UNIVERSIDADE D
COIMBRA

Daniel André Areal Vasconcelos

DISTRIBUTED REPUTATION SYSTEMS AND BLOCKCHAIN DATA STORAGE

Dissertation in the context of the Master in Informatics Security, advised by Prof. Bruno Sousa and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September of 2023

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTMENT OF INFORMATICS ENGINEERING

Daniel André Areal Vasconcelos

Distributed Reputation Systems and Blockchain Data Storage

Dissertation in the context of the Master in Informatics Security, advised by Prof. Bruno Sousa and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September of 2023

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Daniel André Areal Vasconