while the data flow progresses is composed of individual audit records. Should a concern or dispute arise in the future, this receipt holds all the evidence needed to keep all entities accountable.

Figure 5.7 shows a general scenario of a cross-jurisdiction PHI exchange flows with different entities, including the data subject, requesting party, data controller, and verifier/authoriser. Each entity contributes to the overall workflow. The topology of entities is established at the start according to the service request and the direction of data flow and does not change during the specific workflow. It is assumed that the data subject has agreed with the data controller and verifier to exchange its data, which is represented by dashed lines in Figure 5.7. The data controller has already collected and stored the PHI of the data subject. We assume that the workflow graph is acyclic with each entity

being either a sender or recipient throughout the workflow. In other words, data flows sequentially through the graph such that no entity plays the role of both message sender and recipient in a workflow. BRUE is designed on this basis and it is agnostic in terms of the actual health data format.

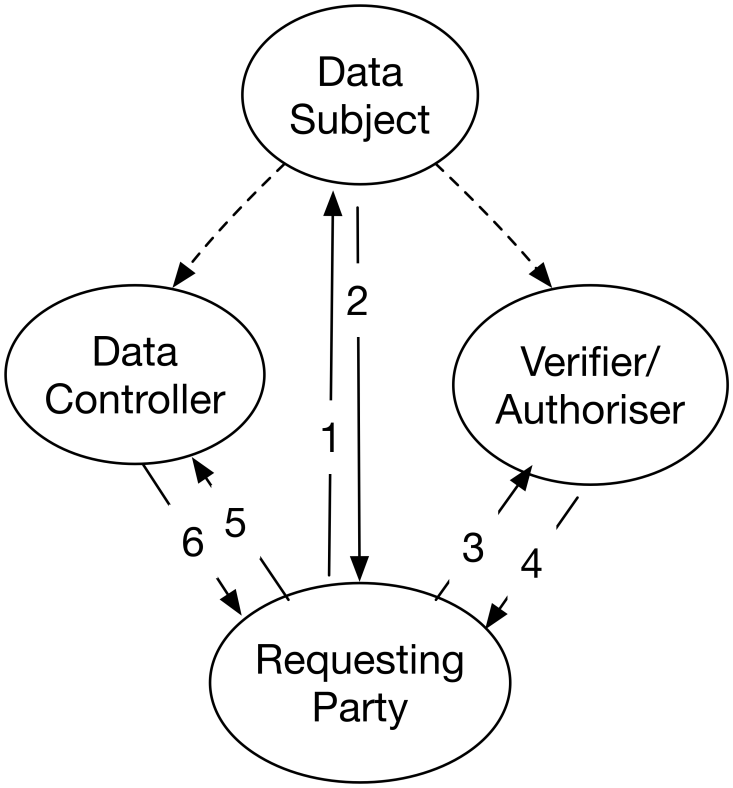


Figure 5.7: Multiple-entities data exchange flow (Zhou et al. 2020)

5.3.3 System Architecture

BRUE provides a possible solution for cross-jurisdiction PHI exchange while ensuring auditability, compatibility, non-repudiation, and confidentiality. It integrates UMA standards and receipts and runs on a public blockchain to achieve these goals. The following sections present how the proposed framework satisfies these requirements, starting with the notation, followed by the structure of architecture and protocol.

Notation

BRUE uses some roles from the UMA workflow (see Chapter 2 section C), including resource owner, authorisation server, resource server, and requesting party. We qualify the roles of the authorisation server as local and remote authorisation server, and the resource server as local and remote resource server to support the PHI exchange across

domains. In addition, we re-design the permission ticket as the permission token that grants remote access permission and construct the requesting party token for the last permission verification between the (remote) resource owner and requesting party. We also add tokens to manage authorisation information between the local authorisation server and requesting party called verified identity tokens. The following summarises these notations:

- *DS* is the *data subject* whose health information is being collected and who should be informed before sharing.
- *RO* is the *resource owner* on behalf of the data subject, an entity with the ability to grant access to protected resources in the resource server.
- *RqP* is the *requesting party* that wants to access the health records.
- *AS* is an *authorisation server*, an organisation that is authorised to manage access to protected healthcare information. We assume, for simplicity, that each jurisdiction has only one *AS*.
- *LAS* means local authorisation server, which is an *AS* located in the same jurisdiction as *RqP*.
- *RAS* is a remote authorisation server, which is an *AS* located in a different jurisdiction from *RqP*.
- *RS* is *a resource server* that stores and manages the actual healthcare records, such as hospitals.
- *LRS* is a local resource server, which is an *RS* located in the same jurisdiction as *RqP*.
- *RRS* is a remote resource server, which is an *RS* located in a different jurisdiction from *RqP*.
- *PT* is a *permission token* that presents the consent of *RO* to the *RqP*'s request for data access.
- *PT*' is a re-encrypted permission token produced by *RqP* to share with *LAS*.

- VIT stands for *verified identity token*, which is identity proof authorised by LAS for RqP.
- VIT' is a re-encrypted verified identity token produced by RqP to share with RAS.
- RPT represents a *requesting party token* which is proof of the data access permission with specified conditions authorised by RAS to RqP.
- RPT' is a re-encrypted permission token produced by RqP to share with RRS.
- $k_{N,M}$ is a shared secret key for entities N and M in the data flow following the Diffie-Hellman key exchange method.
- $RE_{N,M}$ is a receipt of a transaction between entity N and M.

Architecture

Figure 5.8 illustrates data flow in BRUE. RqP (e.g., a regional hospital) located in Jurisdiction 2 needs to access the PHI of DS from RS located in Jurisdiction 1. It is noted that DS selects RO as the role of its delegation in the data exchange workflow to process the request, and then RO has agreements with authorisation servers and resource servers. Those are presented with the dashed lines in the figure. The data flow is represented by the solid lines from sequence numbers 1 to 10 and the solid line with sequence number 11 presents the final step of PHI exchange in the figure. RqP sends a request to RO; RO checks the service request and then returns PT. PT includes the consent from RO about data access. RqP receives PT, re-constructs PT as new token PT', and then signs PT' to share with LAS. LAS receives PT' to verify the identity and permission information and builds token VIT to respond to RqP. RqP receives VIT and re-encrypts it as token VIT'. RqP sends VIT' to RAS for identity and authorisation checks in a different jurisdiction. RAS generates RPT and then grants it to RqP after checking for VIT'. RqP receives RPT and re-encrypts it as new token RPT' to share with RRS. RRS receives RPT and checks its information. Then, RRS returns the required PHI to RqP. The entire data flow terminates at this point.

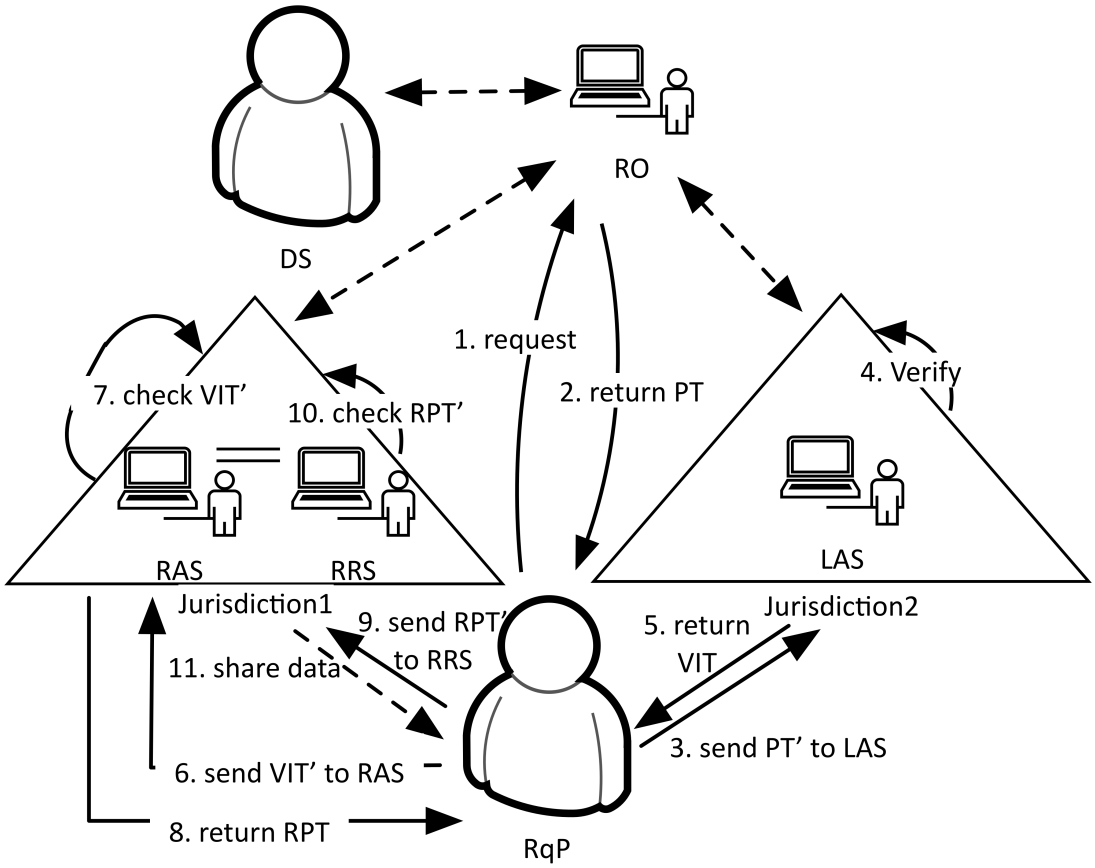


Figure 5.8: The system architecture of the BRUE framework (Zhou et al. 2020)

Key Management

BRUE applies the Diffie-Hellman key exchange (see Appendix B.2) method to produce secret keys for token exchange encryption. This provides full accountability of requests and non-repudiation. Whereas writing in a blockchain requires a secret key (tied to the specific blockchain), reading from a public blockchain is open and unaccountable. This key $k_{N,M}$ is generated which is shown in Figure 5.9. Entity M, as a requester, pushes a modulus p and base g to the blockchain for entity N. Entity N then pulls them from the blockchain. M and N publicly agree to use p and g for key generation. After that, M and N respectively select a secret random integer a or b. N pushes A to the blockchain for M and M sends B to N through the blockchain. Then, N pulls B from the blockchain and generates a secret key $k_{N,M}$ with calculation. M does work similarly to N to obtain an agreed secret key $k_{N,M}$. Thus, they have a shared secret key $k_{N,M}$.

Token Exchange

Tokens are designed to grant access to data. In the process, a record is generated in the form of a receipt which is retained by the entity if it is later audited. Entities share

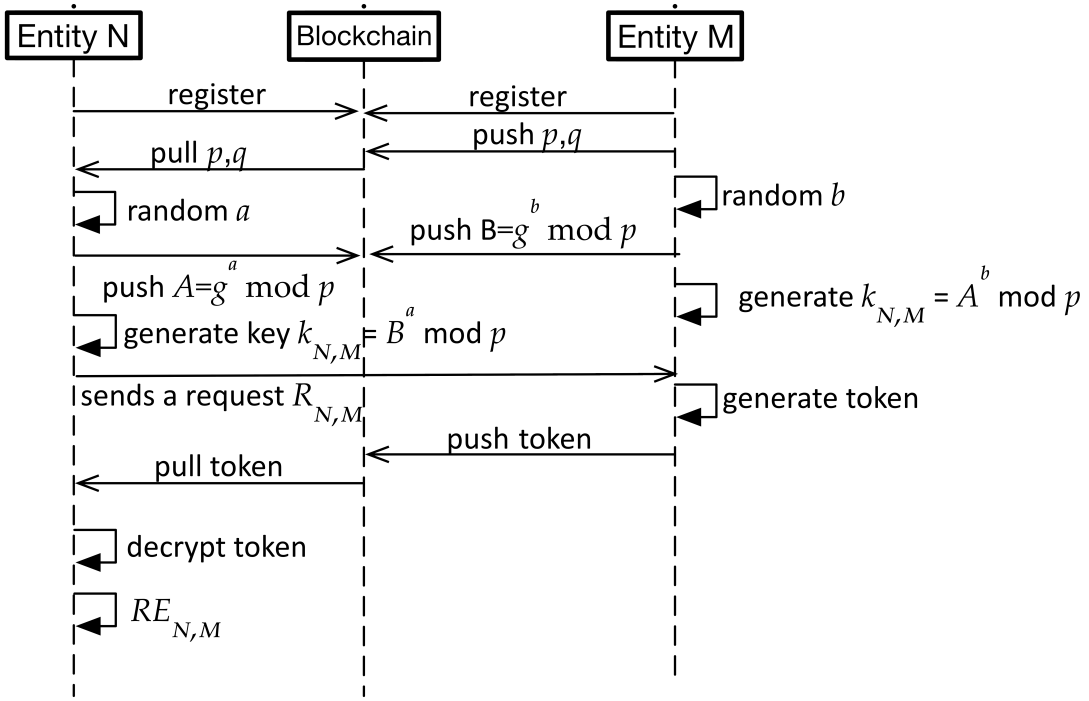


Figure 5.9: Key generation using Diffie-Hellman key exchange method over the blockchain (Zhou et al. 2020)

tokens with each other only via a public blockchain running smart contracts. There are two kinds of tokens. One is generated between RO or AS with RqP. When RO or AS generates a signed token, they send it to the blockchain encrypted by key $k_{RO,RqP}$ or $k_{AS,RqP}$. Then, RqP receives the token and decrypts it with a shared key $k_{RO,RqP}$ or $k_{AS,RqP}$. Another token is used between RqP and AS or RS. RqP shares an authorised token PT with LAS, or shares VIT with RAS, or shares RPT with RRS through the blockchain. Before tokens can be exchanged, RqP needs to sign and encrypt the tokens. After a token push, AS or RS pulls the protected token from the blockchain and decrypts it with key $k_{AS,RqP}$ or $k_{RS,RqP}$. The next subsection about protocol elaborates on tokens in the BRUE framework.

Receipt Management

BRUE uses receipts for transaction records to ensure auditability. When new information (token) is shared between entities, a receipt is generated. For the BRUE, there are two kinds of receipts:

- Receipt for a service request. If RqP starts a service request to access the PHI of DS, a receipt is generated and includes its hash digest.

- Receipt for a new token generated and shared. When entity N generates a token

and then pushes it onto the blockchain, a receipt is returned to N that has the hash value of this transaction.

Protocol

The protocol of BRUE consists of five phases: *initialisation*, *service request*, *identity verification*, *authorisation verification*, and *resource exchange*. The initialisation phase is a preparation step that includes entity delegation, key generation, and entity registration.

- Delegation. The data subject needs to authorise a resource owner and select some authorisation servers and resource servers. RO is only delegated as the representative of DS. DS and RO can be the same individual or different entities. For instance, if DS is a child, RO could be a parent. Furthermore, there is only one AS required and some RSs in the same region. The number of RS is dependent on the selection from DS.

- Registration. After entity delegation, entities are required to register. This step is executed based on UMA. All entities are required to register in the blockchain and initialise their R-wallets. An R-wallet is a client application used to aggregate all generated receipts. RS is also required to register in its local AS. Therefore, AS manages a list of RSs. In other words, LAS has a list of LRS, and RAS has another list of RRS. A receipt is generated to record the registration of RS for the auditability principle.

- Key generation. The secret key of an entity is generated using the described variant of the Diffie-Hellman key exchange scheme over a blockchain (see details in section 5.3.3). This key is a common shared between two entities. For RO and RqP, the key is $k_{RO,RqP}$. The key $k_{AS,RqP}$ is for AS and RqP, and the key of RS and RqP is $k_{RS,RqP}$.

Figure 5.10 shows the message sequence chart of the BRUE framework, including the main phases of *service request*, *identity verification*, *authorisation verification*, and *resource exchange*. The following notations are used in the sequence chart.

- $R_{N,M}$ is a service request sent from entity N to M.

- $R_1 \| R_2$ denotes the concatenation of the request R_1 and R_2.
- I_N expresses the identity of entity N.
- $\text{sign}_N(R_{N,M})$ is a request $R_{N,M}$ signed by entity N.
- $\text{enc}_{k_{N,M}}(R_{N,M})$ presents a request $R_{N,M}$ encrypted with key $k_{N,M}$.
- $\text{hash}(R)$ is a digest of the request R using a one-way collision-resistant function (a "hash").

Tokens and receipts in the proposed framework are presented in the following data formats:

- permission token

$$PT = \text{enc}_{k_{RO,RqP}}(\text{sign}_{RO}(PT))$$

  Entity RO sends a token PT to RqP as a response to the request $R_{RqP,RO}$ and, to assure auditability, it signs the message. For confidentiality, RO encrypts the token with the shared secret key $k_{RO,RqP}$. A plaintext of PT includes the consent status of $R_{RqP,RO}$, URL of LAS, note, the start date of the permission, the expired date of the permission, the name of DS and RqP, and signature of RO.

- re-encrypted permission token

$$PT' = \text{enc}_{k_{LAS,RqP}}(\text{sign}_{RqP}(PT))$$

  Entity RqP receives PT from RO and extracts content of PT, then it encrypts and signs the protected information again to share with LAS using own key $k_{LAS,RqP}$ and signature.

- verified identity token

$$VIT = \text{enc}_{k_{LAS,RqP}}(\text{sign}_{LAS}(VIT))$$

  Entity LAS sends a token VIT to RqP with authorisation after verification as a response proof. VIT is signed by LAS and encrypted with the secret key $k_{LAS,RqP}$. A plaintext of VIT is made of the consent status of data access, URL of RAS,

the status of verification, the start date of the permission, the expired date of the permission, note, the name of DS and RqP, and signatures of RO, LAS, and RqP.

- re-encrypted verified identity token

$$VIT' = \text{enc}_{k_{RAS,RqP}}(\text{sign}_{RqP}(VIT))$$

Entity RqP receives VIT from LAS and extracts content of VIT, then it encrypts and signs the protected information again to share with RAS using own key $k_{RAS,RqP}$ and signature.

- requesting party token

$$RPT = \text{enc}_{k_{RAS,RqP}}(\text{sign}_{RAS}(VIT))$$

Entity RAS shares a token RPT with RqP after verification. RPT is signed and encrypted by RAS with the secret key $k_{RAS,RqP}$. A plaintext of RPT includes the consent status of data access, URL of RRS, the status of verification, the start date of the permission, the expired date of the permission, note, the name of DS and RqP, signatures of RO, RRS, LAS, and RqP.

- re-encrypted requesting party token

$$RPT' = \text{enc}_{k_{RRS,RqP}}(\text{sign}_{RqP}(RPT))$$

Entity RqP receives RPT from RAS and extracts content of PRT, then it encrypts and signs the protected information again to share with RRS using own key $k_{RRS,RqP}$ and signature.

- receipt

$$RE_{N,M} = \text{sign}_N(R_{N,M}) \| \text{hash}(R_{N,M})$$

$$RE_{N,M} = \text{sign}_N(PT) \| \text{hash}(PT)$$

$$RE_{N,M} = \text{sign}_N(VIT) \| \text{hash}(VIT)$$

$$RE_{N,M} = \text{sign}_N(RPT) \| \text{hash}(RPT)$$

After entity N shares information with M, a receipt of delivery is produced that includes this encrypted transaction and a digest.

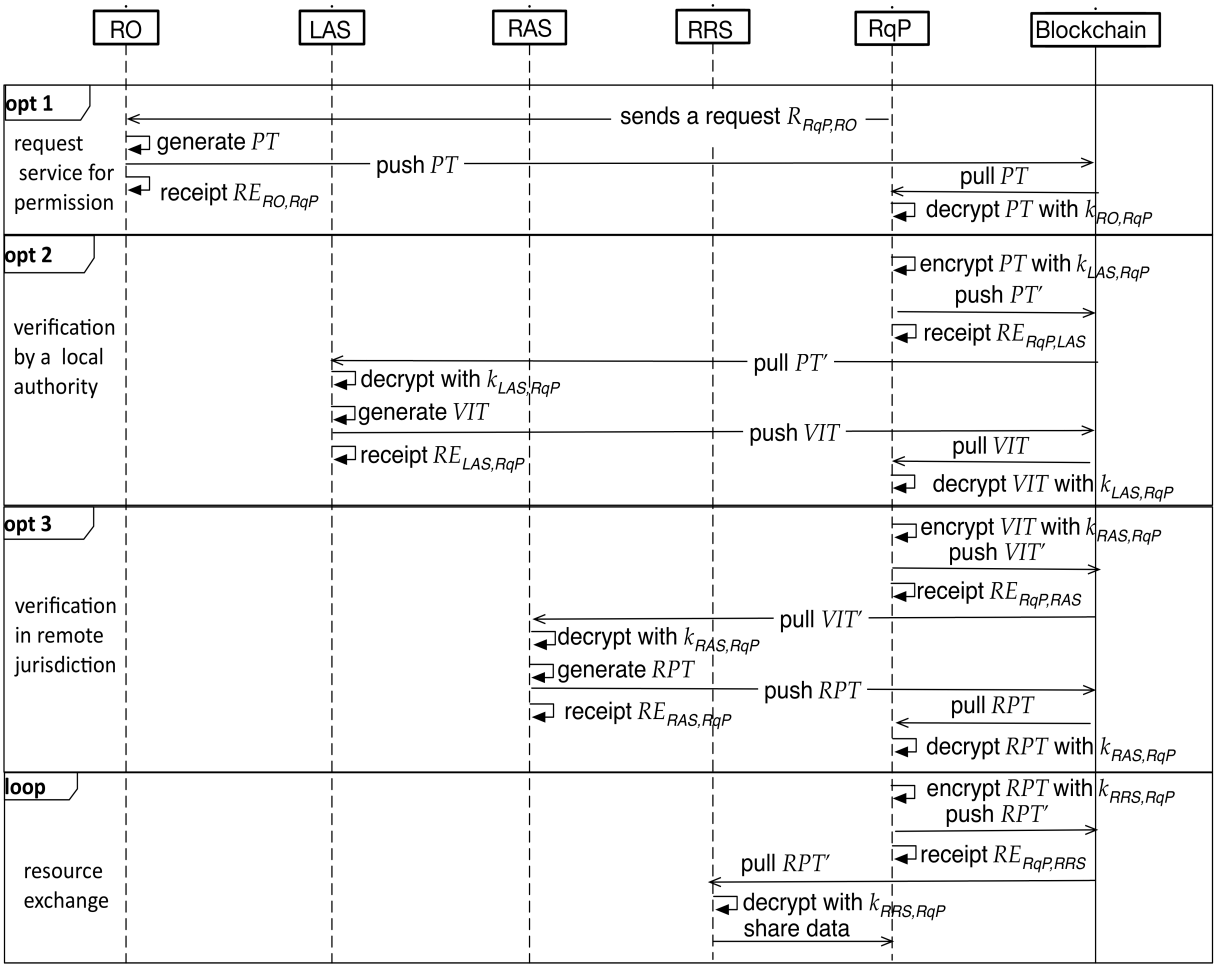


Figure 5.10: The message sequence chart of the BRUE framework (Zhou et al. 2020)

The service request phase happens between RO and RqP and starts the data flow. RqP sends a service request $R_{RqP,RO}$ to RO without any permission for PHI access of DS. A receipt follows. As seen in Figure 5.10, RO checks $R_{RqP,RO}$ and then generates a token PT. PT is signed and encrypted by RO with a secret key $k_{RO,RqP}$. RO pushes this token PT to the blockchain, and RqP pulls it from the blockchain. RqP receives the information of permission authorisation. At the same time, a consent receipt is produced following the push of PT. Note that this transaction is also recorded in the blockchain.

The next phase is about identity verification. To meet the requirement of confidentiality, we do not share the identity information of entities across jurisdictions. It needs to be conducted by an authority located in the same region. In Figure 5.10, requires RqP to first verify identity at a LAS prior to any data exchange. RqP signs and encrypts PT with the secret key $k_{LAS,RqP}$ to share with LAS. RqP finally sends a new token PT' with LAS through the blockchain. Then, LAS receives the shared information through decryption

with the key $k_{LAS,RqP}$. If PT' is verified successfully, LAS generates and sends VIT to the blockchain. VIT is an identity verification proof with permission information for RqP. RqP pulls VIT from the blockchain and decrypts it to obtain permission information and a link to RAS. There are two types of receipts involved as transaction evidence. One of the receipts is generated when RqP pushes the token PT to the blockchain. The other is generated when VIT is pushed by LAS to the blockchain.

In the phase of authorisation verification, entity RqP initialises the cross-jurisdiction request with the identity proof token VIT. RqP signs and re-encrypts VIT with the key $k_{RAS,RqP}$ to share with RAS through the blockchain. Here a receipt is generated for the token VIT exchange transaction. RAS pulls VIT' and decrypts it with the key $k_{RAS,RqP}$ for authorisation verification. When RAS acknowledges the information in VIT', RAS sends a token RPT to RqP for data access. RPT is signed by RAS and encrypted with the key $k_{RAS,RqP}$. After that, RqP pulls and decrypts RPT to confirm and obtain permission information and RRS's contact information. A receipt is produced for the transaction of a token RPT pushed by RAS. This phase ends at this point.

Lastly, in the phase of resource exchange, RqP is now able to access the PHI of DS from RRS. RqP sends an encrypted token RPT' to RRS with its signature. A receipt follows the token sharing. RRS pulls RPT' from the blockchain and decrypts with the key $k_{RRS,RqP}$ to confirm whether RqP is authorised to access the specific PHI. Then, RRS shares the protected PHI with RqP following a receipt as transaction record proof.

5.4 BRESPE: Towards Privacy-Preserving Healthcare Data Connectivity through Sticky Policies and Public Blockchains

This section proposes a framework called BRESPE that is designed to ensure that PHI exchange respects privacy requirements. The framework is used to exchange data between the data subject, data requester, and data controller. It constructs user-preference policies linked to data transactions run on smart contracts by a public blockchain, provides a reliable way to manage consent ensuring authenticity and traceability, and uses the Diffie-Hellman key exchange scheme to encrypt the exchanged permission information

into ciphertext signed by entities ensuring confidentiality and data integrity. To simplify the operation of data access, BRESPE optimises operations between entities based on a rule that if an entity stores data, it is the main player in handling data transactions.

With BRESPE, a data subject has the right to directly control access to its healthcare data and needs to be informed which part of its data is shared with whom, under what conditions, and at what location; a data controller enforces policies of data protection compliant with regulations in the data process and builds trust for data subject ensuring privacy protection. In conclusion, BRESPE is designed to exchange PHI in compliance with data protection regulations and user preference for privacy preservation. The following subsection defines the problem and shows the design of the BRESPE system architecture.

5.4.1 Introduction

Population mobility raises the problem of balancing the need for data interoperability and privacy protection in the healthcare sector. To comply with regulations and protect data privacy, it is necessary to manage access authorisation in data transactions between entities. The scenario of data exchange workflow involves patients, healthcare organisations, and third parties. Patients are data subjects who grant permission to access the data. Third parties are data requesters who want to access data from healthcare organisations. Healthcare organisations are data controllers that are on behalf of data subjects to manage and process related health data. They are independent entities with individual requirements. The data requester and the data controller have only a relationship with the data subject, respectively. We assume that there is no method for both to allow them to communicate directly with or recognise each other. The data requester wants to directly access the data subject's healthcare record from the data controller, while the data controller does not recognise the data requester. Therefore, the data subject needs to mediate the request and then grant access to the data requester. The data subject has the right to authenticate the data transaction and concerns personal data privacy protection during the period of data processing, but may have a poor understanding of data protection regulations. The data controller needs to share the specific data with consent from the data subject in compliance with data regulations. In summary, data subjects have

legal rights to authenticate the operations of data requesters and data controllers to know whether they process data as agreed. The data controller also needs to build trust in the data subject in the business context. That highlights four challenges to healthcare data sharing: (1) data privacy protection and confidentiality for data exchange, (2) auditability of data transactions, (3) policy enforcement, and (4) user consent management.

Data privacy protection and policy enforcement are important requirements in health data sharing. Sticky policy, as a mature technology in compliance with data protection regulations, is successfully applied in data sharing for different domains. The terminology of it was first mentioned by Karjoth, Schunter & Waidner (2003). Conventional approaches provide data privacy protection at a certain static level in data transmission. The data controller is usually the rule maker to formulate policies of data processing permanently for all users with no changes. The data subject must trust the data controller without options and agree with rules from the data controller that protects data privacy in the processing period. Some approaches with sticky policies have benefits for data privacy protection at the per-user and per-data levels. These policies are dynamically formulated. Although the data controller is also the rule maker, when the data controller is required to share the data, it timely formulates the specified policies based on the user preferences and data protection regulations compared with non-sticky policies made in the prior approaches. That supports the data controller to enforce policies ensuring data privacy. Besides, the benefit of using a sticky policy is that the data controller has always performed data privacy protection policies in healthcare data sharing if the request for data access occurs. The prior approaches using sticky policy help data subject to the data transaction (Miorandi et al. 2020), protect data privacy and enforce policies in compliance with data protection regulations by the data controller. The data controller saves and processes policies in the local storage. The data requester starts the request services and also saves the relevant policies of the transaction in the local storage. It raises a problem of huge duplicate policies stored respectively in the data controller and data requester. Besides, the problem of malicious actions from data controllers is not fully recognised and addressed. A dishonest data controller constructs policies to share data with the data requester but is not in compliance with data subject preference or data protection regulations and can deny its dishonest actions after the fact.

To explore the effect of combining the technologies in PHI exchange ensuring privacy

protection, confidentiality, and auditing, the BRESPE (Blockchain, receipts, sticky policy, healthcare data exchange) framework is proposed that leverages sticky policy, PDRs, and blockchain to protect data privacy and ensure confidentiality and provides convenient for granted access to healthcare data exchange. In conclusion, the combining of these technologies aggregates respective advantages and offsets some of their limitations since: (1) Blockchain builds trust without involving a central trusted authority or mediator and supports data confidentiality. It also provides non-tampered, confidential, and permanent records for data transactions ensuring data privacy and auditability. In the BRESPE framework, entities share minimal personal sensitive data in the progress of authorisation and verification in the blockchain network before the sharing of actual health data, which is to avoid a problem of data disclosure. The framework also constructs and shares sticky policies on the blockchain that solves the problem of duplicate local storage. (2) The adoption of PDR mends the completeness of the audit trail and enhances the non-repudiation of actions. A receipt is proof of data transactions ensuring auditability and traceability. (3) Sticky policy gives the right to users to control and authenticate their data in the data exchange between entities. It provides flexibility in policy formulation and enforcement. Compared with the BRUE framework, BRESPE is mainly designed for per-data/per-user privacy preservation in the data transaction. The authorisation is constructed based on user preference and also recorded as proof of integrity for audit. The following subsections show the details of the framework to achieve the requirements, define the problem, and describe the architecture of the framework.

5.4.2 Problem Statement

In the context of BRESPE, all involved entities must register in a public blockchain network, which is assumed to be trustworthy and capable of executing the proposed protocol as agreed upon using smart contracts based on its inherent characteristics. Data stored on public blockchains are saved permanently without being tampered with or deleted. Public blockchain servers serve as a communication channel for sharing associated encrypted information and as a non-tampered space for saving records as evidence proof. It is assumed that the data subject is honest in operating the proposed protocol and sharing its data with consent. The data requester and the data controller are honest and

trustworthy, but may be interested in processing data beyond the boundaries of the data subject's preferences.

Therefore, two threat models are considered: (1) Shares policy without consent – the data controller constructs a sticky policy without the data subject's consent to support data exchange between entities. (2) Shares required PHI without consent – the data controller shares PHI with the data requester beyond the agreed sticky policies or the data subject's consent.

To ensure data privacy and confidentiality in the data flow, BRESPE is designed with the following features: (1) Auditability and non-repudiation: all data transactions of permission and authorisation need to be conducted on the blockchain, and receipts are provided as evidence to entities when each data transaction ends. The data stored on the blockchain cannot be tampered with. BRESPE provides consent management to trace actions timely and enables auditing of data transactions after the fact. (2) Confidentiality: dishonest entities cannot read plaintext information stored in encrypted files on the communication network. All exchanged data are shown on the public transmission channel as ciphertext. (3) Data privacy: entities are required to set up policies in compliance with data protection regulations and user preferences to manage data exchange. The exchanged data in the public communication channel has not or minimally involves personal sensitive data. Dishonest entities cannot learn sensitive information in the exchanged encrypted data files through the public communication channel.

5.4.3 System Architecture

BRESPE leverages a public blockchain, sticky policy, and PDRs to implement data sharing and privacy protection. It uses the Diffie-Hellman key exchange (Diffie & Hellman 1976) scheme to generate a secret key for data exchange encryption between two entities, thereby enhancing confidentiality. Whatever data input in a blockchain requires a secret key to encrypt, while data withdrawal from the blockchain is open only if an associated key is present. The secret key generation principle is consistent with that described in (Zhou et al. 2020) and is therefore not presented in this section. The following notations are used in BRESPE, and the system architecture is introduced to meet the requirements.

Notation

BRESPE uses the following notations to present:

- DS is the *data subject* whose PHI is being collected and who should be informed before sharing.
- DR is the *data requester* that wants to access the healthcare data of DS.
- DC is the *data controller* that hosts the healthcare data of DS and checks authorisation for data access requests.
- $R_{N,M}$ is a service request sent from entity N to M.
- $C_{N,M}$ is a *consent* file that is coupled with the request information of M and the consent information of N that is created by N.
- $SP_{N,M}$ is an authorised *sticky policy* file by entity N for M that includes data protection policies.
- $D_{N,M}$ is an exchanged healthcare data file sent from entity N to M.
- $\mathrm{H}\{D_{N,M}\}$ is a digest of data $D_{N,M}$ using a one-way collision-resistant function (a "hash").
- $RE_{N,M}$ is a personal receipt of the transaction proof between entity N and M.

Architecture

Figure 5.11 shows the architecture of BRESPE. The DR represents an entity that requires access to the PHI of DS. DC stores and processes DS's PHI. The completed data flow is illustrated as follows: Firstly, DR sends a service request $R_{DR,DS}$ to DS to ask for medical data access. DS receives the service request $R_{DR,DS}$ and returns a user consent $C_{DS,DR}$. DC then pulls the consent $C_{DS,DR}$ and views the information of consent from DS. After that, DC runs a smart contract to generate a policy $SP_{DC,DR}$ for DR based on the consent $C_{DS,DR}$. Subsequently, DR can pull the consent $C_{DS,DR}$ and policy $SP_{DC,DR}$ to confirm authorisation of the service request and then stores them as proofs. DC packages the required PHI based on the policy $SP_{DC,DR}$ for DR. The data package also includes a digest of policy $SP_{DC,DR}$. Furthermore, DC pushes the digest $\mathrm{H}\{D_{DC,DR}\}$ of

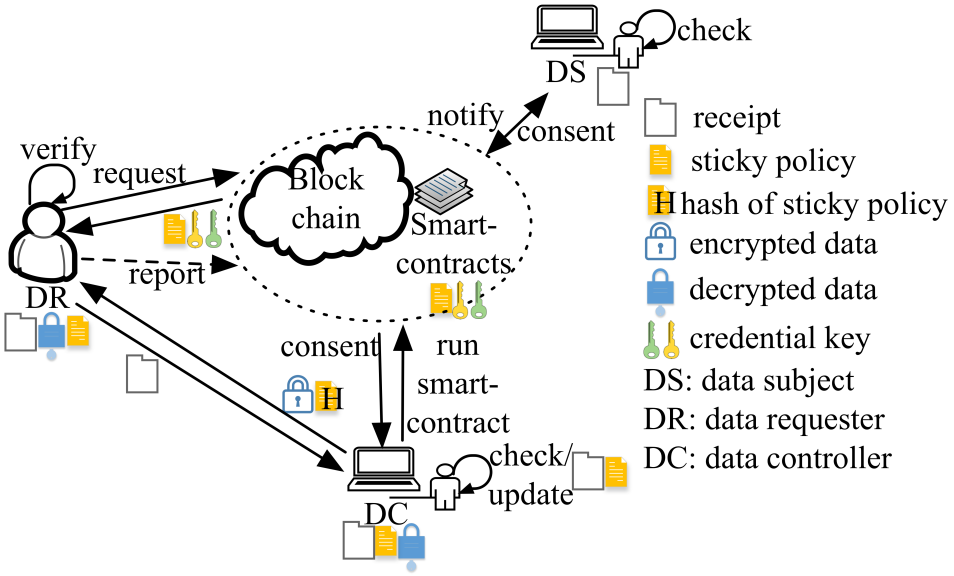


Figure 5.11: The system architecture of the BRESPE framework

the required healthcare data to the blockchain network as proof of data verification. DR receives the required data and pulls the related proofs. Finally, DR reports the verification results to the public through the blockchain if DR recognises errors. The whole data flow ends at this point. Throughout the data transaction progress, all exchanged information regarding permission and authorisation are first pushed to the blockchain network by senders and then pulled by recipients. These transactions are always mediated by the blockchain, except for the actual PHI in the final step. Also, they trigger smart contracts to process the data exchange. A receipt is always generated following the transaction of new information, such as a service request generation and push.

Consent Management

User consent management is a crucial component of data privacy protection in BRESPE's healthcare data connectivity. It is used in two ways, depending on its source and purpose.

The first type of user consent comes from the sender of the data. It is required to sign the data before sharing. This signature confirms the integrity of the data and identifies the original source of the data. For example, when DR sends a service request $R_{DR,DS}$ to DS, DS responds with a receipt $RE_{DR,DS}$.

The second type of user consent comes from the recipient in the data flow. The recipient gives its consent to respond to the request of the sender. In BRESPE, DS and DC are the primary recipients of consent requests from DR. For example, DS can give a consent file $C_{DS,DR}$ in response to DR's request $R_{DR,DS}$. DC can also construct a sticky

policy $SP_{DC,DR}$ to respond to $C_{DS,DR}$ from the DS.

When data files are shared, receipts are generated as proof of evidence to record the data transaction. These receipts are important for auditing purposes, as entities can review their actions after the fact and ensure that they comply with applicable data privacy regulations.

Receipt Management

To build PDRs in the data transaction to audit entities' actions after the fact and assure non-repudiation of actions in BRESPE. PDRs build an audit trace trail as proof of the data transaction. When an entity shares new information with other entities, a receipt is generated to record this transaction and can be downloaded by the sender. The signature of the sender on the receipt ensures the authenticity and integrity of the receipt. There are three kinds of these receipts:

- Receipt for a service request. When DR starts a service request to access the PHI of DS, a receipt $RE_{DR,DS}$ is generated that includes the transaction metadata and a hash digest of this request. DS signs this receipt to confirm receipt of the service request.

- Receipt for user consent or sticky policy. When DS consents to the request from DR, or DC generates a sticky policy and then pushes it onto the blockchain, a receipt is generated and returned to DS or DC as proof of this transaction. The receipt includes the content of this transaction and its hash value.

- Receipt for the shared PHI between DC and DR. When DC shares actual healthcare data with DR, DC needs to push a hash value of the exchanged data to the blockchain. This record is saved to verify the transaction after the fact of the data exchange. DC receives a transaction receipt $RE_{DC,DR}$ to record the data exchange that includes only a digest of the transaction.

Data Format

The proposed framework uses different data formats in the BRESPE protocol that includes a service request, user consent, sticky policy, and shared data package. All in-

volved data files in the BRESPE protocol are encrypted by confidential keys to ensure the confidentiality and privacy of the shared data.

- A request $R_{N,M}$. It is a data file that contains information about the data access request made by DR to DS. This includes unique individual identity information of DR and DS, the expected start and expiration date of data access, purpose, data type, and a signature of DR.

- A user consent $C_{N,M}$. DS creates user consent to allow DR to access the requested data. This consent contains unique identities of DR and DS, the authorised start and expiration date of data access, purpose, data type, status, an anchor linked to DC, and a signature of DS.

- A sticky policy $SP_{N,M}$. DC uses smart contracts to construct sticky policies to grant permission for data access to DR. This record includes identity information of DR, DS, and DC, the agreed start and expiration date of data access, purpose, data type, status, and a signature of DC.

- An exchanged data package $D_{N,M}$. Entity DC shares encrypted required PHI with DR. The exchanged data package has an anchor of $SP_{DC,DR}$ and a digest of the required PHI.

Protocol

BRESPE establishes a protocol to meet security requirements. This protocol has four phases (*initialisation*, *service request*, *consent and sticky policy construction*, and *resource exchange*), which govern data processing between entities within the data flow.

Initialisation It is the first phase of the BRESPE protocol, which involves preparation work for data sharing by performing key generation and entity registration. The initialisation phase is designed similarly to the framework described in section 5.3.

- Registration. This process requires all involved entities in the data flow to register in the blockchain network and initialise their accounts according to the established rules for data exchange in the communication channel.

- Key generation. BRESPE uses the Diffie-Hellman key exchange scheme to generate secret keys and securely exchange cryptographic keys over a public blockchain. During key generation, an entity shares a base and modulus with another entity, and then both entities respectively choose a secret random integer to generate a secret key for data encryption. In the peer-to-peer communication of BRESPE, each entity has its own key pair. If an entity loses its secret key, the data flow is required to end at this point until the entity can agree to generate a new key pair with its recipient to share data. Once the new key pair is generated, the lost key pair becomes obsolete. That does not expose personal credential information because all exchanged data in the blockchain are permission-based. Entities are not permitted to exchange the actual PHI in the communication channel until permission is granted.

Figure 5.12 highlights the key components of data flow, including the phases of service request, consent and sticky policy construction, and resource exchange. The following notations are used in this sequence chart.

- $D_1 \| D_2$ denotes the concatenation of data files D_1 and D_2.
- $k_{N,M}$ is an agreed secret key of entities N and M in the data flow following the Diffie-Hellman key exchange method.
- $\text{sig}_N\{D_{N,M}\}$ is a data file $D_{N,M}$ signed by entity N.
- $\text{sig}_N\{SP_{N,M}\}$ is a sticky policy file $SP_{N,M}$ signed by entity N.
- $\{D_{N,M}\}_{k_{N,M}}$ is a data file $D_{N,M}$ encrypted under the key $k_{N,M}$.

The protocol of BRUE uses the following data formats:

- request $R_{N,M}$

$$N \rightarrow M : \{\text{sig}_N\{R_{N,M}\}\}_{k_{N,M}}$$

Entity N sends a request file $R_{N,M}$ to entity M as a start of the whole data flow. This request $R_{N,M}$ is signed by the sender to assure auditability and encrypted with their exchanged secret key $k_{N,M}$ for confidentiality.

- consent $C_{N,M}$

$$N \rightarrow M : \{\text{sig}_N\{C_{N,M}\}\}_{k_{N,M}}$$

Entity N responds to the request $R_{M,N}$ with a consent file $C_{N,M}$ to M. N signs and encrypts the message.

- sticky policy $SP_{N,M}$

$$N \rightarrow M : \{\text{sig}_N\{SP_{N,M}\}\}_{k_{N,M}}$$

Entity N sends a sticky policy file $SP_{N,M}$ to M as a response to the request $R_{M,N}$ after receiving a consent $C_{N,M}$ and, to assure auditability, it signs the message. For confidentiality, N encrypts this file with their exchanged secret key $k_{N,M}$.

- exchanged healthcare data $D_{N,M}$

$$N \rightarrow M : \{\text{sig}_N\{D_{N,M}\}\}_{k_{N,M}}$$

Entity N sends a data file $D_{N,M}$ to M response for the request of data resource access. $D_{N,M}$ is signed by N and encrypted with their shared secret key $k_{N,M}$.

- receipt $RE_{N,M}$

$$RE_{N,M} = \text{sig}_N\{R_{N,M}\} \| \text{H}\{R_{N,M}\}$$

$$RE_{N,M} = \text{sig}_N\{SP_{N,M}\} \| \text{H}\{SP_{N,M}\}$$

$$RE_{N,M} = \text{sig}_N\{C_{N,M}\} \| \text{H}\{C_{N,M}\}$$

$$RE_{N,M} = \text{sig}_N\{\text{H}\{D_{N,M}\}\}$$

Once entity N shares information with M through the blockchain network, a transaction receipt $RE_{N,M}$ is generated, which includes the transaction metadata and a digest, except for the fourth receipt. The fourth receipt is a proof record that includes only a signature and the data transaction without an extra digest.

Service Request This phase is responsible for initiating data transmission between entities. In this phase, DR initiates the data flow by sending a request $R_{DR,DS}$ to DS through the blockchain network. A receipt $RE_{DR,DS}$ is then generated for DR as proof of integrity. DS receives this request $R_{DR,DS}$ and checks the information. The purpose of these actions is to enable DR to request permission to access healthcare data from the DS in a secure and controlled manner.

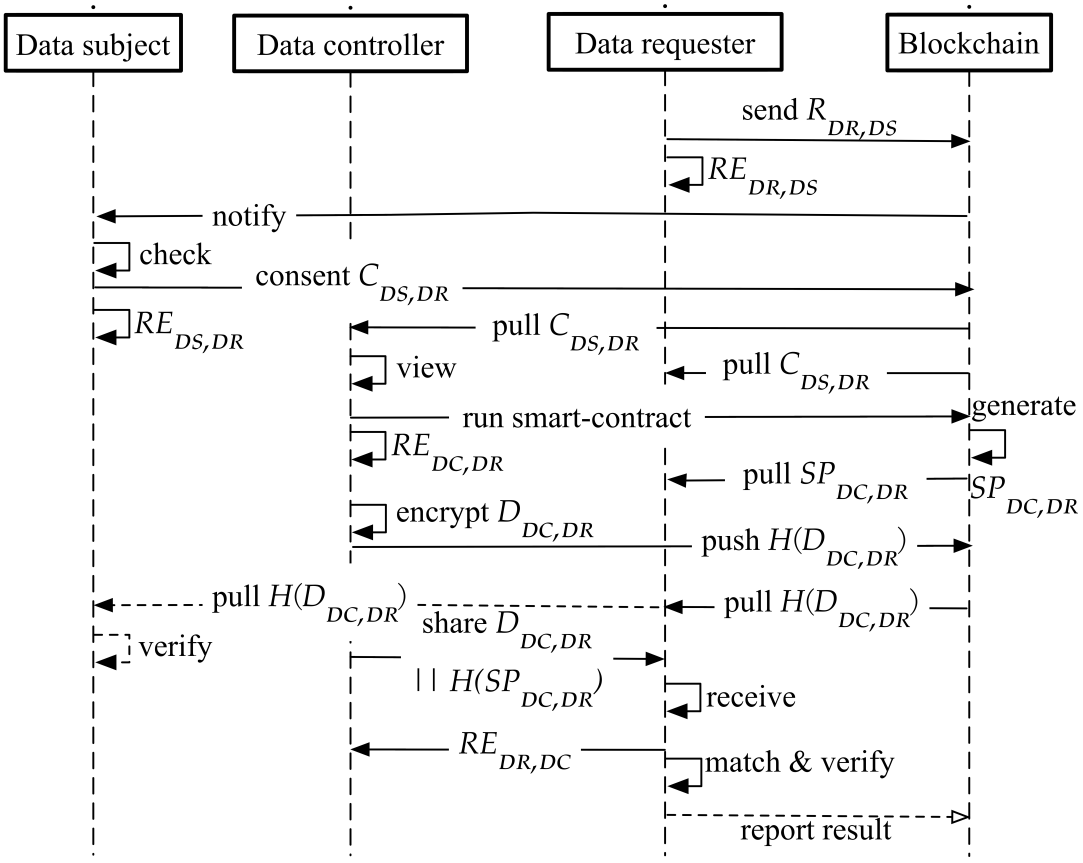


Figure 5.12: The message sequence chart of the BRESPE framework

Consent and Sticky Policy Construction DS and DC play crucial roles in this phase to verify the received information and process the permission for DR. Policies are also constructed during this phase to govern the exchange of data. Once the data transmission is completed, a receipt is generated and returned to the message sender. All receipts are designed to create a complete trace trail of transactions. As an important phase, entities process the service request and obtain consent for the action of data access by the requester. Therefore, DC builds sticky policies based on the user preferences of DS to support DS having rights to authenticate data transactions.

- Consent. When DS receives a service request $R_{DR,DS}$ from DR, DS has the option to either fully consent to the request directly or modify the request authorisation. In the permission process, DS executes smart contracts to generate a confirmed consent $C_{DS,DR}$ in response to the DR's request. This consent $C_{DS,DR}$ reflects DS's preference for data sharing and signifies the DR's authorisation to access the requested PHI. Also, a receipt $RE_{DS,DR}$ returns to DS via the blockchain to confirm the successful completion of the transaction.

- Sticky policy. DR and DC receive the consent $C_{DS,DR}$ from DS through the

blockchain after DS authorises the request. Then, DC runs a smart contract to create a policy $SP_{DC,DR}$ that supports the transaction for DR based on the preference of DS outlined in the consent $C_{DS,DR}$. This policy $SP_{DC,DR}$ describes the authorised data access rules for DR and is designed to comply with data protection regulations. A receipt $RE_{DC,DR}$ is issued to DC following the successful transaction to confirm the integrity of the process. DR receives $SP_{DC,DR}$ by the blockchain and stores it for future reference.

Resource Exchange In the final phase, DC and DR exchange the required PHI. However, this phase may fail if DC does not get the necessary authorisation (a user consent $C_{DS,DC}$) from DS or if DR cannot receive a sticky policy from DC. We assume that the data transaction is completed successfully, and DR receives the policy $SP_{DC,DR}$. DC encrypts the required PHI using the agreed secret key $k_{DC,DR}$ and then signs the data. The DC then generates a hash digest of the transaction for the required exchanged data as integrity proof. DC signs this digest $\mathrm{H}\{D_{DC,DR}\}$ and pushes it to the blockchain for record verification. DC shares the required PHI $D_{DC,DR}$ with DR that includes metadata and a digest of the related sticky policy $\mathrm{H}\{SP_{DC,DR}\}$. This exchanged data package is represented by $\mathrm{sig}_N\{D_{N,M}\}\}_{k_{N,M}}||\mathrm{H}\{SP_{DC,DR}\}$.

DR receives the exchanged data and pulls the digest $\mathrm{H}\{D_{DC,DR}\}$ of the transaction from the blockchain for verification. If the data verification fails, DR reports the result to the blockchain to notify other entities. A receipt $RE_{DC,DR}$ is sent to DC after the required PHI sharing is completed. DS can also pull the digest $\mathrm{H}\{D_{DC,DR}\}$ from the blockchain as proof to verify if DC shared data with DR as agreed. The entire data flow ends at this point.

5.5 Discussion

AudiWFlow is a framework that addresses the challenge of creating an audit trail for entities' interactions in a distributed workflow involving different entities who may not be familiar with each other, while ensuring confidentiality. This framework provides a means to audit for misbehaviour without violating confidentiality. In the context of exchanging data including PHI, the AudiWFlow framework can be used in the case of data exchange

about different entities involved in a single jurisdiction. The BRUE framework has been proposed to securely exchange PHI between entities across different jurisdictions. The BRUE framework is designed to provide user authorisation management with confidentiality during the cross-jurisdiction exchange of PHI. Similarly, BRESPE is a framework to support PHI exchange that focuses on privacy-preserving data transactions in peer-to-peer communication. It addresses the challenge of constructing user consent and policies based on user preferences and data protection regulations to guide the exchange of PHI. The following discusses all three frameworks respectively by revisiting the main requirements: integrity, auditability, non-reputation, confidentiality, collusion detection, compatibility, and key management. Besides, we compare the three proposed frameworks with four selected existing blockchain-based frameworks in terms of security requirements.

5.5.1 Integrity, Auditability, and Non-Repudiation

The AudiWFlow framework is designed to detect any attempts of data tampering or dissemination of incorrect audit records by dishonest entities in a distributed workflow. Each individual audit record (A_{ij}) is immediately verified after generation, and each entity maintains receipts (R_{ij}) of locally exchanged audit records. As long as two adjacent entities do not collude and the key distribution is secure, this mechanism ensures non-repudiation. Additionally, all entities perform the audit record verification after receiving the exchanged data, which further checks for data corruption or concealment at various stages, including data exchange between pairs of entities in the workflow and audit records sent to the blockchain. The running of smart contracts is publicly auditable without disclosing sensitive information about the workflow in AudiWFlow.

In the case of the BRUE and BRESPE frameworks, workflow entities are not allowed to exchange the actual PHI in the workflow until the authorisation process is completed. To satisfy the requirements, both the BRUE and BRESPE frameworks are designed in three ways: (i) All data flows are required to exchange through smart contracts running on a public blockchain. There is no direct interaction between entities without the blockchain involved. BRUE uses tokens to authorise actions across jurisdictions, while BRESPE authorises data transactions based on policies and user consent, which are determined by user preferences and data protection regulations. The blocks in the blockchain

record the trail of data transactions in the authorisation process, which ensures auditing and non-repudiation of actions after the fact. (ii) Transaction records require cryptographic signatures from entities, ensuring the authenticity, integrity, and auditability of transactions. (iii) A personal receipt ($RE_{N,M}$) is produced as evidence proof to prevent intentional actions of entities after the fact. This receipt includes the metadata of the current transaction and its hash digest, which is returned to entities after they share information.

5.5.2 Confidentiality and Privacy Preservation

The AudiWFlow, BRUE, and BRESPE frameworks all share the goal of constructing an audit trail to facilitate data auditing while ensuring the confidentiality of the exchanged data. In AudiWFlow, the audit trail is kept by all entities in a secure manner, using encryption with the workflow key, ensuring the privacy of each entity's participation. If needed, the protected audit trail can be read in case of a dispute and a pre-defined (configurable) K number of entities agreeing. The audit trail is available since all entities keep the whole audit trail. Note that the purpose of using a secret sharing scheme is to prevent the distribution of invalid key shares.

BRUE and BRESPE frameworks use personal receipts $RE_{N,M}$ to construct the audit trail. The personal receipts, which contain the metadata of the current transaction and its hash digest, are compared with the records in the blockchain to audit the actions of data processing in real time and after the fact. The audit records in the blockchain are permanent and cannot be tampered with. The data exchanged in both frameworks are protected using encryption. The Diffie-Hellman key exchange method is used to generate a secret shared key for data encryption. If the secret key is lost or disclosed, the data flow is interrupted until two associated pair entities agree to generate a new key pair. BRUE uses protected tokens to exchange information between entities, instead of the actual personal health data, to prevent data disclosure, while BRESPE shares data in ciphertext on a public transmission channel and only authorises relevant entities with the related secret key to access the data. Both frameworks are designed to protect personal credential data and ensure data privacy.

5.5.3 Collusion Detection

AudiWFlow focuses on ensuring that entities in a workflow can share data with others while maintaining an audit trail of their actions. However, this scheme may be vulnerable to collusion attacks if the entities are dishonest. For instance, in a workflow involving entities A, B, C, and D. Consider this workflow: $A \rightarrow B \rightarrow C \rightarrow D$. Entities B and C could collude to forge, modify, or destroy their internal audit records since no one else can verify or attest to their integrity. While proof of collusion would exist, it is challenging to prevent such an attack between adjacent entities. To mitigate this risk, AudiWFlow proposes a solution where each entity links the previous audit records to its own records, making collusion less likely in small workflows. However, this approach requires that the audit records generated be open for verification and inspection among the colluding entities if required. In the case of larger workflows, the trail of audit records could propagate across the entire topology. Additionally, the audit server in AudiWFlow is implemented as a blockchain, which can permanently store audit records for verification without tampered with to prevent collusion attacks involving the audit server.

In contrast, BRUE and BRESPE prevent collusion attacks by requiring entities to share authorisation data with specified consent only using the blockchain as the communication channel. The roles of related entities in BRUE and BRESPE are the data requester, data subject, and data controller (authorisation server and resource server), who have direct responsibility for data security in the data process based on regulations and standards. While there is no specific design in the BRUE and BRESPE frameworks to prevent collusion, all data transactions of entities are recorded by the blockchain, making it difficult for colluding entities to tamper with the records.

5.5.4 Compatibility and Availability

The AudiWFlow framework presents an attractive solution for data exchange between different entities in a workflow. It employs a blockchain with added verification logic to serve as a simple audit server. The role of the audit server is to process integrity-proof records ($I_{N,M}$) between entities. While the audit records must be securely stored outside the blockchain, they have no impact on the data processing itself. AudiWFlow is flexible to accommodate any workflow topology with an unlimited number of entities.

In comparison, the BRUE framework reuses the concepts of the UMA method with personal data receipts that run on a public blockchain. The protocol of it is designed to comply with data sharing protection regulations and standards. It provides a solution for cross-jurisdiction PHI exchange without the need for a well-known discovering point (Rendez-vous points). This jurisdiction can be a country, a specified federation of some entities, or a city. The jurisdiction scope can be defined by the data subject or resource owner. The protocol supports multiple RSs involved in data sharing. Entities can require access to PHI from the RS located in different or the same jurisdiction. The system compatibility is robust in terms of the number of entities involved and access control between cross-jurisdiction entities.

In contrast, the BRESPE framework prioritises privacy-preserving PHI exchange based on user preference and data protection regulations. The protocol requires entities to use blockchains to process data transactions and minimises direct peer-to-peer communication without the need for a mediator. It emphasises point-to-point communication without the involvement of third parties and ignores regional boundaries between entities. Compared to AudiWFlow, both BRUE and BRESPE are lightweight and reliable frameworks that use blockchain to process authorisation between entities compatible with the existing infrastructure.

5.5.5 Key Management

Suppose that involved entities are honest to apply trustworthy methods to safeguard their credentials, such as those in the key cryptography exchange. In AudiWFlow, Shamir's secret sharing scheme is followed for key distribution of the shares (k_i) of $sk_{\mathcal{W}}$. Through this scheme, entities can verify the correctness of key generation and distribution to ensure that each entity possesses the correct share of the related secret key. However, there remains a concern that $sk_{\mathcal{W}}$ may be exposed by the participating entity generating the key. To minimise the risk of breaching the confidentiality of audit records, workflow entities with the least incentive to cheat are responsible for key generation. In this case, the entity without possession of the audit trail but with an interest in its success, such as the entity requesting the workflow, is responsible for key generation.

BRUE and BRESPE utilise the Diffie-Hellman key exchange mechanism to produce

a secret key pair between entities through a blockchain. If an entity is removed from the workflow, the secret key is changed and updated as well. For instance, when entity A intends to send a service request to entity B, A and B must first agree upon a secret key pair ($K_{A,B}$ and $K_{B,A}$). Although the secret key is at risk of disclosure, the data transaction encrypted with this key is recorded on the blockchain. If the secret key is lost or disclosed, the data flow is immediately terminated, and A and B can agree upon a new key pair to restart the data flow.

5.5.6 Comparative Analysis of Framework Design

The above subsections discussed the security requirements of the three proposed frameworks in terms of data integrity, auditability, confidentiality, data privacy, and system compatibility. As we mentioned in chapter 4, the literature listed some existing blockchain-based frameworks for PHI exchange. We compare three proposed frameworks with four different types of typical existing frameworks (Azaria et al. 2016, Zhang, White, Schmidt, Lenz & Rosenbloom 2018, Mohey Eldin et al. 2023, Lee et al. 2022) in terms of a summary of security requirements, entities involved, the scope of data exchange, and data stored in the blockchain, as shown in Table 5.1. The implementations evaluation comparison are presented in the chapter 6.

Table 5.1: Comparative analysis of framework design

Frameworks (models, schemes)	Data stored in blockchain	Scope of data exchange	Entities	Summary
MedRec (Azaria et al. 2016)	Metadata of permission and EMR ownership	Multiple healthcare providers	Patient, healthcare provider	It is a distributed EMR management system to handle the access of EMRs between medical jurisdictions using Ethereum, ensuring authentication, confidentiality, accountability, and regulatory compliance but it does not mention addressing the limitations of system scalability and the security in databases.
FHIRchain (Zhang, White, Schmidt, Lenz & Rosenbloom 2018)	Access token, transaction log, reference pointers, hash value of exchanged data	Cross-regional healthcare organisations	Patient, medical specialist, clinician, database	It is an Ethereum-based architecture to share clinical data using digital health identity, ensuring data integrity, security, scalability, and regulatory compliance with the FHIR standards.

Frameworks (models, schemes)	Data stored in blockchain	Scope of data exchange	Entities	Summary
FBS (Mohey Eldin et al. 2023)	Hash value of EHR	Cross-regional healthcare organisations	Patient, authority organisation, cloud provider, hospital, IoT device	It uses permissioned blockchains to process EHR, ensuring security, interoperability, data integrity, authentication, and access control but it only shares EHR between healthcare organisations and may not be compatible with systems not supporting FHIR.
PIE (Lee et al. 2022)	Transaction record of uploading EMR	Multiple medical organisations	Patient, IPFS, physician, certificate authority	It is an EMR-sharing system that uses a consortium blockchain and an IPFS to ensure the system throughput, latency, privacy preservation, confidentiality, data integrity, and scalability, however, it involves the CA and does not mention cross-sector data exchange outside of health systems.

Frameworks (models, schemes)	Data stored in blockchain	Scope of data exchange	Entities	Summary
AudiWFlow	Hash value of exchanged health data	A single jurisdiction	Not mention	It utilises a workflow and a public blockchain to directly exchange PHI between entities, ensuring confidentiality, auditability, data integrity, and system compatibility but does not describe user authorisation management and regulatory compliance.
BRUE	Permission information	Multiple jurisdictions	Patient, resource owner, authorisation organisation, hospital, data requester	It provides token-based authorisation management and access control using a public blockchain, ensuring confidentiality, auditability, data integrity, system compatibility, and data privacy but does not detail the actual PHI exchange between entities.
BRESPE	Permission information, the hash value of policies	Not mention	Data subject, data controller, data requester	It provides user consent management and improves privacy preservation of PHI exchange by a public blockchain, ensuring confidentiality, auditability, data integrity, and system compatibility but does not detail the actual PHI exchange between entities.

In the above table, the selected four existing frameworks (MedRec, FHIRchain, FBS, and PIE) are designed to separately share EMR, EHR, or clinical data. AudiWFlow, BRUE and BRESPE are designed to share general health information. The design of three proposed frameworks does not emphasise the exchange of specific health data among workflow participants as the four comparative frameworks do. All involved frameworks choose different types of data stored within the blockchain network to achieve the design requirements. Not all designs of frameworks meets the cross-jurisdiction or cross-organisation data exchange. AudiWFlow framework is only one to share data within a single domain. There is a big difference between the workflow participants in these frameworks. Compared to four comparative frameworks, the design of BRUE and BRESPE highlights the roles and scopes of the involved participants in the data workflow. Each role defines different operational scopes related to data transmission.

Chapter 6

Prototype Implementation and Evaluation

To achieve objective 4, this chapter presents the prototype implementations of three proposed frameworks mentioned in the chapter 5 and conducts a quantitative metrics research method to evaluate these proof-of-concept systems in performance. These proof-of-concept systems demonstrate the feasibility and efficiency of blockchain-based solutions in the healthcare data exchange. The prototype system of AudiWFlow implements the workflow and an auditable server and the proof-of-concept systems of BRUE and BRESPE individually implement a web application. The chapter ends by a performance discussion of three proposed frameworks implementations and a comparative analysis among prototype systems of proposed frameworks and selected four existing frameworks mentioned in the section 5.5 of the previous chapter.

6.1 Introduction

The implementation of prototype systems about AudiWFlow, BRUE, and BRESPE frameworks applied public blockchains, smart contracts, PDRs, and sticky policies. Three frameworks were built on Ethereum, a permissionless blockchain network that established a reliable privacy-preserving framework without the need for central authorities. Although Ethereum is currently one of the most popular blockchain platforms, there are no specific expectations or recommendations as to which platform to use. As long as the blockchain can store a value, such as the hash of the records, it can be used.

The adoption of a permissionless blockchain network in these frameworks provides various security benefits. The smart contracts for the proposed frameworks were deployed on Ethereum using a local network under the development model. The local network employed Ganache-CLI [1] to run as blockchain nodes. Ganache-CLI is the command line version of Ganache [2]. To ensure the universality and accuracy of experimental results, both the BRUE and BRESPE demos were also tested on the Goerli [3] network, a public testnet of Ethereum with the proof-of-stake consensus mechanism. Table 6.1 shows the experimental configuration environment for the demos presented in this chapter.

Before the *London Upgrade* [4], blockchain miners would receive the total gas fee from any transaction included in a block. In the local test network, the transaction fee was calculated based on the gas used and gas price according to the rule prior to the *London Upgrade*. It is represented by the formula: $transactionfee = gasused \times gaspriceperunit$. In the experiment conducted on the Goerli test network, the transaction fee was computed based on the rule after the *London Upgrade*. It is represented by the formula: $gasused \times (basefee + priorityfee)$.

Table 6.1: Experimental configuration environment of demos

	AudiWFlow	BRUE	BRESPE
Network	Local private network	Local network, Goerli	Local network, Goerli
Node and Connector	Geth	Ganache-cli, Remix injected provider	Ganache-cli, Remix injected provider
Hardware	Intel Core i7-6700HQ CPU with 32GB RAM on Windows 10	Intel Core i7 at 2.9GHz with 8GB RAM on macOS High Sierra	Intel Core i7 at 2.9GHz with 8GB RAM on macOS High Sierra

Smart contracts are self-executing codes that encode transactional rules on a network, which are installed and instantiated by authorised entities on channel peers. In this chapter, the discussed prototypes employ smart contracts to support the application logic of the system for data request handling, data transmission, and consent management. These smart contracts facilitate the processing of requesting access to PHI, granting access

[1]Ganache CLI: `https://www.npmjs.com/package/ganache-cli`
[2]Ganache: `https://github.com/trufflesuite/ganache`
[3]Goerli: `https://goerli.net/`
[4]London Upgrade: `https://ethereum.org/en/history/#london`

permissions, and updating permissions. The code of all smart contracts implemented in these prototypes is written in Solidity [5].

Personal Data Receipts are an effective tool to improve the completeness of an evidence trail. They are digital storable artefacts that both users and services can store to provide evidence that satisfies non-repudiation of who, what, and how was agreed upon. BRUE and BRESPE use PDRs to support consent management and establish traceability by constructing an evidence trail of data transactions. PDRs store information about transactions and their hash values, and they are downloaded and saved as JSON documents in local storage in the proposed models.

Sticky policy is a mature technology that complies with data protection regulations and is an effective solution for data sharing. Sticky policies are formulated as JSON files and stuck with exchanged data during transmission, providing data privacy protection at a certain static level or a per-user and per-data level. BRESPE employs smart contracts to build sticky policies that maintain privacy and security at a per-user and per-data level. It provides a user interface (web form) for the data controller to formulate policies based on user preferences and data protection regulations.

This chapter showcases the prototype implementations of three frameworks that were not primarily designed for performance evaluation but rather to demonstrate the feasibility and completeness of the proposed frameworks. Section 6.2 presents the prototype implementation and performance evaluation of AudiWFlow. Section 6.3 discusses the prototype implementation and performance evaluation of BRUE. Section 6.4 elaborates on the prototype implementation and performance evaluation of BRESPE. Section 6.5 provides a discussion of system implementation and performance about three proposed frameworks. The last section takes a research method of comparative analysis between selected existing frameworks and proposed frameworks.

6.2 AudiWFlow

The AudiWFlow framework is an auditing approach for distributed workflows that prioritises confidentiality-friendly and collusion-resistant. Its implementation involves the use of a public blockchain with smart contracts in place of a third party for audit trail records

[5]Solidity: `https://docs.soliditylang.org/en/v0.8.17/`

and sharing. The implementation is mainly written in the programming languages JAVA for the workflow and Solidity for smart contracts, with all the codes being open source [6]. Experimental results obtained from the implementation are also publicly available [7]. This section provides details of the AudiWFlow framework implementation and then evaluates its performance in terms of scalability, encompassing gas cost and processing time.

6.2.1 Implementation

The implementation of AudiWFlow lacks a user interface to view transactions. It involves two main roles, namely, workflow entities and the audit server (blockchain). The implementation is developed through several stages, including workflow construction, data exchange between entities, audit trail construction, and records verification. To generate key pairs, Shamir's Secret Sharing [8] scheme is used to generate shares of the workflow key. The Nimbus-JOSE library [9] is used to process cryptography between workflow entities, facilitating cryptographic functions that include the encryption of audit records and the generation and verification of integrity proofs. As the proposed framework mainly employs smart contracts mainly for audit log transactions and key management, the following information outlines the implementation details of these components. For additional information, please refer to (Zhou et al. 2022).

Audit Trail Construction and Record Verification

The AudiWFlow framework involves the generation of audit logs by workflow entities, which are then pushed to the blockchain for storage and record verification. The implementation utilises the following smart contract codes for record verification and audit trail construction. The function *saveLog* is responsible for storing audit logs into the blockchain along with transaction information. Entities can access historical records of audit proofs from the blockchain using the function *getLogByOwner*. The function *compareLogs* provides record verification for entities. These functions are used to store logs, verify the integrity of audit logs, and ensure that the workflow has been executed as expected.

[6] AudiWFlow demo codes: `https://github.com/antonionehme/AuditingWorkflows-Blockchain`

[7] Relevant experimental data: `https://github.com/Jency/AudiWFlow.git`

[8] Shamir Secret Sharing Scheme: `https://github.com/iancoleman/shamir/blob/master/src/js/secrets.js`

[9] Nimbus JOSE: `https://connect2id.com/products/nimbus-jose-jwt`

```
contract LogFactory {
    using SafeMath for uint256;
    event NewLog (uint logId, string signature, string
       hashOfMessage);
    struct Log {string signature;string hashOfMessage;}
    Log[] public logs;
    mapping (uint => address) public logToOwner;
    mapping (address => string) public signToMessage;
    mapping (address => uint) ownerLogCount;
    function saveLog(string memory _signature,string memory
       _encryptedMessage) public payable returns(bool) {
        uint id=logs.push(Log(_signature,_encryptedMessage))-1;
        logToOwner[id] = msg.sender;
        signToMessage[msg.sender] = _signature;
        ownerLogCount[msg.sender]=ownerLogCount[msg.sender].add
           (1);
        emit NewLog(id, _signature, _encryptedMessage);
        return true; }
    function getLogIdByOwner(address _owner) public view returns
       (uint[] memory) {
        uint[] memory result = new uint[](ownerLogCount[_owner])
           ;
        uint counter = 0;
        for (uint i = 0; i < logs.length; i++) {
            if (logToOwner[i] == _owner) {
                result[counter] = i;
                counter = counter.add(1); }
        } return result; }
    function compareLogs(string memory _signature,string memory
       _payload,address _owner) public view returns(bool) {
        if (keccak256(bytes(signToMessage[_owner])) == keccak256
```

```
            (bytes(_signature))) {
             uint[] memory result = getLogIdByOwner(_owner);
             uint id = 0;
             for (uint i=0; i< result.length; i++) {
                 id = result[i];
                 Log storage preLog = logs[id];
                 while(keccak256(bytes(preLog.hashOfMessage)) ==
                    keccak256(bytes(_payload))) {
                     return true;
                 } } return false;
         } else {
             return false; } }
}
```

Key Distribution

In the AudiWFlow framework, entities share their public keys, including workflow and personal public keys, with other entities to encrypt exchanged data. The shared public keys are managed through three main functions running on a smart contract. The *saveKey* function stores the public keys of entities along with the key name, type, and signature of the owner. The *getKeyIdByOwner* function returns all records of keys for the key owner. The *getKey* function aims to return a specified key for the requester by specified properties. The following is a snapshot of the implementation of these functions in Solidity code:

```
contract KeyFactory {
  using SafeMath for uint256;
  event NewKey(uint keyId,string keyChain,string signature,
     string keyName);
  struct Key{string signature;string keyChain;string keyName;
     uint8 keyType;}
  Key[] public keys;
  mapping (uint => address) public keyToOwner;
```

```
mapping (address => string) public signToKey;
mapping (address => uint) ownerKeyCount;
mapping(uint => string) public idToName;
function saveKey(string memory _signature,string memory
    _keyChain,string memory _keyName,uint8 _keyType)public
    payable returns(bool) {
  uint id=keys.push(Key(_signature,_keyChain,_keyName,_keyType
      ))-1;
  keyToOwner[id]=msg.sender;
  signToKey[msg.sender] = _signature;
  idToName[id] = _keyName;
  ownerKeyCount[msg.sender] = ownerKeyCount[msg.sender].add(1)
      ;
  emit NewKey(id,_signature,_keyChain,_keyName);
  return true;  }
function getKeyIdByOwner(address _owner) public view returns(
    uint[] memory) {
  uint[] memory result = new uint[](ownerKeyCount[_owner]);
  uint counter = 0;
  for (uint i = 0; i < keys.length; i++) {
    if (keyToOwner[i] == _owner) {
      result[counter] = i;
      counter = counter.add(1); } // end if
  } return result; }
function getKey(string memory _signature,address _owner,string
    memory _keyName,uint8 _keyType) public view returns(string
    memory) {
  if (keccak256(bytes(signToKey[_owner]))==keccak256(bytes(
      _signature))){
      uint[] memory result = getKeyIdByOwner(_owner);
       uint id = 0;
```

```
        for (uint i=0; i< result.length; i++) {
          id = result[i];
          Key storage myKey = keys[id];
          while(keccak256(bytes(myKey.keyName))==keccak256(bytes
              (_keyName)) && myKey.keyType==_keyType) {
               return myKey.keyChain;   }
        } return ''this is a wrong key type or name'';
    } else { return ''this is a wrong account or signature'';
    } }
}
```

Functional Test

The AudiWFlow demo has been implemented successfully with JAVA codes running on a configured local Ethereum network. The prototype system has two main components. One is the workflow implementation that not associate with the blokchain network. Another is the blockchain implementation that involves the operations of integrity proof and public key. Table 6.2 lists the involved blockchain-enabled functions of demo and its test result.

Table 6.2: Results of the functional test of the AudiWFlow demo

Functions	pass the test?	use gas?
integrity proof generation and storage	yes	yes
integrity proof verification	yes	no
save a key	yes	yes
get a key	yes	no

According to the above table, the blockchain components implementation has four relevant functions that are all passed the test. This indicates that these blockchain components of the prototype were implemented successfully as expected. Although the implementation of workflow component was not shown in this section, it was also successfully completed as well.

6.2.2 Performance Evaluation

To evaluate the effectiveness of the AudiWFlow prototype system, we conducted experiments using BRITE [10], which generated various workflow topologies based on the Barabasi-Albert algorithm. This experiment used two key parameters: the first parameter was the number of workflow entities N, which was set to $N = 20$ for the AudiWFlow experiment. The second parameter was the connectivity degree m, which refers to the average number of entities that each workflow entity connects to. For example, a connectivity degree of $m = 1$ would result in a linear topology, while a degree of $m = N$ would result in a full mesh where all entities are connected to each other. For this experiment, m was set to $2, 3, 5, 7, 10$, and for each pair of parameters (N, m), we generated 5 different random topologies to ensure sufficient randomness and statistical accuracy. In total, the experiment included approximately 100 different topologies.

We deployed Ethereum on a Windows environment for the smart contract deployment, and the workflow entities were programmed using Java SE 8 and deployed on an Apache Tomcat application server. It was assumed that all workflow entities exchanged the same metadata with each other throughout the workflow. To evaluate the performance of the AudiWFlow model, we measured the response time for each transaction, which encompassed the message propagation time, the generation and reporting time of the audit record, and the time required for verification mechanisms, including cryptography.

```
{
  "config": {
    "chainId": 10,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "alloc": { },
  "coinbase": "0x0000000000000000000000000000000000000000",
  "difficulty": "0x1",
  "extraData": "",
  "gasLimit": "0x2fefd8",
  "nonce": "0x0000000000000042",
  "mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp": "0x00"
}
```

Figure 6.1: Structure of a genesis block for the AudiWFlow framework

Figure 6.1 depicts the configuration of the genesis block in the experiment network. To reduce block mining time, we set the difficulty variable to a low value. The following information records and analyses both the processing time and gas cost of the transac-

[10]BRITE: a network topology generator, `https://github.com/nsol-nmsu/brite-patch`

tions in the AudiWFlow model. Overall, the experiment was well-designed and carefully conducted to produce reliable and meaningful results.

Cost

Figure 6.2 shows the gas cost associated with records of different sizes (3 KB, 6 KB, and 10 KB) in a local private network. The average gas consumption (units) per iteration was recorded as 1134152.95, 1428271.94, and 1613575.64 for the respective record sizes, with a constant gas price per unit of 20 gwei. The results show a linear relationship between the record size and gas consumption, which is expected, as the AudiWFlow model stores minimal information (i.e., digests of the records) in the blockchain.

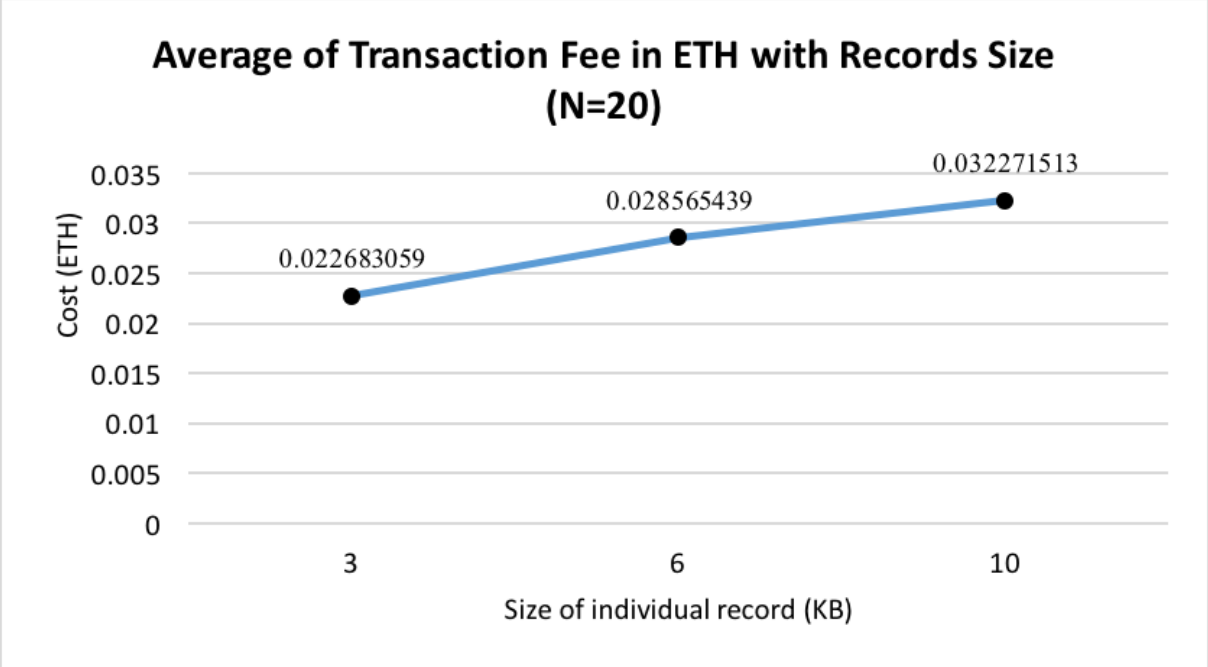


Figure 6.2: Average gas cost of each iteration with the size of records in ETH

As each workflow entity generates a single record, the gas consumption can be expressed as $G \propto N \cdot hash(M) + g_{alg} + g_{setup}$, where g_{alg} is the fixed gas consumed to execute the stored procedure (which is fixed and independent of the topology), and g_{setup} is any initial, one-time, setup of the smart contract. Using O-notation, the costs can be expressed as $C = O(hash(M) \cdot N \cdot C_{gas})$, where C_{gas} is the actual monetary cost per unit of gas, which varies depending on the platform (e.g., Ethereum) and market conditions. We assumed that the gas cost structure will be similar in any smart contract platform, including the current alternatives (e.g., Algorand [11]) as compared to Ethereum. It is essential

[11] Algorand: `https://www.algorand.com/`

to note that the gas cost structure may not be entirely accurate in this implementation, as the source codes of Ethereum used in this study date back to early 2021. It is also essential to note that Ethereum has recently changed how gas is calculated and spent, although the structure of costs remains the same as $C = O(hash(M) \cdot N \cdot C_{gas})$. A similar cost structure will exist even after the transition to Proof-of-Stake from Proof-of-Work.

Response time

Figure 6.3 presents the average response time of each iteration that is required to complete the data transaction. The key parameter in the experiment was the number of records generated and shared in each iteration. The experiment was conducted on a topology of $N = 20$ workflow entities, where each entity generated a 10 KB individual audit record. The relationship between the average response time and the number of records was found to be stable across different graph connectivity values. Specifically, the response time increased linearly with the number of audit records generated and shared.

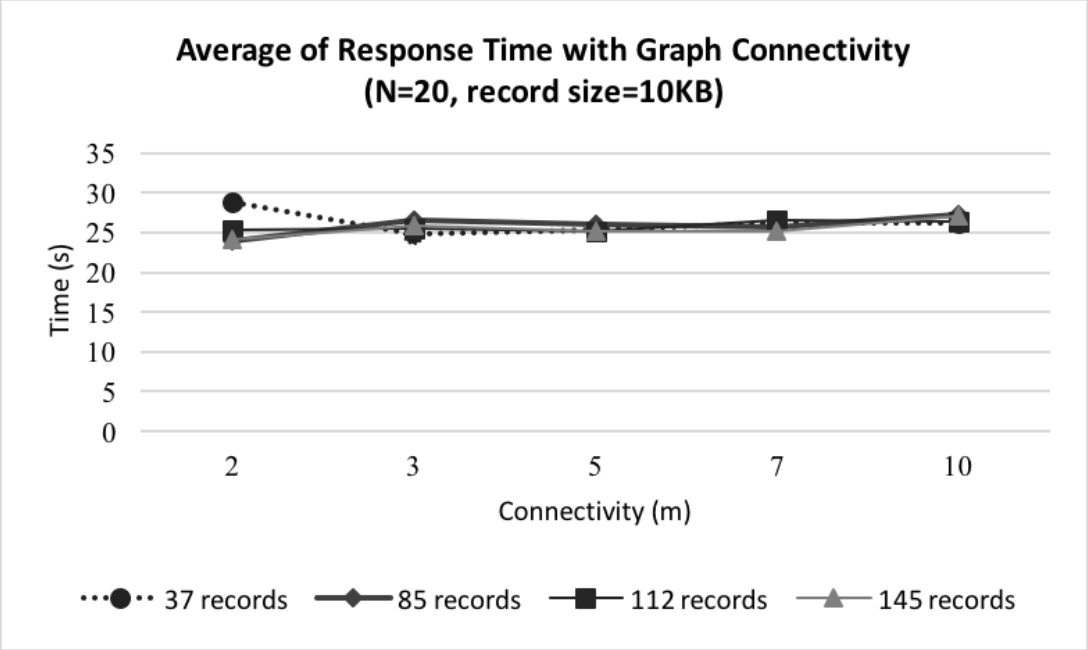


Figure 6.3: Average response time of each iteration with the number of records

6.3 BRUE

The BRUE framework aims to facilitate the cross-jurisdictional sharing of PHI through a public blockchain in a user-controlled and auditable manner. The implementation of

BRUE highlights the access permission process across jurisdictions in a simple and user-friendly manner that involves three key roles: the requesting party (RqP), the resource owner (RO), and the authorisation entity (which includes the authorisation server AS and resource server RS). It is important to note that in BRUE, RO is a delegate of the data subject. Both AS and RS play a similar role in the approach, which is to verify identities, authorise access, and share data with RqP. Therefore, the implementation of the demo in BRUE combines the operations of both roles. All of the demo implementation code is open and available to the public [12]. The implementation includes a user interface (web pages) to display data transactions between entities. The user interface was implemented using the JavaScript language, while the smart contracts were coded in Solidity. This section presents the implementation of the proof-of-concept system and evaluates its performance in terms of gas usage and response time.

6.3.1 Implementation

The implementation of BRUE was built with Truffle framework and webpack [13]. Function *FileSaver* [14] was also used to save documents from the user interface. Based on the BRUE protocol, there are several phases, including initialisation, service request, identity verification, authorisation verification, and resource exchange. Since the demo is concerned about the feasibility of the proposed model, the initialisation phase is not implemented. The service request phase involves data transactions between RqP and RO, while the identity verification and authorisation verification phases mainly include the operations of LAS and RAS. The resource exchange phase involves data exchange between RRS and RqP. The following sections provide an overview of the implementation details of the demo for each phase and give a result of the functional test of the demo.

Figure 6.4 illustrates the user interface structure in the implementation of the BRUE model. The web-based front-end interface provides a complete representation of the data flow within the demo. The directional arrows in the figure represent the direction of the data flow, while the integer labels indicate the sequence of the data flow.

In the client interface, RqP can initiate a data request to RS and receive a token PT from RS. The interface also allows RqP to receive a token from AS after successful

[12] BRUE demo codes: `https://github.com/Jency/BRUE.git`
[13] webpack: a static module bundler for modern JavaScript applications. `https://webpack.js.org/`
[14] FileSaver: `https://github.com/eligrey/FileSaver.js`

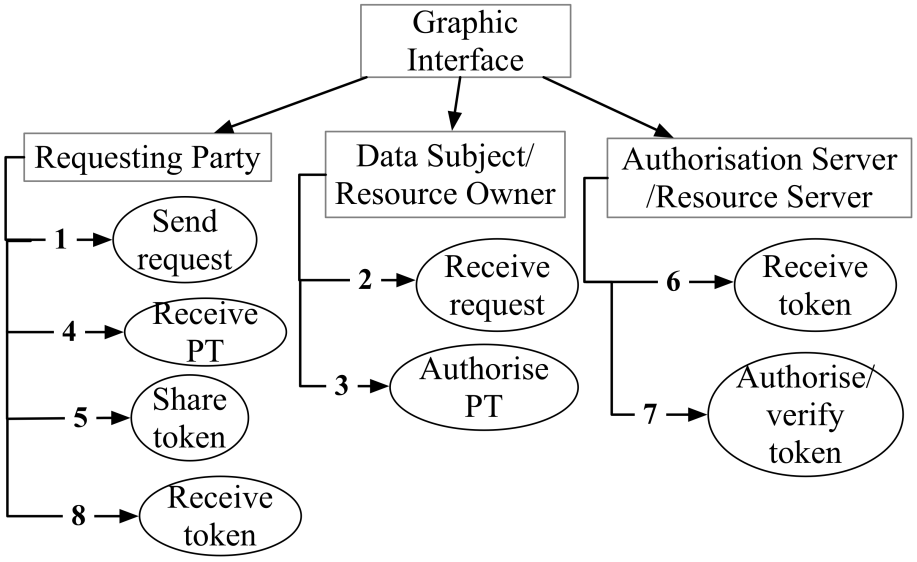


Figure 6.4: Structure of the proof-of-concept system implementation of BRUE

identity verification, and subsequently share the token with *RO*. On the other hand, *RO* (as a delegate of *DS*) receives requests from *RqP* and authorises access to the requested data. Similarly, *AS* receives the token from *RqP* and performs identity verification before authorising the token to *RqP*. *RS* verifies the token from *RqP* for permission information before sharing data with *RqP*.

Service Request

In this phase of the demo, the data transaction of the request service was implemented. *RqP* submits a request form to initiate a smart contract for information exchange using the blockchain network. Subsequently, *RO* views and authorises this request by pulling it from the blockchain. If authorised, a signed and encrypted token is generated and shared, and a receipt is downloaded as proof of evidence after the transaction.

The following code snippet shows the main functions of a smart contract for the data transaction of the request service. The *saveRequest* function saves the request information from *RqP*, while the *getRequest* function returns the request records for *RO*. The *authRequest* function implements the authorisation of the request and saves the request information to the blockchain.

```
contract RequestService {
  event NewRequest(string dataSubject,string requestingParty,
     string content);
  struct Request{string dataSubject;string requestingParty;
```

```
    string content;}
Request[] public request;
mapping (uint => address)requestToOwner;
mapping (address => string)public signToRequest;
mapping (address => uint) ownerRequestCount;
function saveRequest(string memory _dataSubject,string memory
    _requestingParty,string memory _content) public payable
    returns(bool){
   uint id=request.push(Request(_dataSubject, _requestingParty,
       _content))-1;
   requestToOwner[id] = msg.sender;
   signToRequest[msg.sender] = _requestingParty;
   ownerRequestCount[msg.sender] = ownerRequestCount[msg.sender
      ].add(1);
   emit NewRequest(_dataSubject,_requestingParty,_content);
   return true; }
function getRequest(string memory _dataSubject,string memory
    _requestingParty)public view returns (string memory) {
    for(uint i=0;i<request.length;i++){
        Request storage myRequest = request[i];
        if(keccak256(bytes(_dataSubject)) == keccak256(bytes(
           myRequest.dataSubject)) && keccak256(bytes(
           _requestingParty)) == keccak256(bytes(myRequest.
           requestingParty))){
           return myRequest.content; }
    } return ''no record''; }
 function authRequest (string memory _signature,string memory
    _status,string memory _url,string memory _requestContent)
    public payable returns(bool){
     uint id=agree.push(Agreement(_signature,_status, _url,
        _requestContent))-1;
```

```
        agreeToOwner[id] =msg.sender;
        signToAgree[msg.sender] = _signature;
        ownerAgreeCount[msg.sender] = ownerAgreeCount[msg.sender
            ].add(1);
        emit NewAgreement(_signature,_status,_url,_requestContent
            );
        return true;}
}
```

The proof-of-concept system implemented service requests using JavaScript codes and presented the user interface through web pages as depicted in Figure 6.5. Figure 6.5a illustrates a request form that allows *RqP* to request permission for accessing data. Figure 6.5b presents a screenshot of *RO* page where the request from *RqP* can be viewed and reviewed. *RO* is required to verify the request form and authorise a permission token for data access. Upon completion of each transaction, the entities involved can save a receipt of the transaction using the "getReceipt" button available in the client interface.

(a) The interface of the request form

(b) The client interface of the *RO* page

Figure 6.5: Web pages of BRUE demo in the data transaction of service request

Identity and Records Verification

The identity and records verification phase of the demo is implemented using data transactions between *RqP*, *LAS*, and *RAS*. As there is no direct communication between *LAS*

and *RAS*, both entities have only a relationship with *RqP*. *RqP* shares a permission token *PT* with *LAS*. *LAS* views this token for identity verification and generates a verified identity token *VIT* for *RqP*. *RqP* gets a *VIT* from *LAS* and then shares *VIT*' with *RAS* for verification. *RAS* checks *VIT*' and authorises a token *RPT* with permission information to access PHI from *RRS*. The implementation of the verification phase involves the sharing of tokens between entities using smart contracts. There are three types of tokens exchanged in this phase, which are implemented using smart contracts: *PT*, *VIT*, and *RPT*. The following code snippets show how these tokens are implemented in smart contracts. The *saveToken* function generates and saves tokens in the blockchain, while the *getToken* function retrieves token information from the blockchain for the recipient.

```
contract Token {
  event NewToken (string signature,string receiver,string token)
     ;
  struct Token {string signature;string receiver;string token;}
  Token[] public token;
  mapping (uint => address)tokenToOwner;
  mapping (address => string)signToToken;
  function saveToken (string memory _signature,string memory
     _receiver,string memory _token) public payable returns (
     bool) {
    uint id = token.push(Token(_signature,_receiver,_token)) -
       1;
    tokenToOwner[id] = msg.sender;
    signToToken[msg.sender]= _signature;
    emit NewToken(_signature,_receiver,_token);
    return true;  }
  function getToken (string memory _signature,string memory
     _receiver) public view returns (string memory){
    for(uint i=0;i<token.length;i++){
        Token storage myToken = token[i];
        if(keccak256(bytes(_signature)) == keccak256(bytes(
           myToken.signature)) && keccak256(bytes(_receiver)) ==
```

```
            keccak256(bytes(myToken.receiver))){
                return myToken.token; }
    } return ''wrong input or no record'';}
}
```

In the above code, the *saveToken* function takes in the token type, recipient address, and token information as input. It generates a unique token ID using the keccak256 hash function and saves the token information in the tokenList mapping. The *getToken* function takes in the token type, recipient address, and token information as input and returns the token information for the recipient.

The visual representation of the implemented system is depicted in Figure 6.6, showcasing the user interface of *RqP* web pages during the phases of identity and records verification. Figure 6.6a illustrates the client interface, where *RqP* uploads a token and shares it with *AS* or *RS* through smart contracts. The interface also features a download button, enabling *RqP* to obtain a receipt after token sharing. Figure 6.6b exhibits an interface where *RqP* obtains an authorised token from *RAS* or *LAS* and allows the user to save the token after viewing it, through a download button.

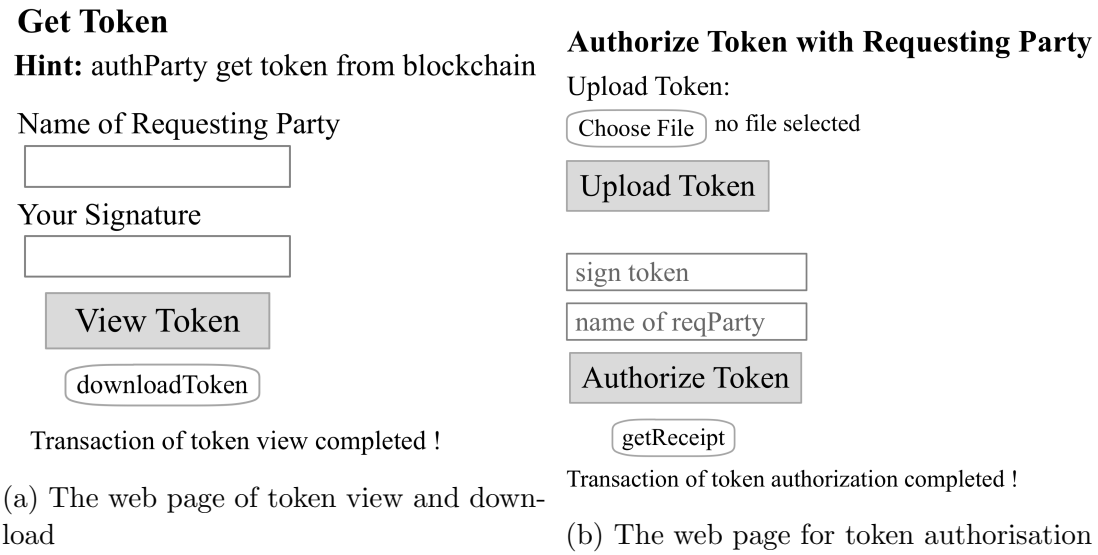
(a) The web page of the token exchange

(b) The web page of token view and download

Figure 6.6: Web pages of BRUE demo- *RqP* page

LAS and *RAS* are entities responsible for authorising data access requests, which require them to obtain tokens from the blockchain to verify permission information. After identity verification, they generate a new authorised token for *RqP*. The client interface for authority organisation operations is shown in Figure 6.7. Authority entities retrieve tokens from the blockchain using a signature and the sender's identity, as shown in Figure 6.7a. Figure 6.7b is an interface used to upload a new protected token and authorise it for

RqP by *AS* or *RS*. After each data transaction, entities can download a receipt record for future reference.



(a) The web page of token view and download

(b) The web page for token authorisation

Figure 6.7: Web pages of BRUE demo - *AS* or *RS* page

Resource Exchange

The *resource exchange* phase involves sharing PHI between *RqP* and *RS*. In the case of cross-jurisdiction data sharing in this demo, *RS* is located in a different domain from *RqP*, and an *RRS* is used. To access PHI, *RqP* is required to share an *RPT*' token with *RRS* for permission verification, after *RRS* sends the information to *RqP*. The same interface as shown in Figure 6.5a is used by *RqP* to share *RPT*' with *RRS*. The *RRS* gets the token *RPT*' and checks the permission information as shown in Figure 6.7a. It should be noted that the final stage of the real PHI exchange was not implemented in this demo. Rather, the focus was on demonstrating the ability of the BRUE system to process and manage permission information and data access authorisation in accordance with the aims and objectives of this study.

Functional Test

The BRUE demo was designed with a user-oriented web application, and it successfully implemented the main phases of data processing, including data sharing, data authorisation, and storage. The above section provides a detailed description of the implementation process, and Table 6.3 provides a result of the functional test of the demo. The BRUE

demo underwent various functional tests, such as data transfer and modification, and it passed all tests. In total, there were three functions that spent gas.

Table 6.3: Results of the functional test of the BRUE demo

Functions	pass the test?	use gas?
send a request	yes	yes
receive a request	yes	no
authorise a token	yes	yes
receive and check a token	yes	no
upload a token	yes	no
share a token	yes	yes
download a receipt	yes	no

6.3.2 Performance Evaluation

In the design of the BRUE system, the entities and data flow topologies are pre-defined. RqP sends a request to start the data flow and RRS receives an RPT token and verifies it to the end. The experiment focuses on the number of request records and their sizes, rather than the number of topologies. The experiment assumes the involvement of an RqP, a LAS, an RAS, and an RRS. In each complete iteration, the demo ran 100 request records of varying sizes. The performance evaluation includes the gas cost and response time of each full iteration. A full iteration involves sending a request, receiving a request, the request authorisation, sharing PT' token, receiving a PT' token, the VIT token authorisation, sharing an VIT' token, receiving an VIT' token, the RPT token authorisation, sharing an RPT' token, and receiving an RPT' token. The process of uploading a token into the client interface is not included in a complete iteration because it does not require gas and has a negligible response time. The gas cost and response time experiments were run on both a local network and the Goerli test network. The following sections present the experimental results for gas cost and response time.

Cost

Figure 6.8 depicts the average gas cost of each complete iteration of data flow with different record sizes run on the local network. It presents the average transaction fee in ether with the record size. The record sizes used were 0.151 KB, 1.312 KB, and 2.312 KB, and the average gas used in each complete iteration was 1246884.27, 5039247, and 6378550 units, respectively, with a gas price of 20 gwei per unit. As the record size increased, the average

transaction fee of each complete iteration increased linearly. Since the experiment runs on the local network, the transaction fee of each complete iteration is stable based on the gas used and gas price.

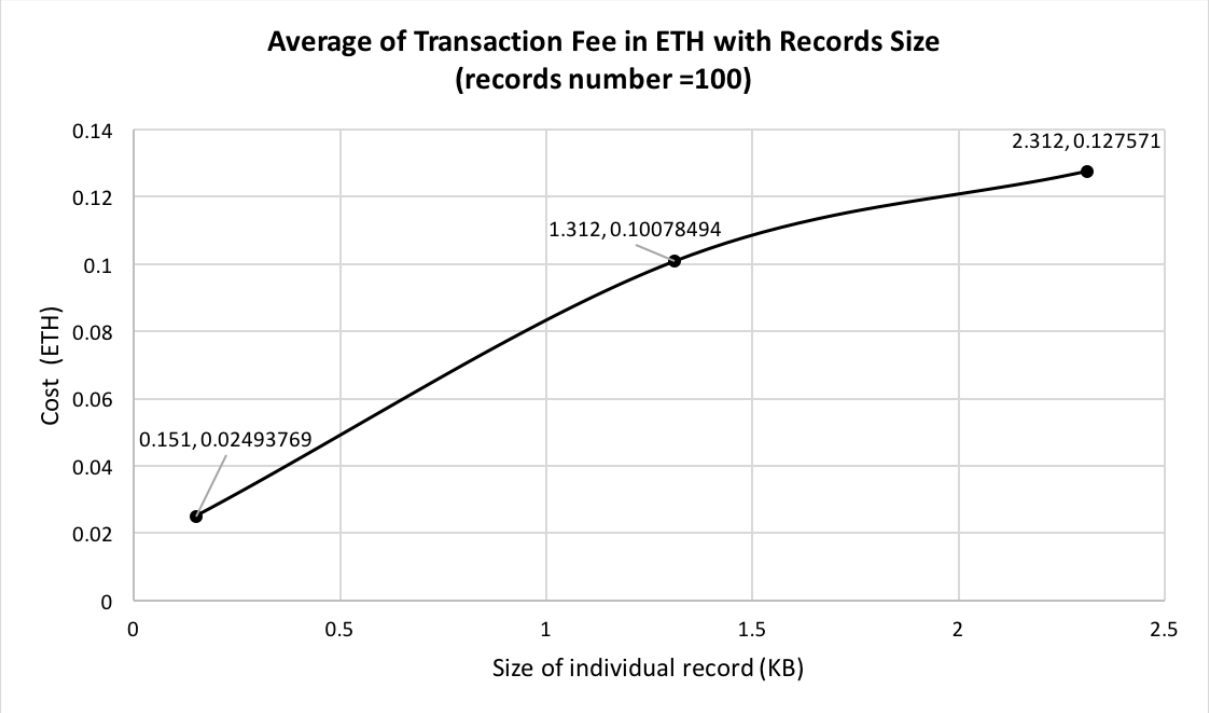
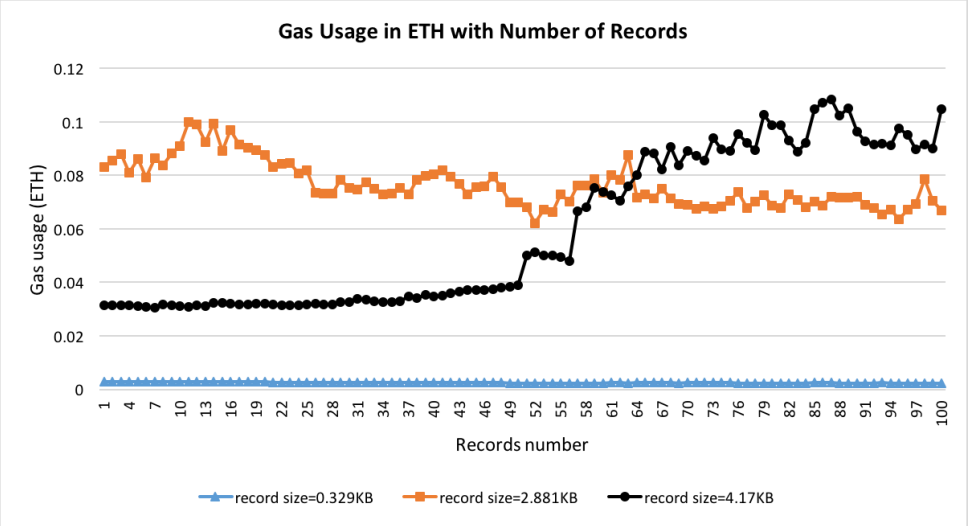


Figure 6.8: Average gas consumption of each complete iteration with the size of records in the local network

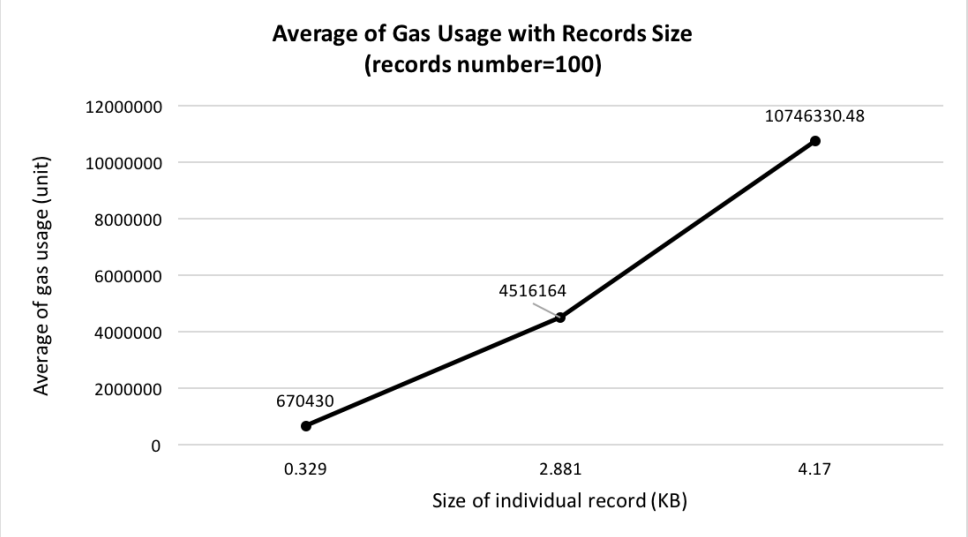
Figure 6.9 presents the gas cost of each iteration with the number of records run on the Goerli test network. The size of the records used were 0.329 KB, 2.881 KB, and 4.17 KB. Figure 6.9b shows the gas used in each iteration with the size of the records. It can be observed that the gas used (units) of each complete iteration was the same if the record size was the same. However, with different sizes of records exchange, the gas used was significantly increased. Figure 6.9a shows the transaction fee of each complete iteration with the number of records.

To compare the transaction fee in different sizes of records, the transaction fee is continuously increased as shown. The transaction fee of each complete iteration in a specific size of records was changeable based on the base fee and priority fee. Here priority fee was 2.5 gwei. The base fee was stable if exchanged record size was 0.329 KB and 2.881 KB. However, when the size of the record was 4.17 KB, the base fee was significantly increased since the number of records up to 30. The transaction fee with ether was also significantly increased.

Overall, the results indicate that the gas cost and transaction fee of each iteration



(a) Transaction fee in ETH with the number of records



(b) Average gas usage with the size of records

Figure 6.9: Gas consumption of each iteration in the Goerli network

are affected by the number and size of the records. As the number and size of records increase, the gas cost and transaction fee also increase. Therefore, in a real-world scenario, it is crucial to consider the gas cost and transaction fee for optimising the data exchange process.

Response Time

Figure 6.10 displays the processing time for each complete iteration based on the number of records processed on the local network. As the size of records increased, the processing time for each iteration also increased. Notably, the response time for each iteration at record sizes of 0.151 KB and 1.312 KB varied as expected before stabilising at a certain point. However, when the record size reached 2.312 KB, the response time for a complete

iteration continued to increase.

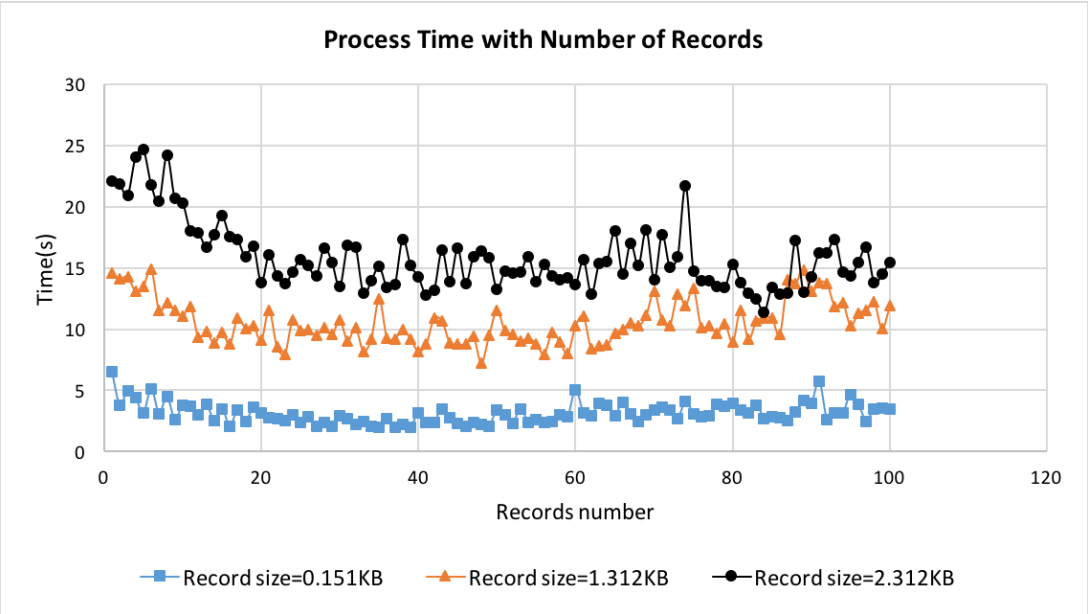


Figure 6.10: Response time of each complete iteration with the number of records in the local network

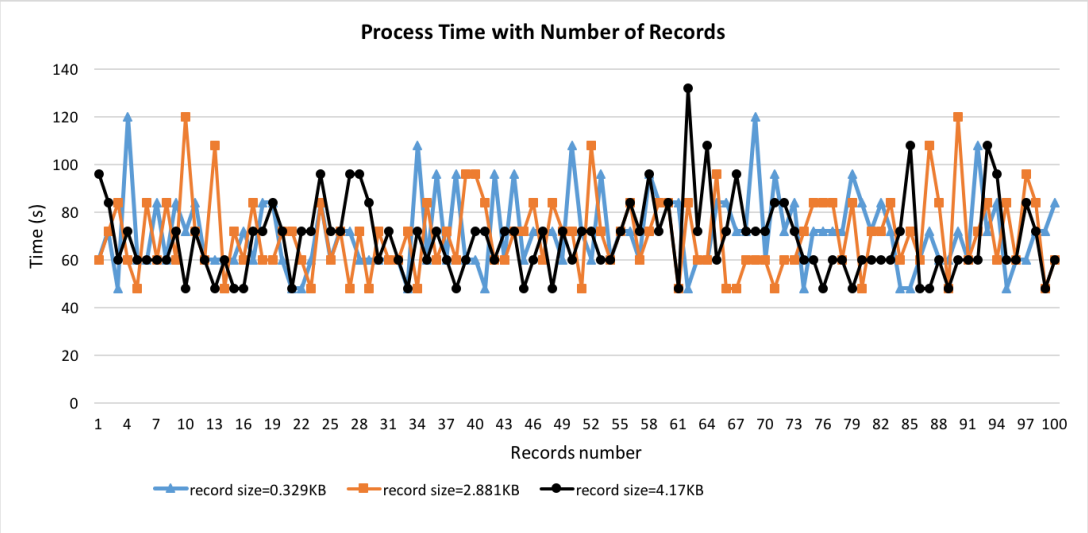


Figure 6.11: Response time of each iteration with the number of records in the Goerli network

Figure 6.11 displays the response time for each iteration based on the number of records processed on the Goerli network. The experiment utilised 100 records with individual record sizes set to 0.329 KB, 2.881 KB, and 4.17 KB. Each iteration solely consisted of transactions that consumed gas, including request sending, token generation, and exchange, as the experiment ran smart contract codes on the testnet network without any user interface involvement. The response time of each iteration was primarily influenced by block mining time (typically 12 seconds per block) in the Goerli network since the

response time did not include transaction time for data exchange between the client interface and smart contracts. During busy times on the test network, block mining time was observed to increase. Figure 6.11 shows that while the response time for each iteration at a specific record size fluctuated, it tended to stabilise when the record size was variable. These results suggest that there is no strong relationship between record size and response time within a certain range of record sizes.

6.4 BRESPE

BRESPE is a framework designed to enable privacy-preserving PHI exchange using blockchain technology. It offers a different solution for implementing secure PHI exchange in the thesis. To simplify the process, the BRESPE demo involves only three entities: *DR*, *DS*, and *DC*. The demo's source code [15] is open to the public. The demo has been implemented using a client interface (web pages) developed with the React [16] framework, Truffle, JavaScript language, and smart contracts in Solidity. Additionally, the demo utilised the *FileSaver* function for document download and the *SHA* [17] function for hash calculation. This section presents the details of the BRESPE demo implementation and its performance in terms of gas cost and response time for each complete iteration, both on the local network and the Goerli test network.

6.4.1 Implementation

The main implementation of the BRESPE demo includes data transactions in three phases: service request, consent and sticky policy construction, and resource exchange. The initialisation phase, which involves entity registration, key generation, and distribution, has not been included as it is similar to the AudiWFlow implementation presented in Section 6.2. Figure 6.12 depicts the client interface structure of the BRESPE demo, which provides a comprehensive representation of the complete data flow among entities. The figure illustrates the data flow between entities through graph arrows. Using the client interface, *DR* can send a request, receive consent, obtain a sticky policy, and obtain a hash value of the record. *DS* can receive a request, view the request, and authorise it.

[15] BRESPE demo codes: `https://github.com/Jency/BRESPE`
[16] React: `https://www.trufflesuite.com/boxes/react`
[17] js-sha3: `https://github.com/emn178/js-sha3`

DC can receive consent, build a sticky policy, and share the hash value of the record. The following section provides a detailed description of the demo implementation by the sequence of the data transactions between DR, DS, and DC.

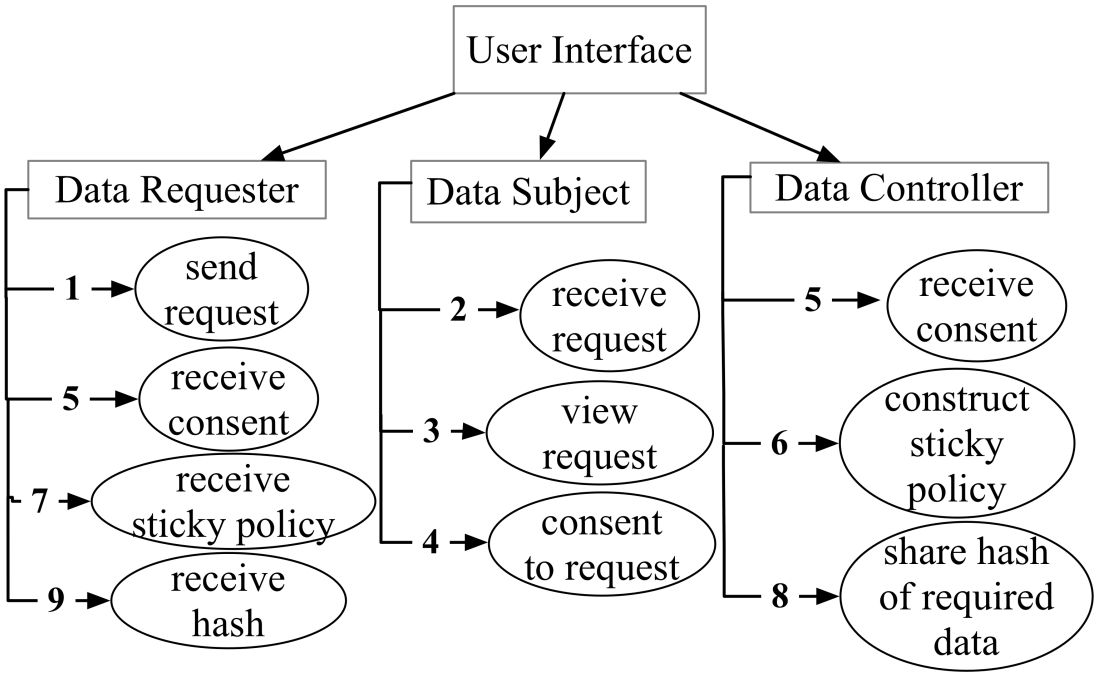
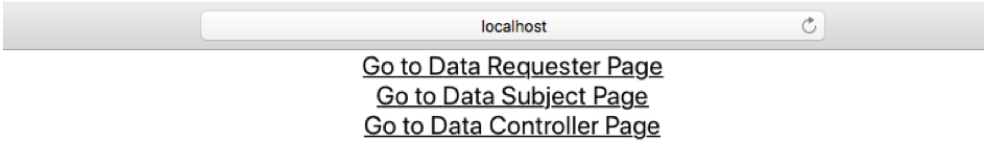
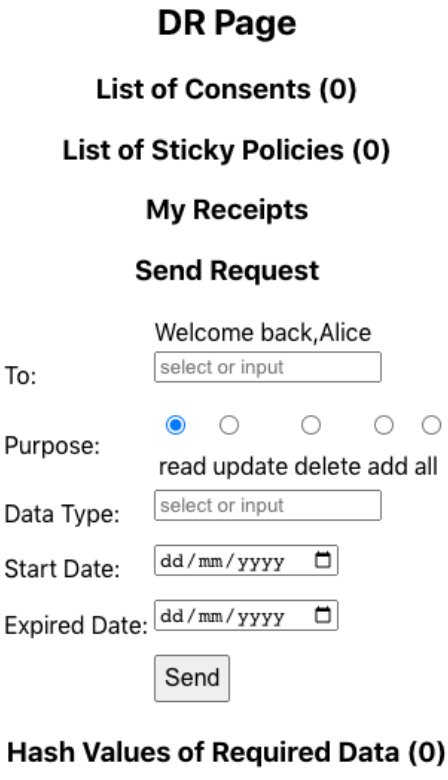


Figure 6.12: Structure of proof-of-concept system implementation of BRESPE

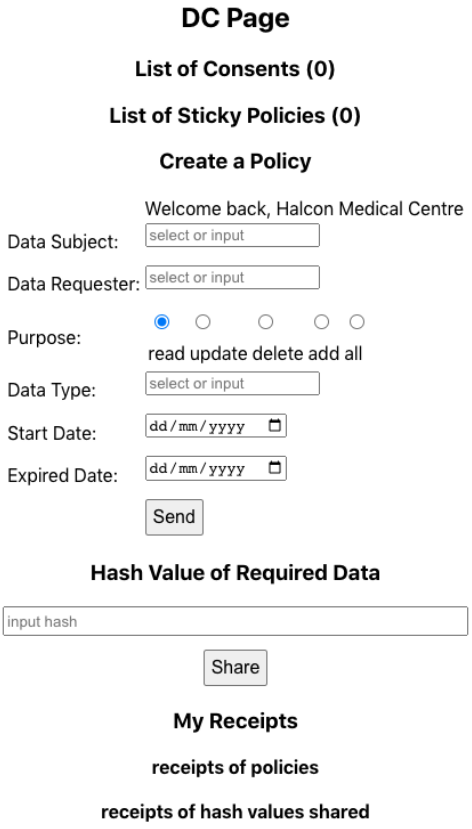
The client interface of the BRESPE demo comprises a total of four web pages, including a home page and three sub-pages (see Figure 6.13). The home page as shown in Figure 6.13a, serves as a navigation page, providing links to each entity's sub-page. One of the sub-pages is designed for DR to send a request to DS. A screenshot of this web page is shown in Figure 6.13b. The page includes a request form, lists of consents, sticky policies, receipts, and hash values of records. In the request form, DR selects the receiver of the request, purpose, data type, expected start date, and expiration date. Another sub-page is dedicated to DC. A screenshot of this page is presented in Figure 6.13c. It comprises a form to build a policy based on consents for DR, a list of authorised consents, historical policy records, a list of receipt records, and a form to share a hash value of data. The third sub-page is for DS, as shown in Figure 6.13d. The top part of this page displays a list of request records from DR, which DS can view and edit using the "View Details" button. The middle part presents a list of consent records, while the next part lists receipt records. The bottom part is for historical records of hash values.



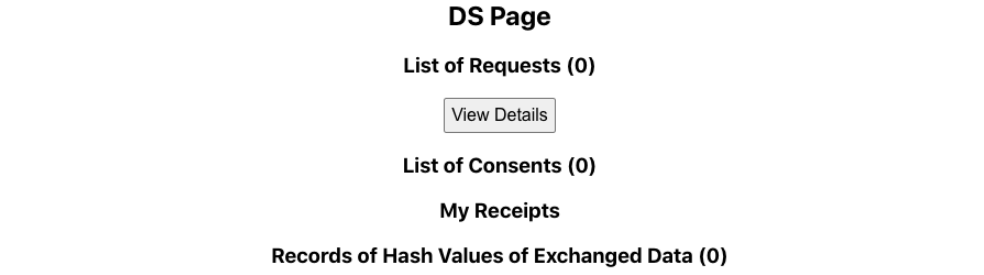
(a) Homepage of the BRESPE demo



(b) Screenshot of the *DR* web page



(c) Screenshot of the *DC* web page



(d) Screenshot of *DS* web page

Figure 6.13: The main web pages of the user interface in the BRESPE demo

Transactions Implementation of the Data Requester

In the BRESPE demo, the data flow was initiated by *DR* through a request. A smart contract was then activated to send the request to *DS* without permission in the blockchain. After this transaction, *DR* received a receipt. When *DS* authorised the request, *DR* received a record of consent, and when *DC* permitted, it also received a record of the policy. The following is an implementation of the smart contract for the *DR*'s request service. The *addRequest* function is responsible for saving the request information in the blockchain. It collects all information of a request from *DR* and counts the records of requests. To prevent duplicate records of requests, the *hashCompareInternal* and *checkCondition* functions are used to verify the records submitted by *DR*. The *hashCompareinternal* function returns a Boolean value after comparing the hash for the entity.

```
contract Request {
  struct Number {uint b_count;uint b_number;}
  Number[] public number;
  string[] public condition=new string[](0);
  string[] public nameDO = new string[](0);
  string[] public nameRqP = new string[](0);
  string[] public purpose= new string[](0);
  string[] public startDate= new string[](0);
  string[] public expired = new string[](0);
  string[] public status = new string[](0);
  function hashCompareInternal(string memory a,string memory b)
     pure private returns (bool) {
    return keccak256(bytes(a)) == keccak256(bytes(b));}
  function checkCondition(string memory _condition) public view
     returns(bool){
    for(uint i=0; i<condition.length; i++){
      if(hashCompareInternal(condition[i],_condition)){
        return true; } } return false; }
 function addRequest(string memory _condition,string memory
    _nameDO,string memory _nameRqP,string memory _purpose,string
```

```
    memory _startDate, string memory _expired, string memory
    _status) public{
    if(checkCondition(_condition)){ return;
    }else{
        condition.push(_condition);
        nameDO.push(_nameDO);
        nameRqP.push(_nameRqP);
        purpose.push(_purpose);
        startDate.push(_startDate);
        expired.push(_expired);
        status.push(_status);
        number.push(Number(condition.length, block.number));}}
}
```

Figure 6.14 shows the web page of DR after transactions are completed. The demo implementation of BRESPE allows DR to view and download important information such as consent records, policy records, hash values, and receipts of transactions. This helps ensure that all entities have access to the necessary information and can keep a record of the transactions. Having a "download" button also allows DR to save the records locally, which can be useful for future auditing purposes.

Figure 6.14: Screenshot of the DR web page with transactions

The BRESPE demo has implemented the feature to generate receipts as proof of evidence for record verification (see Figure 6.14). The receipt includes a block id, a block hash, information about the request (requester, receiver, purpose, start date, expired date, and conditions), and a hash value of the request. By including this information, the receipt can provide proof that a transaction has occurred on the blockchain and that the information in the request has not been tampered with since it was recorded. Figure 6.15

shows an example of a receipt if downloaded. The receipt in the BRESPE demo is saved in JSON format. It is a good choice because it can be easily parsed by the client interface and can be easily shared and verified by different entities.

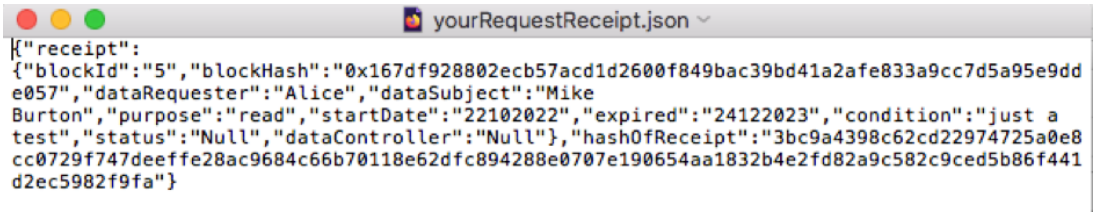


Figure 6.15: An example of a receipt

Transactions Implementation of the Data Subject

Figure 6.16 presents the web page of *DS* after the completion of transactions. The *DS* is provided with the capability to view, modify, and authorise a request via this web page. Upon receiving a request from *DR*, *DS* can either view the request details and then grant consent or directly grant consent. The web page displays a record of the request, a record of consent, and a transaction receipt for the current transaction. An orange folded form is visible at the top of the web page, which contains the details of a request form with editable properties such as date, purpose, date type, and receiver (the data controller). Additionally, *DS* receives a record of the hash value for the required PHI from *DC*. The "download" button enables *DS* to save receipts and historical records of the hash value for required PHI. The format of a receipt is the same as the receipt displayed in Figure 6.15.

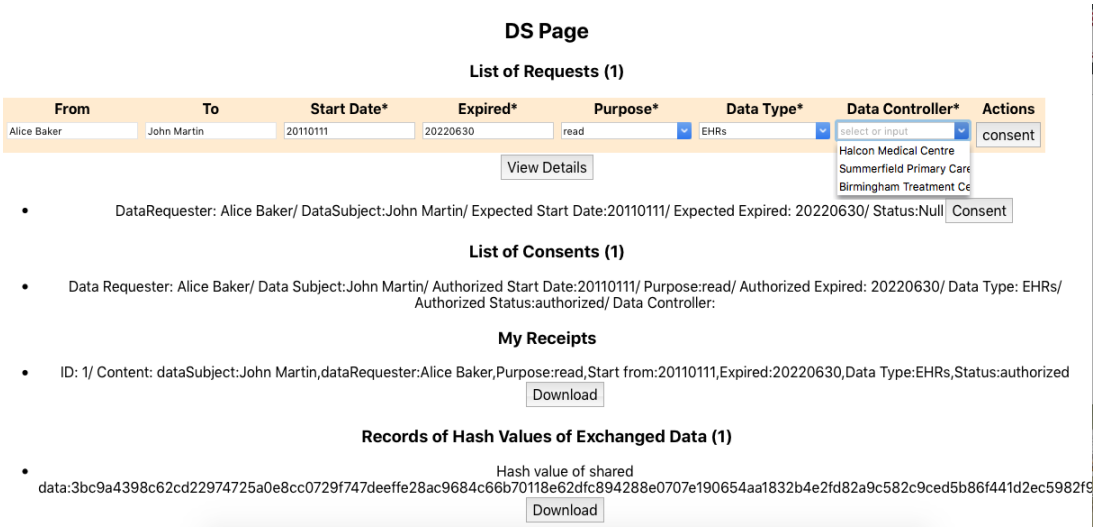


Figure 6.16: Screenshot of the *DS* web page with transactions

The implementation of data transactions in the presented system is realised through a smart contract. The following are codes of the main functions of the smart contract. The function *addConsent* is responsible for saving a record of consent to the blockchain. It accepts the input information of a request, including the requester, receiver, purpose, start date, expired date, and conditions. Upon execution, the function adds the consent record to the blockchain for record-keeping and auditing purposes. To retrieve information from the blockchain, several functions are implemented in the smart contract. For example, the *getCondition* function returns the conditions associated with a particular request stored in the blockchain. The *getNameDo* function returns the name of the data subject associated with a particular request. These functions play a crucial role in enabling the different entities to access and interact with the data in the system.

```
contract Consent {
  string[] public c_condition=new string[](0);
  string[] public c_nameDO = new string[](0);
  string[] public c_nameRqP = new string[](0);
  string[] public c_purpose= new string[](0);
  string[] public c_startDate= new string[](0);
  string[] public c_expired = new string[](0);
  string[] public c_status = new string[](0);
  string[] public c_nameRO = new string[](0);
  function getCondition(uint index) view public returns(string
     memory){
    return (condition[index]); }
  function getNameDo(uint index) view public returns(string
     memory){
    return (nameDO[index]); }
  function getNameRqP(uint index) view public returns(string
     memory){
    return (nameRqP[index]); }
  function getPurpose(uint index) view public returns(string
     memory){
    return (purpose[index]); }
```

```
  function getStartDate(uint index) view public returns(string
      memory){
    return (startDate[index]); }
  function getExpired(uint index) view public returns(string
      memory){
    return (expired[index]); }
  function getStatus(uint index) view public returns(string
      memory){
    return (status[index]); }
  function addConsent(string memory _condition, string memory
      _nameDO, string memory _nameRqP, string memory _purpose,
      string memory _startDate,string memory _expired, string
      memory _status, string memory _nameRO) public{
       c_condition.push(_condition);
       c_nameDO.push(_nameDO);
       c_nameRqP.push(_nameRqP);
       c_purpose.push(_purpose);
       c_startDate.push(_startDate);
       c_expired.push(_expired);
       c_status.push(_status);
       c_nameRO.push(_nameRO); }
}
```

Transactions Implementation of the Data Controller

Figure 6.17 outlines the web page interface of *DC* and the types of receipts it generates. The interface contains three parts: a list of consents from *DS*, a list of sticky policies, and a list of receipts. Each part displays the number of transactions associated with it. The interface also includes a "download" button to save records of consents and receipts. Two types of receipts are generated by *DC*: a general receipt and a receipt of a hash value. The general receipt records the transaction of sticky policy construction and has the same structure as the receipt shown in Figure 6.15. The receipt of a hash value only

includes a hash because the transaction involves the exchange of a hash value of a record.

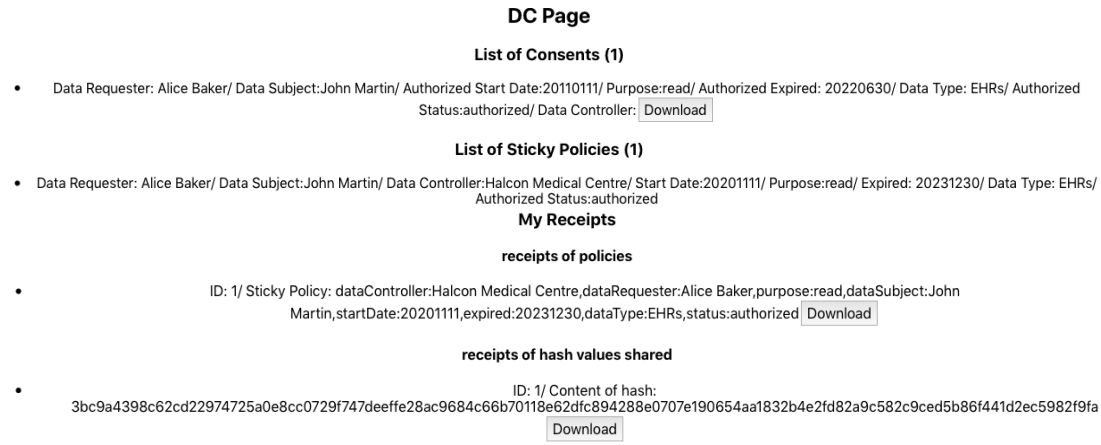


Figure 6.17: Screenshot of the *DC* web page with transactions

A smart contract is activated to save a record of the sticky policy when *DC* generates a policy for *DR* based on consent from *DS*. The following shows codes of a function to achieve this transaction. The *addStickyPolicy* function appears to save a record of a policy with some properties. This indicates that the function is used to create a record of a policy generated by *DC* in response to consent from *DS*. The *saveHash* function appears to save a hash value of the required PHI by *DC*.

```
contract Policy {
  string[] public s_condition=new string[](0);
  string[] public s_nameDO = new string[](0);
  string[] public s_nameRqP = new string[](0);
  string[] public s_purpose= new string[](0);
  string[] public s_startDate= new string[](0);
  string[] public s_expired = new string[](0);
  string[] public s_status = new string[](0);
  string[] public s_nameRO = new string[](0);
  string[] public hashData = new string[](0);
  function addStickyPolicy(string memory _condition, string
     memory _nameDO, string memory _nameRqP, string memory
     _purpose, string memory _startDate,string memory _expired,
     string memory _status, string memory _nameRO) public{
    s_condition.push(_condition);
    s_nameDO.push(_nameDO);
```

```
        s_nameRqP.push(_nameRqP);
        s_purpose.push(_purpose);
        s_startDate.push(_startDate);
        s_expired.push(_expired);
        s_status.push(_status);
        s_nameRO.push(_nameRO); }
    function saveHash(string memory _hash) public{
          hashData.push(_hash); }
}
```

Functional Test

The BRESPE demo has been implemented successfully with a simple web application. Table 6.4 listing the results of the functional test provides a clear overview of the tests that were performed and their outcomes. According to the below table, a complete iteration of the experiment involves eight passed functions. This indicates that the main steps of data processing including service request, consent management, record verification and exchange were implemented successfully in the client interface.

Table 6.4: Results of the functional test of the BRESPE demo

Functions	pass the test?	use gas?
send a request	yes	yes
receive and view a request	yes	no
edit a request	yes	no
authorise a request	yes	yes
build a policy	yes	yes
receive a policy	yes	no
share a hash	yes	yes
receive a hash	yes	no

6.4.2 Performance Evaluation

The BRESPE demo was deployed on both the development environment of the Ethereum local network and the Goerli test network. It involved three entities: data requester, data subject, and data controller. The flow of data transactions between them was pre-defined, with the data requester initiating the flow by making a request, and the data

subject verifying and consenting to the request. The data controller then verified the consent, built policies, and shared the hash value. Each iteration of the data transactions, including request sending, request viewing, request authorisation, consent check, sticky policy construction and sharing, and hash exchange, involved 100 requests. To evaluate the scalability and feasibility of the proof-of-concept system, the performance was analysed in terms of the gas cost and response time for each complete iteration below.

Cost

The experimental results pertaining to gas cost and transaction fees are presented in the above figures. Figure 6.18 illustrates the average gas cost of each complete iteration in ether with different sizes of request records run on the local network. The sizes of request records used were 0.329 KB, 2.881 KB, and 4.17 KB. The corresponding gas used was 653506, 2446571, and 5919102, respectively. The gas price per unit was fixed at 20 gwei. It was observed that the average transaction fee increased slowly with an increase in the size of request records, as expected.

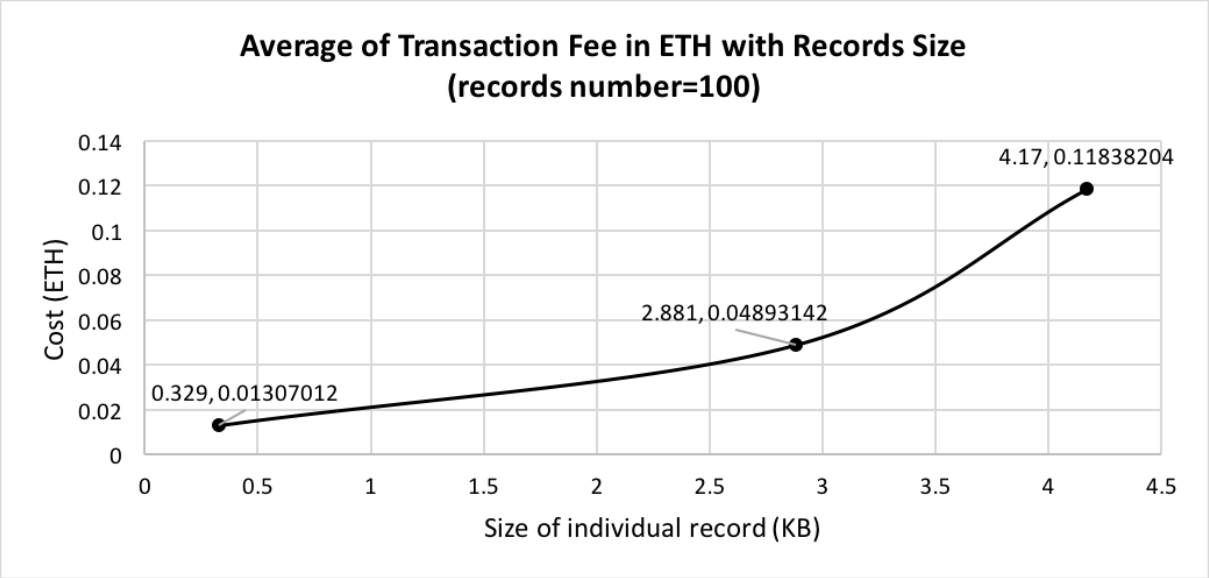
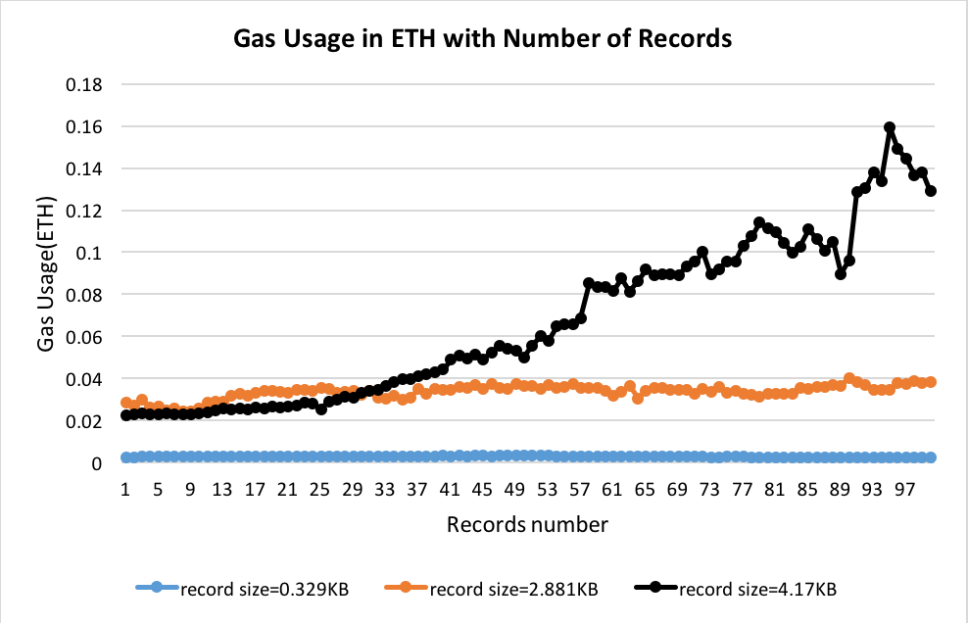
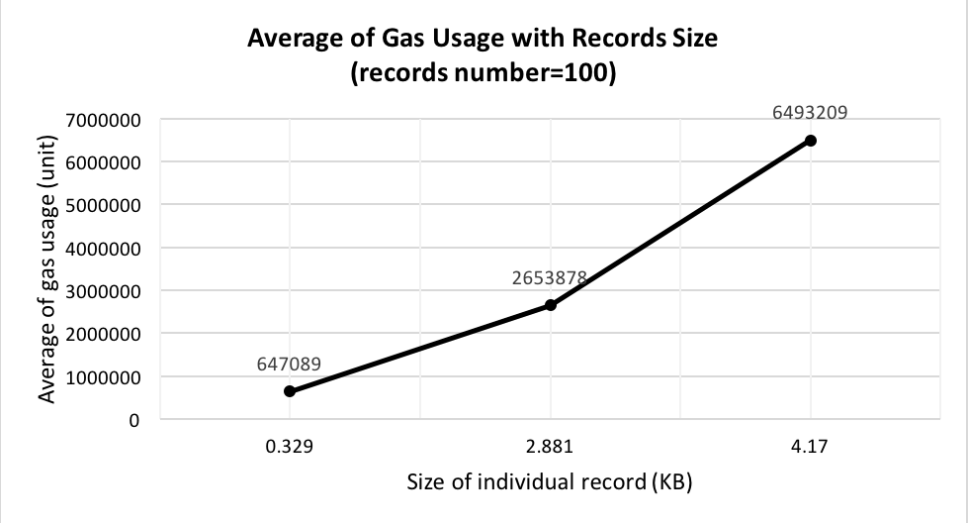


Figure 6.18: Average of gas consumption of each iteration with the size of records in the local network

Figure 6.19 depicts the gas cost of each iteration with a different number of records run on the Goerli test network, where the sizes of records used were 0.329 KB, 2.881 KB, and 4.17 KB. Figure 6.19b shows the gas used (in units) of each complete iteration with different sizes of records. The results indicated a strong relationship between the gas used and the size of the records. Each iteration had a stable point of gas usage, and when the



(a) Transaction fee in ETH with the number of records



(b) Average gas usage with the size of records

Figure 6.19: Gas consumption of each iteration in the Goerli network

size of individual records was increased through the network, the gas used also increased accordingly. Figure 6.19a presents the transaction fee of each iteration with a different number of records and sizes of records. The transaction fee continuously increased with an increase in the size of records. For record sizes of 0.329 KB or 2.881 KB, the transaction fee of each iteration had minor fluctuations compared to that of the record size of 4.17 KB. The priority fee of each transaction was stable up to 2.5 gwei, while the base fee changed irregularly. Furthermore, if the demo was run on the Goerli network during busy times, the base fee was observed to be higher.

Response Time

The response time of a data transaction is a key performance indicator that encompasses the duration of various processes such as request service processing, consent, sticky policy construction, and record exchange. Figure 6.20 illustrates the response time of each complete iteration with the number of records processed on a local network. It is evident from the figure that with an increasing number of request records, the response time varies considerably depending on the size of the records. Specifically, larger-sized records correspond to longer response times. However, it should be noted that the response time may also be affected by the hardware used in the experiment, except for the size of individual records.

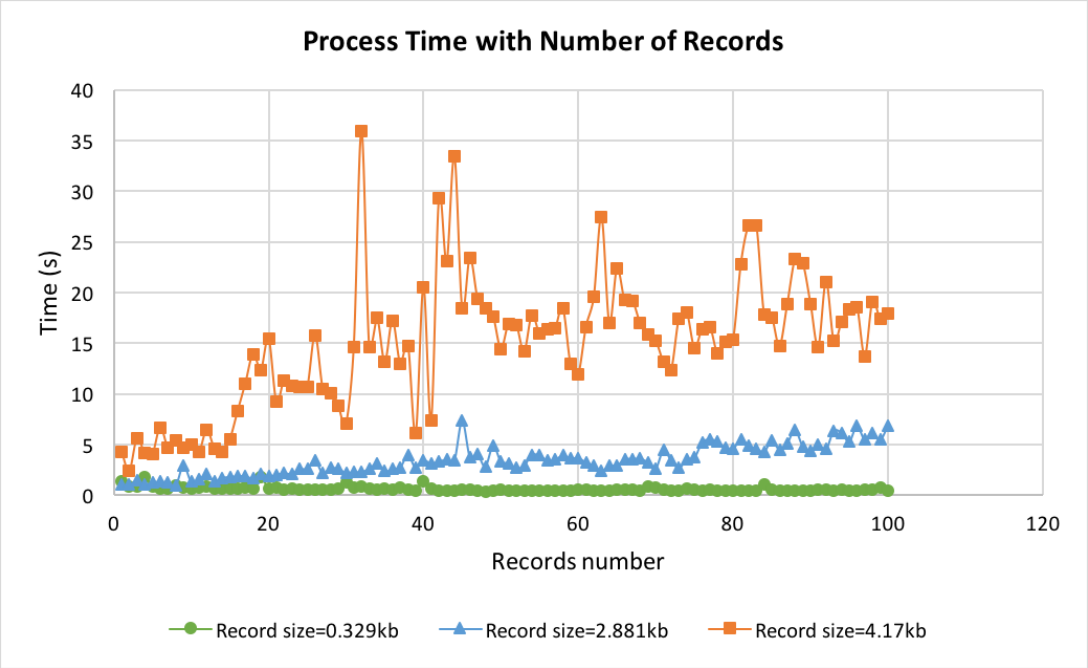


Figure 6.20: The response time of each complete iteration with the number of records in the local network

Figure 6.21 presents the response time of each iteration with the number of records processed on the Goerli network. In this case, the response time of each iteration for a given record size was found to be fluctuating. Nonetheless, the average response time tends to converge to a stable value with an increasing number of records, even as the record size grows. An iteration only included transactions that used gas. The response time here refers to the time taken for transactions in request sending, request authorisation, policy construction, and hash value exchange. Notably, the total time taken for block mining of an iteration is approximately equal to the response time of a complete iteration since the

response time does not account for the time taken for data exchange between the client interface and smart contracts. The block mining time can be affected by the network status and the number of miners involved in the network.

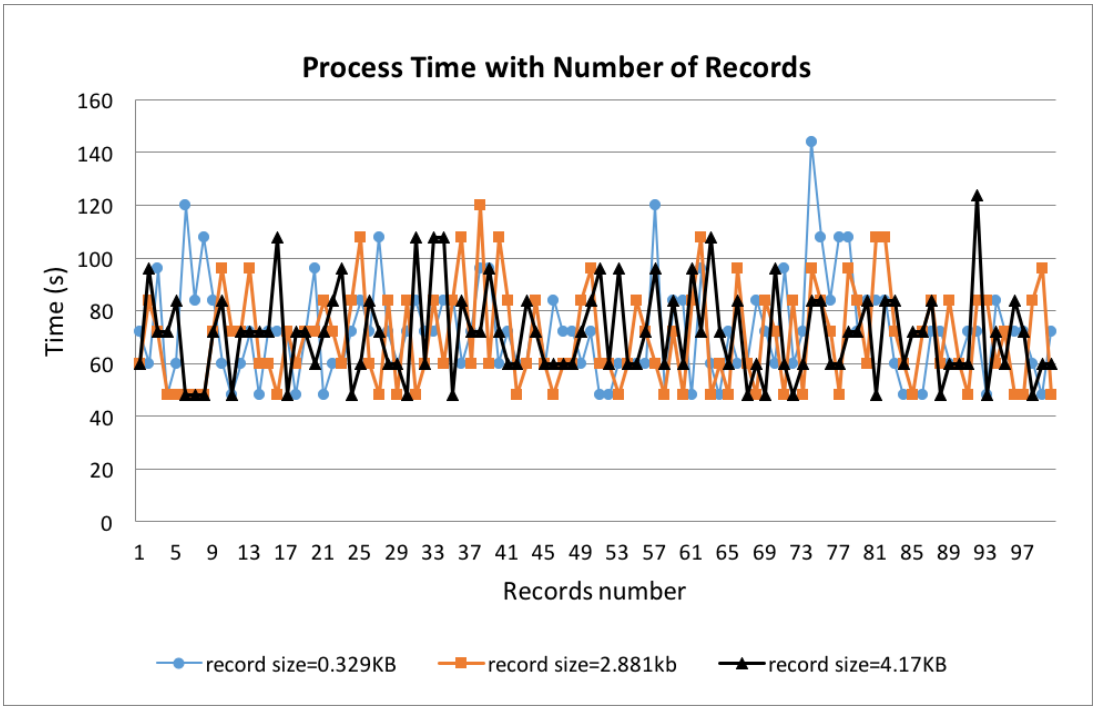


Figure 6.21: The response time of each iteration with the number of records in the Goerli network

6.5 Discussion

In the context of PHI exchange, the AudiWFlow framework was designed for data exchange workflow between different entities located in the same jurisdiction; the BRUE framework was designed to share PHI across jurisdictions; and the BRESPE framework was designed for secure PHI exchange while preserving privacy. The following discusses three frameworks implementations how meets objective 4 and takes a comparison between system performance between BRUE and BRESPE frameworks.

The problem of accountability can arise when different entities cooperate towards a common goal. AudiWFlow was proposed as a solution to this problem by implementing a peer-to-peer distributed architecture based on a public Blockchain. The approach discussed in Section 5.2 was designed to address the issue of auditing workflows while ensuring accountability, non-repudiation, confidentiality, availability, and collusion detection. In the experiments conducted in Section 6.2, the AudiWFlow prototype was tested

on a local private test network with a maximum of 20 entities and a low difficulty parameter of the blockchain block generation to improve the response time. While a public blockchain makes the framework more robust, it does come at the cost of processing delay and integration complexity.

BRUE, a PHI exchange scheme, focuses on data sharing across jurisdictional borders using cutting-edge technologies. In Section 6.3, a proof of concept system was implemented to demonstrate the feasibility of this framework. BRESPE is a privacy-preserving framework for sharing PHI between different entities, which aims to balance the need for data privacy protection and the convenience of data access and sharing. This framework refers to sticky policies that govern data use in potentially complex ways due to the varying concerns of different entities involved in information sharing in this setting. A simple prototype system of BRESPE was implemented in Section 6.4.

The implementation of AudiWFlow uses a public blockchain and smart contracts to ensure traceability and non-repudiation with an audit trail. It also guarantees confidentiality and integrity by encrypting data and accompanying it with signatures. However, there still involves direct data transactions between entities in the workflow that causes a possible problem of cross-jurisdictional PHI exchange if entities do not know each other. The prototype of BRUE is implemented to highlight the cross-jurisdictional PHI exchange and the BRESPE's system concerns the need for data privacy protection and authorisation management. BRESPE designs a protocol for data sharing using sticky policies and enables entities to execute these policies through smart contracts. The use of PDRs in BRUE and BRESPE provides evidence of proof for traceability and non-reputation. Additionally, the rule of minimal sensitive data exchange through the communication channel enhances data privacy protection. These features enable BRUE and BRESPE to ensure data privacy protection, confidentiality, transaction auditing, and policy enforcement based on user consent and data governance regulations. Given the focus of the research on PHI exchange, the rest of this section mainly discusses the performance of BRUE and BRESPE.

In order to evaluate the performance of BRUE and BRESPE, both prototype systems were experimentally tested on the Goerli test network. The experiments consisted of 100 iterations of a completed run, and the size of the individual exchanged data records varied between 0.329 KB, 2.881 KB, and 4.17 KB. Due to the universality of data, only experi-

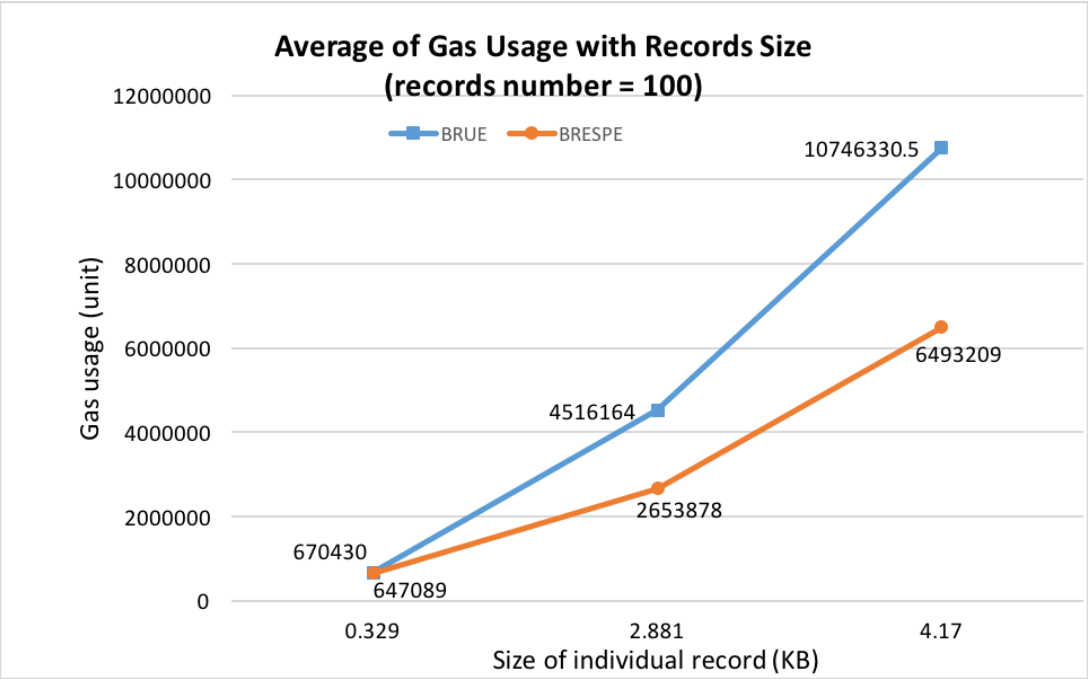
ments conducted on the Goerli test network were discussed. To enable a fair comparison between the two frameworks, the experiments were conducted using the same hardware and configuration environment. The transaction fee was determined by the gas usage G_{used}, base fee B_{fee}, and priority fee P_{fee}. The priority fee was set to 2.5 gwei and the base fee was variable. The transaction fee T_{fee} was calculated as $T_{fee} = G_{used} \times (P_{fee} + B_{fee})$.

Figure 6.22 compares the gas cost and response time of the BRUE and BRESPE implementations. The average gas usage for 100 iterations of both implementations are presented in Figure 6.22a. The data sources of both figures come from Figure 6.9 and Figure 6.19. The gas usage of each iteration was found to be stable and not related to the number of records. As expected, Figure 6.22a shows a strong positive relationship between the size of individual exchanged records and gas usage. Despite the BRUE demo involving fewer steps that spent gas, the average gas usage for BRUE was found to be higher than BRESPE.

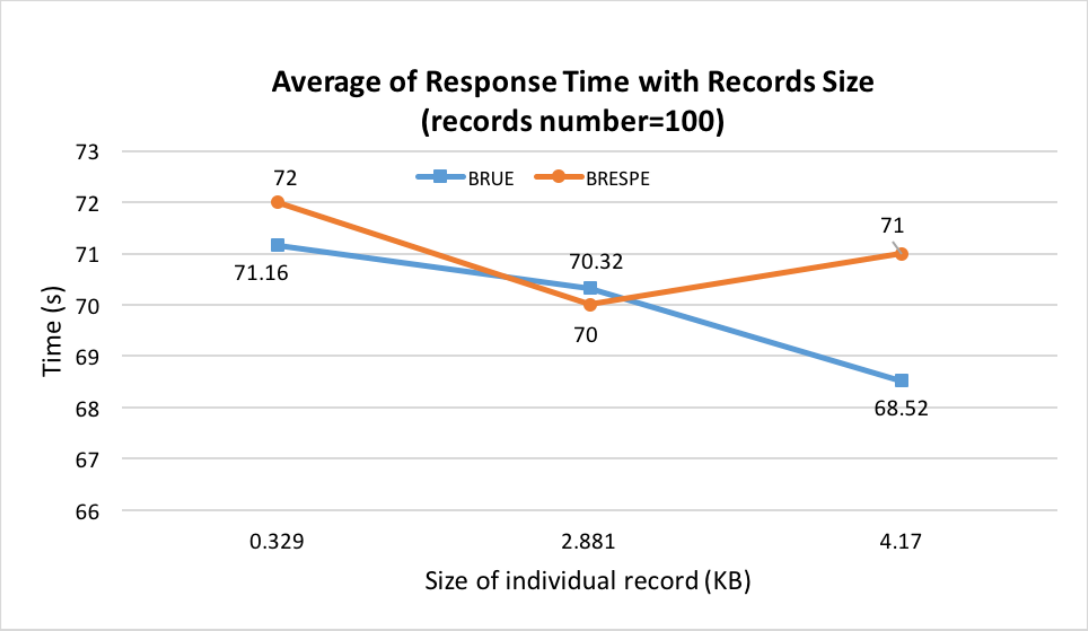
Figure 6.22b shows the average response time for a full iteration run 100 times. The average response time of BRUE decreased while that of BRESPE was an irregular curve. Although both prototypes showed fluctuation in their average response times, all data only showed a minor wave. Since all involved data transactions were selected that used gas, the response time T_{model} of a complete iteration was approximately equal to the total mining time of n blocks. It can be expressed as $\sum_{m=1}^{n} T_{block}m = T_{block}1 + T_{block}2 + \cdot + T_{block}(n - 1) + T_{block}n$. The average response time was calculated as $T_{model-avg} = \frac{\sum_{m=1}^{n} T_{block}m}{n}$.

BRUE involved three data transactions in a complete iteration in the experiment, and the average response time of a complete iteration was calculated as $T_{BRUE-avg} = \frac{\sum_{m=1}^{3} T_{block}3}{3}$. In comparison, BRESPE involved four data transactions in a complete iteration of experiments, and the average response time was calculated as $T_{BRESPE-avg} = \frac{\sum_{m=1}^{4} T_{block}4}{4}$. The experiments were run on the Goerli test network, where the time of block generation is approximately 12 seconds. With results shown in Figure 6.11 and Figure 6.21, although few response times were not a multiple of 12 seconds (such as 60 seconds), the data in Figure 6.22b were as expected.

In summary, the performance evaluation showed that while BRESPE spent less gas than BRUE, it required more response time to process data than BRUE. These results provide insight into the performance trade-offs between the two proposed frameworks for PHI exchange.



(a) Average gas usage with the size of records



(b) Average response time with the size of records

Figure 6.22: Performance comparison between BRUE and BRESPE implementations

6.6 Comparative Analysis of Proof-of-concept System Implementation and Performance

Sections 6.2, 6.3 and 6.4 individually introduced a prototype system implementation and evaluated system performance about gas cost and response time. Section 6.5 then discussed these implementations and compared performance between BRUE and BRESPE frameworks. To better answer research question 3 and objective 4 mentioned in the chapter 1, this section conducts a research method of comparative analysis to evaluate frameworks implementations and performances including three proposed frameworks and four selected frameworks. Four selected existing frameworks (Azaria et al. 2016, Zhang, White, Schmidt, Lenz & Rosenbloom 2018, Mohey Eldin et al. 2023, Lee et al. 2022) are also discussed framework design in the section 5.5 of the chapter 5. The following table 6.5 summarises the comparative metrics among these frameworks, including coding language, selected blockchain platform and test network for prototypes implementations and average gas usage and response time for prototypes systems performance evaluation.

Table 6.5: Comparative analysis of system implementation and performance

Frameworks (models, schemes)	Coding language	Selected deployed blockchain	Test Network	Performance metric of evaluation
MedRec [18] (Azaria et al. 2016)	Python, Go	Ethereum	Geth	Not mentioned

[18]More details of implementation: `http://hdl.handle.net/1721.1/109658`

Frameworks (models, schemes)	Coding language	Selected deployed blockchain	Test Network	Performance metric of evaluation
FHIRchain (Zhang, White, Schmidt, Lenz & Rosenbloom 2018)	JavaScript, Solidity	Not mentioned	A private testnet of the Ethereum blockchain	Not mentioned
FBS (Mohey Eldin et al. 2023)	Not mentioned	Consortium blockchain	Not mentioned	Average speed of different transactions
PIE (Lee et al. 2022)	Node.js, Go	Hyperledger 2.3.1	Not mentioned	Execution time of different transactions
AudiWFlow	Java, Solidity	Ethereum	Ganache-CLI	Average gas usage with records size, response time
BRUE	JavaScript, Solidity	Ethereum	Ganache-CLI, Goerli	Average gas usage with records size, response time
BRESPE	JavaScript, Solidity	Ethereum	Ganache-CLI, Goerli	Average gas usage with records size, response time

From the table above, it can be seen that these selected frameworks for comparison have all implemented a prototype system, mainly used to demonstrate the feasibility of framework design. These prototypes implementations selected JAVA, JavaScript, Python or Node.js to code. They coded smart contracts with Go and Solidity. Different prototypes deployed on different blockchains. However, not all prototype implementations use quantifiable metrics for performance evaluation. MedRec and FHIRchain were not introduced the performance evaluation in their papers. Besides, gas usage is not consid-

ered to compare because it is not be discussed in these four selected existing prototypes. Therefore, the following only discusses the single performance metric of execution time for prototype systems between FBS, PIE, AudiWFlow, BRUE, and BRESPE frameworks.

The prototype system of FBS executed queries and messages multiple times to test the average execution time for different record sizes. The average execution time of queries was between 68 and 100 milliseconds, and the average execution speed of write operations was between 0.944 and 19.041 seconds. Lee et al. (Lee et al. 2022) evaluated the execution time of EMR-sharing process and key-sharing process of PIE system. The execution time of EMR-process includes time of EMR upload and download. The EMR records ranged from 0.4 KB to 100 MB. The average execution time of EMR upload was between 2.1 and 5.7 seconds. The average execution time of EMR download was between 0.01014 and 2.7 seconds. The average execution time of re-encryption key sharing was 3.3543 seconds. The prototype implementations of AudiWFlow, BRUR and BRESPE frameworks executed a complete blockchain-based transaction process to get the average response time. The average of response time of AudiWFlow prototype in a record size 10 KB was 25 seconds. The execution time of BRUE prototype was between 68.52 and 71.16 seconds and of BRESPE prototype was between 70.32 and 72 seconds. If only compared the speed of blockchain-based transaction process, AudiWFlow, BRUE and BRESPE prototype systems execute slower than FBS and PIE systems. This is because the execution time obtained from experiments about the three proposed framework systems includes more operational steps than FBS and PIE systems. In addition, the BRUE and BRESPE systems were simulated in the Goerli network. These results were affected by network congestion, but these execution time will be very close to their actual execution time in the Ethereum network. To be noted, there is necessary to optimise the protocol design and implementation for the proposed frameworks to reducing the execution time.

Chapter 7

Conclusion

This chapter serves as the concluding section of the thesis, we begin by summarising the research work undertaken to explore the use of blockchain technology in facilitating the exchange of personal health information. The rest of this chapter discusses the limitations encountered during the proposed frameworks design and prototypes implementations process. Furthermore, the chapter ends by discussing future avenues of research aimed at improving the proposed frameworks.

7.1 Summary

The goal of this thesis was to examine how blockchain technology could be used to share personal health information between entities (nature person or organisations) within and across domains. The thesis achieved five objectives to accomplish the goal. Objective 1 was to identify the main requirements for the secure exchange of PHI. This objective was accomplished in chapter 3. Objective 2 was to explore the employment of blockchain technology in PHI exchange that was met in chapter 4. Objective 3 was to design and construct blockchain-enabled frameworks for PHI exchange between different entities located in a single or multiple jurisdiction. To achieve this, chapter 5 was shown three blockchain-based frameworks, named AudiWFlow, BRUE and BRESPE, that designed for meeting different security requirements. Objective 4 aimed to implement and evaluate proof-of-concept systems of three proposed frameworks that was presented chapter 6. Finally, objective 5 was to summarise and make recommendations about all related work. The following paragraphs briefly summarise the chapters in the thesis.

Chapter 1 outlined the entire research work of this thesis. It highlighted the aim and objectives, research domains and questions, and the main contributions of this thesis. The research domain was defined as an intersection of blockchain technology, health information, and data sharing. The main contributions of the research can be categorised into two types. Firstly, the thesis provided theoretical support for secure PHI exchange by exploring the requirements of PHI exchange and challenges associated with adopting blockchain technology in PHI exchange. This theoretical support lays the groundwork to design and construct a more robust and secure PHI exchange framework. Secondly, the thesis proposed three new blockchain-enabled frameworks to support PHI exchange within and across regions. The first framework, AudiWFlow, is designed for PHI exchange between different entities located in a single jurisdiction. The second framework, BRUE, is specifically tailored for PHI exchange across jurisdictions. The third framework, BRESPE is designed to exchange PHI between different entities improving privacy preservation.

Chapter 2 introduced a methodological method to support the whole research including processes exploratory descriptive study, technology assessment, design science research, proof-of-concept implementation and evaluation. It also reviewed all related research methods used in the thesis. In the data collection, associated with objective 1, the process of exploratory descriptive study selects research methods of document analysis and literature review to define the requirements of PHI exchange. Regarding objective 2, the process of technology assessment refers to literature review for exploring blockchain employment in PHI exchange. As for objective 3, the process of design science research involves the method prototyping to design and construct the blockchain-enabled frameworks. The process of proof-of-concept implementation and evaluation employs research methods of functional test, quantitative metrics and comparative analysis to achieve objective 4. Besides, this chapter explains selected technologies used in the frameworks design and implementations. The design of all proposed frameworks involves Ethereum, smart contracts and truffle suite. The implementations of them involves JAVA language. Both BRUE and BRESPE frameworks applied UMA and PDR architectures to define the scope of roles and concepts.

Chapter 3 investigated the requirements of PHI exchange. Together with appendix A, this chapter provided background information on PHI and described the relevant regulations and rules that govern PHI exchange. Moreover, the chapter reviewed the related

literature on the exchange of data in the healthcare sector and health technology. In the context of data exchange, PHI possesses several characteristics that need to be taken into consideration. Firstly, PHI has high research value as it can provide insight into various health conditions and trends. Secondly, PHI has a rich data format or structure, which may pose challenges for efficient exchange and storage. Thirdly, PHI contains a significant amount of sensitive personal data, which needs to be protected to maintain privacy and security. Fourthly, PHI is fragmented across various data sources, making it challenging to access and exchange. Finally, PHI is often managed in an organisation-centric manner, which may create interoperability issues between different organisations.

Chapter 4 introduced blockchain technology and its applications in PHI exchange. The chapter explained the different types of blockchains, namely, permissionless and permissioned blockchains. It provided an example of a popular permissionless blockchain, Ethereum, which allows public data communication with confidentiality. Smart contracts, which enforce agreements between entities that are run on Ethereum. Then, the chapter reviewed the literature on the application of blockchain technology to PHI exchange. It explored blockchain technology in auditing, and PHI exchange with blockchain-enabled framework design and blockchain-enabled implementation in security and privacy preservation. Besides, the chapter also discussed the challenges of using blockchain technology in PHI exchange, including trust-building barrier, interoperability, security, auditing, and low patient engagement. Addressing these challenges is crucial for ensuring the effectiveness and feasibility of blockchain-based solutions used in PHI exchange.

Chapter 5 proposed three blockchain-enabled frameworks, named AudiWFlow, BRUE, and BRESPE, to target objective 3. The AudiWFlow framework was intended for data exchange in multiple-entities workflows, where it created an audit trail to track and verify data on-the-fly and after the fact. When it comes to sharing PHI, the AudiWFlow framework was designed to exchange data in the health sector between entities located in the same jurisdiction. A blockchain is utilised as an audit server to store and share audit records of the current data transaction between entities. The BRUE framework, on the other hand, was designed for PHI exchange across jurisdictions to manage authorisation of data access between organisations (entities) using lightweight tokens. This framework involves minimal sensitive personal health data in the exchange to ensure privacy preservation and provides PDRs to audit every transaction. The framework processes PHI

between data subject, resource owner, authorisation server, and resource server. Finally, the BRESPE framework was built to improve privacy preservation in PHI exchange using policies that are attached with data records for user preferences and regulations. This framework concerns PHI exchange between data subject, data requester, and data controller. In summary, AudiWFlow serves as the initial framework that uses blockchain to exchange PHI between entities located in a single jurisdiction; BRUE was built for cross-jurisdictional PHI exchange between entities (organisations); and BRESPE was designed to improve data privacy protection in PHI exchange between data subject, data requester, and data controller.

Chapter 6 presented prototype implementations and performance evaluations of three proposed frameworks. All implementations were completed using smart contracts run on Ethereum, and the blockchain part was developed with a Truffle suite. Besides, the chapter evaluated three proposed frameworks prototypes and compared them with four selected existing frameworks. The AudiWFlow framework implementation consisted of two sections: the audit server and the workflow. The audit server and workflow were respectively implemented with Solidity and Java codes. Experiments were conducted on a local private test network with topology up to $N = 20$, and individual audit records up to 10 KB. The experiments showed that the average response time had a stable relationship with the number of records at different graph connectivity levels. The gas used increased linearly as the record size increased. The BRUE and BRESPE frameworks were separately implemented as a web application that provide user interface for users to share data. Both prototype systems have been deployed on the Ganache network and Goerli network with three different sizes of request records. The results indicated that the average gas cost and response time in each complete iteration were stable and increased if the size of the records increased in the Ganache network. In the Goerli network, there were 100 records of requests for each transaction in experiments. The gas usage in ETH of BRUE and BRESPE significantly increased with different size of the records. With smaller individual record sizes, the average gas usage of both implementations was stable up to a certain point. The response time of each iteration of BRUE and BRESPE prototypes has no significant relationship with the number of records. Although the response time varied, the average response time was close to a straight line. Compared the experimental results between BRUE and BRESPE prototypes, the average cost of gas in BRUE was higher

than that of BRESPE, but the average response time in each iteration in BRUE was less than BRESPE.

7.2 Research Limitations

This thesis proposed three blockchain-based frameworks to respectively address different challenges in the health sector, including confidentiality, non-repudiation, auditability, privacy, and compatibility. The AudiWFlow framework facilitates the challenges in PHI exchange between different entities located in a single jurisdiction, BRUE refers to requirements of PHI exchange across jurisdictions, and BRESPE improves privacy preservation in the health data exchange. However, there were some limitations in the framework designs and implementations. Both BRUE and BRESPE are more applicable to specific cases related to health information exchange across organisations and regions, and their results may not be proved a generalisable state of all cases. The following introduces details regarding the limitations of the research work in this thesis.

Methodology Although the chapter 2 outlined the details of a methodological approach used in the thesis that supports all processes of related work, there is necessary to improve the research methods related to the work shown in the chapters 3 and 4 meeting objectives 1 and 2. There is a lack of critical evaluation of the literature on exploring the requirements and challenges of blockchain employed in PHI exchange through deeper criticism and synthesis of existing knowledge. Besides, the discussion of research methods supporting the evaluation of proof-of-concept systems implementations should focus on critical analysis and implications. Last but not the least, the involved technologies need to be updated during the research period. For example, truffle suite was sunset since 2023. Ganache and Goerli test network were both updated during the writing of this thesis and Ganache is also replaced by a new one because Ethereum and its ecosystem has a significant update since 2022.

Framework Design The AudiWFlow framework focuses on PHI exchange between entities located in the same jurisdiction and it allows entities to directly interact and share PHI with each other in the workflow. This direct data exchange between entities is hard to authenticate and audit if the involved entities are located in different jurisdictions

because of regulatory requirements.

In the design of the BRUE and BRESPE frameworks, the focus is primarily on data access control and authorisation management, with less attention given to the final step of actual PHI exchange between data controller and data requester. While these proposed frameworks provide mechanisms for granting permission to access data, they lack details regarding the actual data exchange process. Therefore, the research work fails to fully consider the importance of the last step of the exchange, which is also a critical part of the data transaction process.

Furthermore, the thesis does not provide specific details regarding personal identity and cryptographic approaches used in the proposed frameworks. Although the BRUE and BRESPE frameworks incorporate minimal personal identification information in the authorisation management process, the required identity information has not been explicitly defined. Moreover, the cryptographic approaches utilised in the protocols have not been thoroughly explored. While the proposed frameworks apply a particular cryptographic approach for encrypting and decrypting data, other symmetric and asymmetric key cryptography methods should be explored and compared to ensure the most effective security measures are in place.

Implementation and Evaluation There are several limitations in the evaluation of the AudiWFlow, BRUE, and BRESPE frameworks implementations. Although these proof-of-concept systems were tested on a local test network or a Goerli test network, the lack of experiments with actual datasets on a real Ethereum network may limit the persuasiveness of the conclusions drawn. Additionally, the thesis does not integrate prototype systems of these frameworks with an actual EHR system or healthcare database. The thesis does not include enough feedback on transaction latency and throughput. The number of requests in the experiments was limited to 100, it may not provide sufficient statistical significance. Furthermore, the experiments do not simulate multiple data access requests occurring at the same time, which could affect the overall performance of the frameworks.

Regarding the BRESPE framework, while it improves data privacy protection in health information exchange through the use of blockchain technology, personal data receipts, and sticky policy, the current framework does not provide detailed guidance on how to

create effective and user-friend sticky policies based on data protection regulations and user preferences. It is important to note that the implementation of BRESPE is still in the proof-of-concept stage, and further research is necessary to evaluate its effectiveness in real-world scenarios.

Besides, the implementations of proof-of-concept systems has a limit of cost in the deployment if they deploy in the real Ethereum network. In real-world scenarios, the cost of data transactions using these proposed frameworks could be significantly increased, especially given the large amount of data exchanged daily in the healthcare sector. While the experiments conducted in this thesis provide some insight into the gas usage and response time of the proposed frameworks, it is important to consider the potential cost implications of deploying these frameworks on a larger scale.

7.3 Future Work

Blockchain technology is a promising solution for secure data exchange in the healthcare sector. However, to effectively utilise its benefits while overcoming its drawbacks, there still have future research. One key aspect that needs to be addressed in future research does not include sensitive personal information in the identity during data transmission. Anonymous identity can be explored in authorisation management to further enhance privacy protection. Additionally, existing records of permission and sticky policies stored in the blockchain should be reused to reduce duplication records, as these cannot be deleted in Ethereum. A proper method for these records reuses should be developed. Moreover, future research should focus on optimising gas usage and reducing transaction costs in the Ethereum network. These could involve investigating alternative blockchain platforms or implementing more efficient data storage and access control mechanisms. Ultimately, the possibility of using blockchain-based frameworks on a large scale for health information exchange in practice will depend on the potential cost of data transactions.

To improve the existing proposed frameworks, an upgraded framework can be developed by integrating three proposed frameworks to exchange health information. The prototype implementation needs to experiment with a real dataset related to healthcare. The experiment should concentrate on transaction throughput, latency, and concurrently happening multiple requests for data access. The new framework aims to achieve a lower

average response time and cost for a complete iteration of data transmission.

References

Abbas, A. & Khan, S. U. (2014), 'A review on the state-of-the-art privacy-preserving approaches in the e-health clouds', *IEEE journal of Biomedical and health informatics* **18**(4), 1431–1441. https://doi.org/10.1109/JBHI.2014.2300846.

Abdelhamid, M., Gaia, J. & Sanders, G. L. (2017), 'Putting the focus back on the patient: How privacy concerns affect personal health information sharing intentions', *Journal of Medical Internet Research* **19**(9), e169. https://doi.org/10.2196/jmir.6877.

Abreu, P. W., Aparicio, M. & Costa, C. J. (2018), Blockchain technology in the auditing environment, *in* '2018 13th Iberian Conference on Information Systems and Technologies (CISTI)', pp. 1–6. https://doi.org/10.23919/CISTI.2018.8399460.

Ahmad, A., Saad, M., Bassiouni, M. & Mohaisen, A. (2018), Towards blockchain-driven, secure and transparent audit logs, *in* 'Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services', pp. 443–448. https://doi.org/10.1145/3286978.3286985.

Alrebdi, N., Alabdulatif, A., Iwendi, C. & Lian, Z. (2022), 'SVBE: Searchable and verifiable blockchain-based electronic medical records system', *Scientific Reports* **12**, 266. https://doi.org/10.1038/s41598-021-04124-8.

Anderson, J. (2018), 'Securing,standardizing, and simplifying electronic health record audit logs through permissioned blockchain technology'. https://digitalcommons.dartmouth.edu/senior_theses/135/.

Angraal, S., Krumholz, M. H. & Schulz, L. W. (2017), 'Blockchain technology:application in health care', *Circulation: Cardiovascular Quality and Outcomes* **10**(9), e003800. https://doi.org/10.1161/CIRCOUTCOMES.117.003800.

Antipova, T. (2018), Using blockchain technology for government auditing, *in* '2018 13th Iberian Conference on Information Systems and Technologies (CISTI)', pp. 1–6. https://doi.org/10.23919/CISTI.2018.8399439.

Azaria, A., Ekblaw, A., Vieira, T. & Lippman, A. (2016), MedRec: Using blockchain for medical data access and permission management, *in* '2016 2nd International Conference on Open and Big Data (OBD)', pp. 25–30. https://doi.org/10.1109/OBD.2016.11.

Back, A. (1997), 'A partial hash collision based postage scheme', http://www.hashcash.org/papers/announce.txt.

Baran, P. (1964), 'On distributed communications: I. introduction to distributed communications networks memorandum (rm-3420-pr)'. https://www.rand.org/content/dam/rand/pubs/research_memoranda/2006/RM3420.pdf.

BBC NEWS (2017), 'NHS 'could have prevented' wannacry ransomware attack', https://www.bbc.co.uk/news/technology-41753022.

Bentov, I., Lee, C., Mizrahi, A. & Rosenfeld, M. (2014), 'Proof of activity: Extending bitcoin's proof of work via proof of stake [extended abstract]', *ACM SIGMETRICS Performance Evaluation Review* **42**(3), 34–37. https://doi.org/10.1145/2695533.2695545.

Buterin, V. (2014), 'A next-generation smart contract and decentralized application platform', *white paper* **3**(37), 2–1. https://finpedia.vn/wp-content/uploads/2022/02/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf.

Caban, K. (2017), '2017 on track to exceed 2016 trend of 'one health data breach per day", https://blog.protenus.com/2017-on-track-to-exceed-2016-trend-of-one-health-data-breach-per-day.

Casola, V., Castiglione, A., Choo, K.-K. R. & Esposito, C. (2016), 'Healthcare-related data in the cloud: Challenges and opportunities', *IEEE cloud computing* **3**(6), 10–14. https://doi.org/10.1109/MCC.2016.139.

Castaldo, L. & Cinque, V. (2018), Blockchain-based logging for the cross-border exchange of ehealth data in Europe, *in* 'Security in Computer and Information Sciences: First International ISCIS Security Workshop 2018', Springer International Publishing, pp. 46–56. https://doi.org/10.1007/978-3-319-95189-8_5.

Castro, M. & Liskov, B. (1999), Practical byzantine fault tolerance, *in* 'Proceedings of the Third Symposium on Operating Systems Design and Implementation', Vol. 99, pp. 173–186. https://pmg.csail.mit.edu/papers/osdi99.pdf.

Chuen, L. K. D. & Deng, R. (2017), *Handbook of blockchain, digital finance, and inclusion: cryptocurrency, FinTech, InsurTech, regulation, ChinaTech, mobile security, and distributed ledger*, Vol. 2, 1 edn, Academic Press, an imprint of Elsevier, chapter 7.

Clancy, C. M., Anderson, K. M. & White, P. J. (2009), 'Investing in health information infrastructure: Can it help achieve health reform?', *Health Affairs* **28**(2), 478–482. https://doi.org/10.1377/hlthaff.28.2.478.

Collins, K. (2015), 'A quick guide to the worst corporate hack attacks', https://www.bloomberg.com/graphics/2014-data-breaches/.

Consent & Information Sharing Work Group (2018), 'Consent receipt specification', https://kantarainitiative.org/download/7902/.

Crosby, M., Nachiappan, Pattanayak, P., Verma, S. & Kalyanaraman, V. (2016), *Blockchain technology: Beyond bitcoin*, Berkeley Engineer: Sutardia center for Entrepreneurship & Technology. https://scet.berkeley.edu/wp-content/uploads/AIR-2016-Blockchain.pdf.

Cucurull, J. & Puiggalí, J. (2016), Distributed immutabilization of secure logs, *in* 'Security and Trust Management: 12th International Workshop, STM 2016, Heraklion, Crete, Greece, September 26-27, 2016, Proceedings 12', Springer international Publishing, pp. 122–137. https://doi.org/10.1007/978-3-319-46598-2_9.

Diffie, W. & Hellman, M. (1976), 'New directions in cryptography', *IEEE Transactions on Information Theory* **IT-22**(6), 644–654. https://doi.org/10.1109/TIT.1976.1055638.

Ding, M., He, H., Qiao, R. & Zhou, X. (2022), RIPPB: A robust and improved pbft protocol for blockchain, *in* '2022 IEEE 17th Conference on Industrial Electronics and Applications (ICIEA)', pp. 384–389. https://doi.org/10.1109/ICIEA54703.2022.10005892.

Esmaeilzadeh, P. (2019), 'The impacts of the perceived transparency of privacy policies and trust in providers for building trust in health information exchange: empirical study', *JMIR medical informatics* **7**(4), e14050. https://doi.org/10.2196/14050.

Esposito, C., De Santis, A., Tortora, G., Chang, H. & Choo, K.-K. R. (2018), 'Blockchain: A panacea for healthcare cloud-based data security and privacy?', *IEEE Cloud Computing* **5**(1), 31–37. https://doi.org/10.1109/MCC.2018.011791712.

European Commission (2016), 'GDPR'. https://eur-lex.europa.eu/eli/reg/2016/679/2016-05-04.

Fabian, B., Ermakova, T. & Junghanns, P. (2015), 'Collaborative and secure sharing of healthcare data in multi-clouds', *Information Systems* **48**, 132–150. https://doi.org/10.1016/j.is.2014.05.004.

Fan, K., Wang, S., Ren, Y., Li, H. & Yang, Y. (2018), 'MedBlock: Efficient and secure medical data sharing via blockchain', *Journal of Medical Systems* **42**, 1–11. https://doi.org/10.1007/s10916-018-0993-7.

Gai, K., Wu, Y., Zhu, L., Qiu, M. & Shen, M. (2019), 'Privacy-preserving energy trading using consortium blockchain in smart grid', *IEEE Transactions on Industrial Informatics* **15**(6), 3548–3558. https://doi.org/10.1109/TII.2019.2893433.

Gai, K., Wu, Y., Zhu, L., Xu, L. & Zhang, Y. (2019), 'Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks', *IEEE Internet of Things Journal* **6**(5), 7992–8004. https://doi.org/10.1109/JIOT.2019.2904303.

Gilad, Y., Hemo, R., Micali, S., Vlachos, G. & Zeldovich, N. (2017), Algorand: Scaling byzantine agreements for cryptocurrencies, *in* 'Proceedings of the 26th symposium on operating systems principles', pp. 51–68. https://doi.org/10.1145/3132747.3132757.

Guan, Z., Lyu, H., Zheng, H., Li, D. & Liu, J. (2019), Distributed audit system of SDN controller based on blockchain, *in* 'Smart Blockchain 2019, LNCS', Vol. 11911, Springer International Publishing, pp. 21–31. https://doi.org/10.1007/978-3-030-34083-4_3.

Guo, R., Shi, H., Zhao, Q. & Zheng, D. (2018), 'Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems', *IEEE Access* **6**, 11676–11686. https://doi.org/10.1109/ACCESS.2018.2801266.

Haried, P., Claybaugh, C. & Dai, H. (2019), 'Evaluation of health information systems research in information systems research: A meta-analysis', *Health informatics journal* **25**(1), 186–202. https://doi.org/10.1177/1460458217704259.

Hartmann, K. & Steup, C. (2015), 'On the security of international data exchange services for e-governance systems', *Datenschutz und Datensicherheit-DuD* **39**(7), 472–476. https://doi.org/10.1007/s11623-015-0452-2.

He, H., Liu, C., Zhou, X. & Feng, K. (2021), FMSM: A fuzzy multi-keyword search scheme for encrypted cloud data based on multi-chain network, *in* '50th International Conference on Parallel Processing Workshop', pp. 1–8. https://doi.org/10.1145/3458744.3474040.

Health Level Seven International (2011), ' HL7 FHIR', http://hl7.org/fhir/.

Hersh, W. R., Totten, A. M., Eden, K. B., Devine, B., Gorman, P., Kassakian, S. Z., Woods, S. S., Daeges, M., Pappas, M. & McDonagh, M. S. (2015), 'Outcomes from health information exchange: Systematic review and future research needs', *JMIR Medical Informatics* **3**(4), e39. https://doi.org/10.2196/medinform.5215.

Hillestad, R., Bigelow, J., Bower, A., Girosi, F., Meili, R., Scoville, R. & Taylor, R. (2005), 'Can electronic medical record systems transform health care? potential health benefits savings and costs', *Health Affairs* **24**(5), 1103–1117. https://doi.org/10.1377/hlthaff.24.5.1103.

Hölbl, M., Kompara, M., Kamišalić, A. & Nemec Zlatolas, L. (2018), 'A systematic review of the use of blockchain in healthcare', *Symmetry* **10**(10), 470. https://doi.org/10.3390/sym10100470.

Häyrinen, K., Saranto, K. & Nykänen, P. (2008), 'Definition, structure, content, use and impacts of electronic health records: A review of the research literature', *International Journal of Medical Informatics* **77**(5), 291–304. https://doi.org/10.1016/j.ijmedinf.2007.09.001.

IBM Institution for Business Value (2016), 'Healthcare rallies for blockchains: Keeping patients at the center'. https://www.ibm.com/downloads/cas/BBRQK3WY.

Japan Personal Information Protection Commission (2017), 'Amended Act on the Protection of Personal Information (Tentative Translation)'. https://www.ppc.go.jp/files/pdf/Act_on_the_Protection_of_Personal_Information.pdf.

Jesus, V. (2020), 'Towards an accountable web of personal information: the web-of-receipts', *IEEE Access* **8**, 25383–25394. https://doi.org/10.1109/ACCESS.2020.2970270.

Jones, L. A., Nelder, J. R., Fryer, J. M., Alsop, P. H., Geary, M. R., Prince, M. & Cardinal, R. N. (2022), 'Public opinion on sharing data from health services for clinical and research purposes without explicit consent: an anonymous online survey in the UK', *BMJ Open* **12**(4), e057579. http://dx.doi.org/10.1136/bmjopen-2021-057579.

Kaelber, D. C. & Bates, D. W. (2007), 'Health information exchange and patient safety', *Journal of biomedical informatics* **40**(6), S40–S45. https://doi.org/10.1016/j.jbi.2007.08.011.

Karjoth, G., Schunter, M. & Waidner, M. (2003), Platform for enterprise privacy practices: Privacy-enabled management of customer data, *in* 'Lecture notes in computer science', Vol. 2482, Springer, Berlin, Heidelberg, pp. 69–84. https://link.springer.com/content/pdf/10.1007/3-540-36467-6.pdf#page=78.

King, S. & Nadal, S. (2012), 'Ppcoin: Peer-to-peer crypto-currency with proof-of-stake'. https://bitcoin.peryaudo.org/vendor/peercoin-paper.pdf.

Kish, L. J. & Topol, E. J. (2015), 'Unpatients—why patients should own their medical data', *Nature Biotechnology* **33**, 921–924. https://doi.org/10.1038/nbt.3340.

Kokoris Kogias, E., Jovanovic, P., Gailly, N., Khoffi, I., Gasser, L. & Ford, B. (2016), Enhancing bitcoin security and performance with strong consistency via collective signing, *in* 'Proceedings of the 25th USENIX Security Symposium', USENIX Association, pp. 279–296. https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_kokoris-kogias.pdf.

Kuperman, G. J. (2011), 'Health-information exchange: why are we doing it, and what are we doing?', *Journal of the American Medical Informatics Association* **18**(5), 678–682. https://doi.org/10.1136/amiajnl-2010-000021.

Larimer, D. (2014), 'Delegated proof-of-stake (DPOS)', https://how.bitshares.works/en/master/technology/dpos.html.

Lee, S., Kim, J., Kwon, Y., Kim, T. & Cho, S. (2022), 'Privacy preservation in patient information exchange systems based on blockchain: System design study', *Journal of Medical Internet Research* **24**(3), e29108. https://doi.org/10.2196/29108.

Li, M., Yu, S., Zheng, Y., Ren, K. & Lou, W. (2013), 'Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption', *IEEE transactions on parallel and distributed systems* **24**(1), 131–143. https://doi.org/10.1109/TPDS.2012.97.

Lumpkin, R. J., Cohn, S. P. & Blair, J. S. (2000), 'Uniform data standards for patient medical record information', *National Committee on Vital and Health Statistics* **53**. https://www.ncvhs.hhs.gov/wp-content/uploads/2014/08/hipaa000706.pdf.

Machulak, M. & Richer, J. (2016), 'User-managed access (UMA) 2.0', https://docs.kantarainitiative.org/uma/ed/uma-core-2.0-01.html#terminology.

Mackey, T. K., Kuo, T.-T., Gummadi, B., Clauson, K. A., Church, G., Grishin, D., Obbad, K., Barkovich, R. & Palombini, M. (2019), "fit-for-purpose?' - challenges and opportunities for applications of blockchain technology in the future of healthcare', *BMC Medicine* **17**(1), 1–17. https://doi.org/10.1186/s12916-019-1296-7.

Matthews, G. J., Harel, O. & Aseltine, R. H. (2016), 'Privacy protection and aggregate health data: a review of tabular cell suppression methods (not) employed in public

health data systems', *Health Services and Outcomes Research Methodology* **16**, 258–270. https://doi.org/10.1007/s10742-016-0162-8.

Mazieres, D. (2015), 'The stellar consensus protocol: A federated model for internet-level consensus', *Stellar Development Foundation* **32**, 1–45. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=c0f66fd90213631153b8ca2fc22eb327e88bbbe4.

Mazlan, A. A., Mohd Daud, S., Mohd Sam, S., Abas, H., Abdul Rasid, S. Z. & Yusof, M. F. (2020), 'Scalability challenges in healthcare blockchain system—a systematic review', *IEEE Access* **8**, 23663–23673. https://doi.org/10.1109/ACCESS.2020.2969230.

McGhin, T., Choo, K.-K. R., Liu, C. Z. & He, D. (2019), 'Blockchain in healthcare applications: Research challenges and opportunities', *Journal of Network and Computer Applications* **135**, 62–75. https://doi.org/10.1016/j.jnca.2019.02.027.

Mello, M. M., Adler-Milstein, J., Ding, K. L. & Savage, L. (2018), 'Legal barriers to the growth of health information exchange—boulders or pebbles?', *The Milbank Quarterly* **96**(1), 110–143. https://doi.org/10.1111/1468-0009.12313.

Milne, R., Morley, K. I., Almarri, M. A., Anwer, S., Atutornu, J., Baranova, E. E., Bevan, P., Cerezo, M., Cong, Y., Costa, A. et al. (2021), 'Demonstrating trustworthiness when collecting and sharing genomic data: public views across 22 countries', *Genome medicine* **13**(1), 92. https://doi.org/10.1186/s13073-021-00903-0.

Miorandi, D., Rizzardi, A., Sicari, S. & Coen-Porisini, A. (2020), 'Sticky policies: A survey', *IEEE Transactions on Knowledge and Data Engineering* **32**(12), 2481–2499. https://doi.org/10.1109/TKDE.2019.2936353.

Mohey Eldin, A., Hossny, E., Wassif, K. & Omara, F. A. (2023), 'Federated blockchain system (FBS) for the healthcare industry', *Scientific Reports* **13**, 2569. https://doi.org/10.1038/s41598-023-29813-4.

Monga, S. & Singh, D. (2022), 'MRBSChain a novel scalable medical records binance smart chain framework enabling a paradigm shift in medical records management', *Scientific Reports* **12**, 17660. https://doi.org/10.1038/s41598-022-22569-3.

Nakamoto, S. (2008), 'Bitcoin: A peer-to-peer electronic cash system'. https://assets.pubpub.org/d8wct41f/31611263538139.pdf.

Nasir, M. H., Arshad, J., Khan, M. M., Fatima, M., Salah, K. & Jayaraman, R. (2022), 'Scalable blockchains—a systematic review', *Future Generation Computer Systems* **126**, 136–162. https://doi.org/10.1016/j.future.2021.07.035.

National People's Congress Standing Committee of China (2016), 'Cyber Security Law of the People's Republic of China', https://flk.npc.gov.cn/detail2.html?MmM5MDlmZGQ2NzhiZjE3OTAxNjc4YmY4Mjc2ZjA5M2Q%3D, English Translation: https://digichina.stanford.edu/work/translation-cybersecurity-law-of-the-peoples-republic-of-china-effective-june-1-2017/.

National People's Congress Standing Committee of China (2021), 'Personal Information Protection Law of the People's Republic of China'. https://flk.npc.gov.cn/detail2.html?ZmY4MDgxODE3YjY0NzJhMzAxN2I2NTZjYzIwNDAwNDQ%3D, English Translation: https://digichina.stanford.edu/work/translation-personal-information-protection-law-of-the-peoples-republic-of-china-effective-nov-1-2021/.

Nehme, A., Jesus, V., Mahbub, K. & Abdallah, A. (2019), Decentralised and collaborative auditing of workflows, *in* 'Trust, Privacy and Security in Digital Business: 16th International Conference, TrustBus 2019, Linz, Austria, August 26–29, 2019, Proceedings 16', pp. 129–144. https://doi.org/10.1007/978-3-030-27813-7_9.

NHS (2019), 'The NHS long term plan', https://www.longtermplan.nhs.uk/wp-content/uploads/2019/08/nhs-long-term-plan-version-1.2.pdf.

Nofer, M., Gomber, P., Hinz, O. & Schiereck, D. (2017), 'Blockchain', *Business & Information Systems Engineering* **59**(3), 183–187. https://doi.org/10.1007/s12599-017-0467-3.

Nxt Community (2014), 'Nxt whitepaper'. https://www.jelurida.com/sites/default/files/NxtWhitepaper.pdf.

O'Donoghue, O., Vazirani, A. A., Brindley, D. & Meinert, E. (2019), 'Design choices and

trade-offs in health care blockchain implementations: systematic review', *Journal of Medical Internet Research* **21**(5), e12426. https://doi.org/10.2196/12426.

ONC (2015), 'Report on health information blocking'. https://www.healthit.gov/sites/default/files/reports/info_blocking_040915.pdf.

Pappel, I., Pappel, I., Tepandi, J. & Draheim, D. (2017), Systematic digital signing in Estonian e-government processes, *in* 'Transactions on large-scale data-and knowledge-centered systems XXXVI: Special Issue on Data and Security Engineering', Springer, Berlin, Heidelberg, pp. 31–51. https://doi.org/10.1007/978-3-662-56266-6_2.

Peter, S., Jan, W., Douglas, J., Eric, P., Julia, A.-M., Middleton, B. & Bates, D. W. (2007), 'The economic benefits of health information exchange interoperability for Australia', *Australian Health Review* **31**(4), 531–539. https://doi.org/10.1071/AH070531.

Pourmajidi, W. & Miranskyy, A. (2018), Logchain: Blockchain-assisted log storage, *in* '2018 IEEE 11th International Conference on Cloud Computing (CLOUD)', pp. 978–982. https://doi.org/10.1109/CLOUD.2018.00150.

Putz, B., Menges, F. & Pernul, G. (2019), 'A secure and auditable logging infrastructure based on a permissioned blockchain', *Computers & Security* **87**, 101602. https://doi.org/10.1016/j.cose.2019.101602.

PwC Health Research Institution (2018), 'Global top health industry issues: Defining the healthcare of the future'. https://www.pwc.com/gx/en/healthcare/pdf/global-top-health-industry-issues-2018-pwc.pdf.

Reddick, C. & Anthopoulos, L. (2014), 'Interactions with e-government, new digital media and traditional channel choices: citizen-initiated factors', *Transforming Government: People, Process and Policy* **8**(3), 398–419. https://doi.org/10.1108/TG-01-2014-0001.

Schabetsberger, T., Ammenwerth, E., Andreatta, S., Gratl, G., Haux, R., Lechleitner, G., Schindelwig, K., Stark, C., Vogl, R., Wilhelmy, I. et al. (2006), 'From a paper-based transmission of discharge summaries to electronic communication in health care

regions', *International Journal of Medical Informatics* **75**(3), 209–215. https://doi.org/10.1016/j.ijmedinf.2005.07.018.

Shamir, A. (1979), 'How to share a secret', *Communications of the ACM* **22**(11), 612–613. https://doi.org/10.1145/359168.359176.

Snell, E. (2017), 'Healthcare data breach costs highest for 7th straight year', https://healthitsecurity.com/news/healthcare-data-breach-costs-highest-for-7th-straight-year.

Steward, M. (2005), 'Electronic medical records', *Journal of Legal Medicine* **26**(4), 491–506. https://doi.org/10.1080/01947640500364762.

Suzuki, S. & Murai, J. (2017), Blockchain as an audit-able communication channel, *in* '2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)', Vol. 2, pp. 516–522. https://doi.org/10.1109/COMPSAC.2017.72.

Szabo, N. (1994), 'Smart contracts', https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html.

Szabo, N. (1997*a*), 'Formalizing and securing relationships on public networks', *First Monday* **2**(9). https://firstmonday.org/ojs/index.php/fm/article/view/548/469.

Szabo, N. (1997*b*), 'The idea of smart contracts'. https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/idea.html.

Tapas, N., Merlino, G., Longo, F. & Puliafito, A. (2019), Blockchain-based publicly verifiable cloud storage, *in* '2019 IEEE International Conference on Smart Computing (SMARTCOMP)', pp. 381–386. https://doi.org/10.1109/SMARTCOMP.2019.00076.

The World Economic Forum's Global Agenda Council on the Future of Software & Society (2015), 'Deep shift: Technology tipping points and societal impact'. http://www3.weforum.org/docs/WEF_GAC15_Technological_Tipping_Points_report_2015.pdf.

Tian, F. (2017), A supply chain traceability system for food safety based on HACCP, blockchain & internet of things, *in* '2017 International Conference on Service Sys-

tems and Service Management', pp. 1–6. https://doi.org/10.1109/ICSSSM.2017.7996119.

UK Digital (2017), 'New cyber security service to boost NHS protection', https://digital.nhs.uk/services/data-security-centre/data-security-centre-latest-news/new-cyber-security-service-to-boost-nhs-protection.

UK Digital (2018), 'Data Security and Protection Toolkit', https://digital.nhs.uk/data-and-information/looking-after-information/data-security-and-information-governance/data-security-and-protection-toolkit.

U.S. Department of Health and Human Services (2002), 'Standards for Privacy of Individually Identifiable Health Information', https://www.federalregister.gov/documents/2002/08/14/02-20554/standards-for-privacy-of-individually-identifiable-health-information.

USA Department of Health and Human Services (2018), 'Health industry cybersecurity practices: managing threats and protecting patients', https://405d.hhs.gov/Documents/HICP-Main-508.pdf.

Vazirani, A. A., O'Donoghue, O., Brindley, D. & Meinert, E. (2020), 'Blockchain vehicles for efficient medical record management', *NPJ digital medicine* **3**, 1. https://doi.org/10.1038/s41746-019-0211-0.

Verizon (2017), 'Data breach investigations report 10th edition'. https://www.ictsecuritymagazine.com/wp-content/uploads/2017-Data-Breach-Investigations-Report.pdf.

Vest, J. R. & Gamm, L. D. (2010), 'Health information exchange: persistent challenges and new strategies', *Journal of the American Medical Informatics Association* **17**(3), 288–294. https://doi.org/10.1136/jamia.2010.003673.

Wang, W., Hoang, D. T., Hu, P., Xiong, Z., Niyato, D., Wang, P., Wen, Y. & Kim, D. I. (2019), 'A survey on consensus mechanisms and mining strategy management in blockchain networks', *IEEE Access* **7**, 22328–22370. https://doi.org/10.1109/ACCESS.2019.2896108.

Wang, Y., Li, P.-F., Tian, Y., Ren, J.-J. & Li, J.-S. (2017), 'A shared decision-making system for diabetes medication choice utilizing electronic health record data', *IEEE Journal of Biomedical and Health Informatics* **21**(5), 1280–1287. https://doi.org/10.1109/JBHI.2016.2614991.

Weber, I., Xu, X., Riveret, R., Governatori, G., Ponomarev, A. & Mendling, J. (2016), Untrusted business process monitoring and execution using blockchain, *in* 'Business Process Management', Springer, pp. 329–347. https://doi.org/10.1007/978-3-319-45348-4_19.

Wohrer, M. & Zdun, U. (2018), Smart contracts: security patterns in the ethereum ecosystem and solidity, *in* '2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)', pp. 2–8. https://doi.org/10.1109/IWBOSE.2018.8327565.

Wüst, K. & Gervais, A. (2018), Do you need a blockchain?, *in* '2018 Crypto Valley Conference on Blockchain Technology (CVCBT)', pp. 45–54. https://doi.org/10.1109/CVCBT.2018.00011.

Xia, Q., Sifah, E. B., Asamoah, K. O., Gao, J., Du, X. & Guizani, M. (2017), 'MeDShare: Trust-less medical data sharing among cloud service providers via blockchain', *IEEE Access* **5**, 14757–14767. https://doi.org/10.1109/ACCESS.2017.2730843.

Yeung, K. (2021), 'The health care sector's experience of blockchain: a cross-disciplinary investigation of its real transformative potential', *Journal of Medical Internet Research* **23**(12), e24109. https://doi.org/10.2196/24109.

Zawoad, S., Dutta, A. k. & Hasan, R. (2016), 'Towards building forensics enabled cloud through secure logging-as-a-service', *IEEE Transactions on Dependable and Secure Computing* **13**(2), 148–162. https://doi.org/10.1109/TDSC.2015.2482484.

Zhang, A. & Lin, X. (2018), 'Towards secure and privacy-preserving data sharing in e-health systems via consortium blockchain', *Journal of Medical Systems* **42**(140). https://doi.org/10.1007/s10916-018-0995-5.

Zhang, L., Wang, H., Li, Q., Zhao, M.-H. & Zhan, Q.-M. (2018), 'Big data and medical research in china', *BMJ* **360**. https://doi.org/10.1136/bmj.j5910.

Zhang, P., Schmidt, D. C., White, J. & Lenz, G. (2018), Chapter one - blockchain technology use cases in healthcare, *in* P. Raj & G. C. Deka, eds, 'Blockchain Technology: Platforms, Tools and Use Cases', Vol. 111 of *Advances in Computers*, Elsevier, pp. 1 – 41. https://doi.org/10.1016/bs.adcom.2018.03.006.

Zhang, P., Walker, M. A., White, J., Schmidt, D. C. & Lenz, G. (2017), Metrics for assessing blockchain-based healthcare decentralized apps, *in* '2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)', pp. 1–4. https://doi.org/10.1109/HealthCom.2017.8210842.

Zhang, P., White, J., Schmidt, D. C., Lenz, G. & Rosenbloom, S. T. (2018), 'FHIRChain: Applying blockchain to securely and scalably share clinical data', *Computational and Structural Biotechnology Journal* **16**, 267 – 278. https://doi.org/10.1016/j.csbj.2018.07.004.

Zheng, Z., Xie, S., Dai, H., Chen, X. & Wang, H. (2017), An overview of blockchain technology: Architecture, consensus, and future trends, *in* '2017 IEEE international congress on big data', pp. 557–564. https://doi.org/10.1109/BigDataCongress.2017.85.

Zhou, X., Jesus, V., Wang, Y. & Josephs, M. (2020), User-controlled, auditable, cross-jurisdiction sharing of healthcare data mediated by a public blockchain, *in* '2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)', IEEE, pp. 87–96. https://doi.org/10.1109/TrustCom50675.2020.00025.

Zhou, X., Nehme, A., Jesus, V., Wang, Y., Josephs, M. & Mahbub, K. (2019), Towards blockchain-based auditing of data exchanges, *in* 'Smart Blockchain: Second International Conference, SmartBlock 2019, Birmingham, UK, October 11–13, 2019, Proceedings 2', Springer International publishing, pp. 43–52. https://doi.org/10.1007/978-3-030-34083-4_5.

Zhou, X., Nehme, A., Jesus, V., Wang, Y., Josephs, M., Mahbub, K. & Abdallah, A. (2022), 'AudiWFlow: Confidential, collusion-resistant auditing of distributed workflows', *Blockchain: Research and Applications* **3**(3), 100073. https://doi.org/10.1016/j.bcra.2022.100073.

Appendix A: Legal Definitions

This appendix lists some legal definitions from GDPR, Personal Information Protection Law of the People's Republic of China, Act on the Protection of Personal Information of Japan, and HIPAA. Their sources are as follows:

Section 1: https://eur-lex.europa.eu/eli/reg/2016/679/2016-05-04

Section 2: https://digichina.stanford.edu/work/translation-personal-information-protection-law-of-the-peoples-republic-of-china-effective-nov-1-2021/

Section 3: https://www.ppc.go.jp/files/pdf/Act_on_the_Protection_of_Personal_Information.pdf

Section 4: https://aspe.hhs.gov/reports/health-insurance-portability-accountability-act-1996

A.1 GDPR

The data subject has rights to the following: transparency and modalities; information and access to personal data; rectification and erasure; and the right to object and automated individual decision-making.

- 'Personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person.

- 'Controller' means the natural or legal person, public authority, agency or other body which, alone or jointly with others, determines the purposes and means of the

processing of personal data; where the purposes and means of such processing are determined by Union or Member State law, the controller or the specific criteria for its nomination may be provided for by Union or Member State law.

- 'Processor' means a natural or legal person, public authority, agency or other body which processes personal data on behalf of the controller.

- 'Recipient' means a natural or legal person, public authority, agency or another body, to which the personal data are disclosed, whether a third party or not. However, public authorities which may receive personal data in the framework of a particular inquiry in accordance with Union or Member State law shall not be regarded as recipients; the processing of those data by those public authorities shall be in compliance with the applicable data protection rules according to the purposes of the processing.

- 'Third party' means a natural or legal person, public authority, agency or body other than the data subject, controller, processor and persons who, under the direct authority of the controller or processor, are authorised to process personal data.

- 'Data concerning health' means personal data related to the physical or mental health of a natural person, including the provision of health care services, which reveal information about his or her health status.

- Personal data concerning health should include all data pertaining to the health status of a data subject which reveal information relating to the past, current or future physical or mental health status of the data subject. This includes information about the natural person collected in the course of the registration for, or the provision of, health care services as referred to in Directive 2011/24/EU of the European Parliament and of the Council to that natural person; a number, symbol or particular assigned to a natural person to uniquely identify the natural person for health purposes; information derived from the testing or examination of a body part or bodily substance, including from genetic data and biological samples; and any information on, for example, a disease, disability, disease risk, medical history, clinical treatment or the physiological or biomedical state of the data subject independent of its source, for example from a physician or other health professional, a

hospital, a medical device or an in vitro diagnostic test.

A.2 Personal Information Protection Law of the People's Republic of China

Individuals have the right to know and the right to decide relating to their personal information, and have the right to limit or refuse the handling of their personal information by others, unless laws or administrative regulations stipulate otherwise.

- Personal information is all kinds of information, recorded by electronic or other means, related to identified or identifiable natural persons, not including information after anonymization handling. Personal information handling includes personal information collection, storage, use, processing, transmission, provision, disclosure, deletion, etc.

- Sensitive personal information means personal information that, once leaked or illegally used, may easily cause harm to the dignity of natural persons grave harm to personal or property security, including information on biometric characteristics, religious beliefs, specially-designated status, medical health, financial accounts, individual location tracking, etc., as well as the personal information of minors under the age of 14. Only where there is a specific purpose and a need to fulfill, and under circumstances of strict protection measures, may personal information handlers handle sensitive personal information.

- "Personal information handler" refers to organisations and individuals that, in personal information handling activities, autonomously decide handling purposes and handling methods.

A.3 Act on the Protection of Personal Information of Japan

- A "principal" in relation to personal information in this Act means a specific individual identifiable by personal information.

- "Personal data" in this Act means personal information constituting a personal information database etc.

- "Personal information" in this Act means that information relating to a living individual which falls under any of each following item: (i) those containing a name, date of birth, or other descriptions etc. (meaning any and all matters (excluding an individual identification code) stated, recorded or otherwise expressed using voice, movement or other methods in a document, drawing or electromagnetic record (meaning a record kept in an electromagnetic form (meaning an electronic, magnetic, or other forms that cannot be recognised through the human senses; the same shall apply in the succeeding paragraph, item (ii)); ; hereinafter the same) whereby a specific individual can be identified (including those which can be readily collated with other information and thereby identify a specific individual); (ii) those containing an individual identification code.

- "Special care-required personal information" in this Act means personal information comprising a principal's race, creed, social status, medical history, criminal record, fact of having suffered damage by a crime, or other descriptions etc.

- A "personal information handling business operator" in this Act means a person providing a personal information database etc. for use in business; however, excluding a person set forth in the following: a central government organisation; a local government; an incorporated administrative agency etc; a local incorporated administrative agency.

A.4 HIPAA

Individuals have the right to the following: ask to see and get a copy of personal health records; have corrections added to personal health information; to be noticed about how personal health information may be used and shared; decide if gives permission before personal health information can be used or shared for certain purposes; request that a covered entity restrict how it uses or discloses personal health information; get a report on when and why personal health information was shared for certain purposes; complain with the service provider or health insurer and HHS if the rights are being denied or

personal health information is not being protected.

- Health information means any information, whether oral or recorded in any form or medium, that: is created or received by a health care provider, health plan, public health authority, employer, life insurer, school or university, or health care clearinghouse; and relates to the past, present, or future physical or mental health or condition of an individual, the provision of health care to an individual, or the past, present, or future payment for the provision of health care to an individual.

- Covered entities include health plans, most health care providers, and health care clearinghouses.

- Health plan means an individual or group plan that provides, or pays the cost of, medical care.

- Most health care provider includes a provider of services, a provider of medical or other health services, and any other person furnishing health care services or supplies.

- Health care clearinghouse means a public or private entity that processes or facilitates the processing of nonstandard data elements of health information into standard data elements.

Appendix B: Cryptographic Methods of Secret Sharing and Key Exchange

We use Shamir's secret sharing and Diffie-Hellman key exchange schemes in the proposed frameworks. This appendix briefly introduces both schemes, as described in (Shamir 1979) and (Diffie & Hellman 1976).

B.1 Shamir's Secret Sharing

Shamir's secret sharing is used to secure a secret in a distributed form, most often to secure encryption keys. The secret is split into several shares, which individually do not give any information about the secret. There is required to have a number of shares to reconstruct a secret using Shamir's secret sharing. This amount of shares is called the threshold. No information about the secret can be gained from any number of shares below than the threshold.

This mechanism aims to divide a secret S into n pieces S_1, ..., S_n in such a way that:

1. Knowledge of any k or more S_i pieces makes S easily computable. Any combination of k shares can reconstruct the entire secret S.

2. Knowledge of any k-1 or fewer S_i pieces leaves S completely undetermined (in the sense that all its possible values are equally likely).

This (k, n) scheme is called the threshold scheme.

If $n = k$, all of the pieces are needed to reconstruct the secret S. By using this scheme with n = 2k - 1 there is a very robust key management scheme: We can recover the

original key even when $[n/2]= k$ - 1 of the n pieces are destroyed, but our opponents cannot reconstruct the key even when security breaches expose $[n/2] = k$ - 1 of the remaining k pieces. The scheme is suited to applications in which a group of mutually suspicious individuals with conflicting interests must cooperate.

B.2 Diffie-Hellman Key Exchange

Diffie–Hellman key exchange allows two parties who have not previously met to securely establish a key which they can use to secure their secret communication for exchanging data over a public network. It is a mathematical method of securely exchanging cryptographic keys over a public channel.

The simplest and the original implementation of the Diffie-Hellman key exchange algorithm uses the multiplicative group of integers modulo p, where p is prime, and g is a primitive root modulo p. These two values are chosen in this way to ensure that the resulting shared secret can take on any value from 1 to p–1. Here is an example below. g is a public base known to Alice and Bob. p is a public modulus known to Alice and Bob. a is Alice's private key known only to Alice. b is Bob's private key known only to Bob. A is Alice's public key known to Alice and Bob. B is Bob's public key known to Alice and Bob. s is the shared secret key and it is known to Alice and Bob.

1. Alice and Bob publicly agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).

2. Alice chooses a secret integer $a = 4$, then sends Bob $A = g^a \bmod p$. $A = 5^4 \bmod 23 = 4$ (in this example both A and a have the same value 4, but this is usually not the case).

3. Bob chooses a secret integer $b = 3$, then sends Alice $B = g^b \bmod p$. $B = 5^3 \bmod 23 = 10$.

4. Alice computes $s = B^a \bmod p$. $s = 10^4 \bmod 23 = 18$.

5. Bob computes $s = A^b \bmod p$. $s = 4^3 \bmod 23 = 18$.

6. Alice and Bob now share a secret (the number 18).

Both Alice and Bob have arrived at the same values because under mod p, $A^b \bmod p = g^{ab} \bmod p = g^{ba} \bmod p = B^a \bmod p$.

More specifically, $(g^a \bmod p)^b \bmod p = (g^b \bmod p)^a \bmod p$.

Appendix C: User-Managed Access

UMA is an award-winning OAuth-based federated authorisation standard protocol that helps individuals manage third-party access to their data, content, and service resources across different identity and resource ecosystems. It was introduced by the Kantara Initiative. The first version, UMA 1.0, was released in 2015. The Kantara Initiative officially announced the approval and publication of UMA Version 2.0's technical specifications in February 2018. UMA 2.0 was designed to be closely associated with the well-known OAuth protocol, making it easier to implement while improving its security. The following describes its roles and key concepts, as can be found in `https://docs.kantarainitiative.org/uma/ed/uma-core-2.0-01.html#terminology`.

C.1 Roles

- Resource owner. An entity capable of granting access to a protected resource, the "user" in User-Managed Access. The resource owner may be an end-user (natural person) or may be a non-human entity treated as a person for limited legal purposes (legal person), such as a corporation.

- Requesting party. A natural or legal person that uses a client to seek access to a protected resource. The requesting party may or may not be the same party as the resource owner.

- Resource server. A server that hosts resources on a resource owner's behalf and is capable of accepting and responding to requests for protected resources.

- Authorisation server. A server that protects, on a resource owner's behalf, resources hosted at a resource server.

- Client. An application that is capable of making requests for protected resources with the resource owner's authorisation and on the requesting party's behalf.

The manner in which the resource owner manages resources at the resource server and how policies are defined at the authorisation server are out of the scope of the UMA specification.

C.2 Key Concepts

- Requesting party token (RPT). An OAuth access token associated with the UMA grant. An RPT is specific to five given entities, namely, requesting party, client, authorisation server, resource server, and resource owner.

- Permission. Authorised access to a particular resource with some number of scopes bound to that resource. A permission ticket represents some number of requested permissions. An RPT represents some number of granted permissions. Permissions are part of the authorisation server's process and are opaque to the client.

- Permission ticket. A correlation handle representing requested permissions that is created and maintained by the authorisation server, initially passed to the client by the resource server, and presented by the client at the token endpoint and during requesting party redirects.

- Authorisation process. The process through which the authorisation server determines whether it should issue an RPT to the client on the requesting party's behalf, is based on a variety of inputs. A key component of the process is authorisation assessment.

- Claim. A statement of the value or values of one or more attributes of an entity. The authorisation server typically needs to collect and assess one or more claims of the requesting party or client against policy conditions as part of protecting a resource. The two methods available for UMA claims collection are claims pushing and interactive claims gathering. Note: Claims collection might involve authentication for unique user identification, but depending on policy conditions might additionally or

instead involve the collection of non-uniquely identifying attributes, authorisation for some actions, or other statements of agreement.

- Token. A packaged collection of data meant to be transmitted to another entity. A token could be used for authorised access (an "access token"), or could be used to exchange information about a subject (a "claim token").

- Claim token. A package of claims is provided directly by the client to the authorisation server through claims pushing.

- Persisted claims token (PCT). A correlation handle issued by an authorisation server that represents a set of claims collected during one authorisation process, available for a client to use in attempting to optimise a future authorisation process.

- Protection API access token (PAT). An OAuth access token with the scope uma_protection, used by the resource server at the protection API, consisting of the resource set registration, permission registration, and token introspection endpoints.

- Authorisation API token (AAT). An OAuth 2.0 token with a scope of uma_authorization, used by the client at the authorisation API. It enables a client application to query the server for user permissions.

- Saved consent token (SCT). A correlation handle that is conveyed from an authorisation server to a client and optionally returned by the client to that authorisation server representing an end-user requesting party's consent to subsequent direct client engagement in the trust elevation process.

UMA defines a workflow (Figure C.1) to allow resource owners to manage access to protected resources using authorisation policies on a centralised authorisation server. UMA workflow uses the following actions in the figure:

- Manage. The resource owner manages their resources on the resource server.

- Protect. The resource owner links their resource server and chosen authorisation server. The authorisation server provides a protection API so that the resource server can register sets of resources. Use of the protection API requires a protection API token.

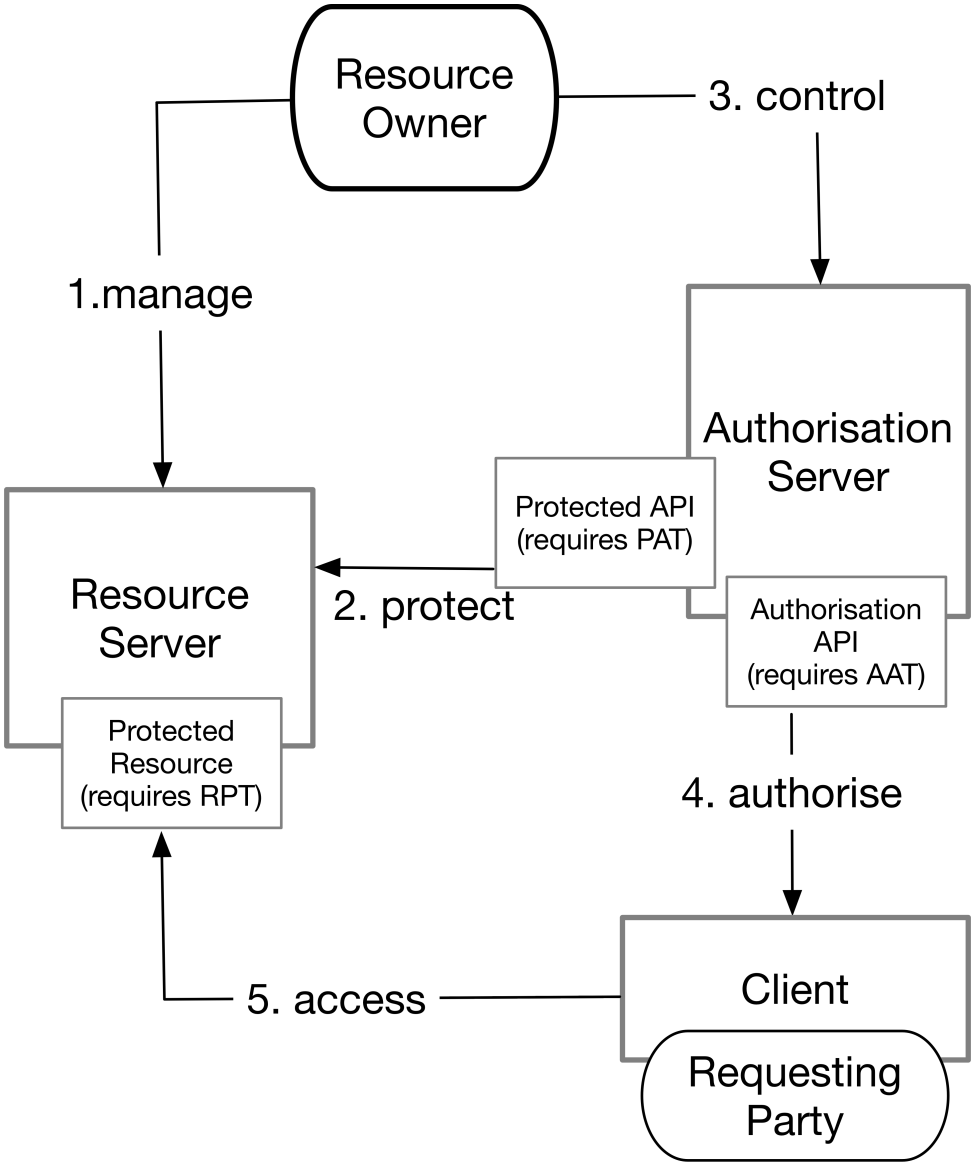


Figure C.1: UMA workflow

- Control. The resource owner controls who has access to their registered resources by creating policies on the authorisation server.

- Authorise. The resource owner controls who has access to their registered resources by creating policies on the authorisation server.

- Access. The client, acting on behalf of the requesting party, uses the authorisation server's authorisation API to acquire a requesting party token. The requesting party or client may need further interaction with the authorisation server at this point, for example, to supply identity claims. Use of the authorisation API requires an authorisation API token.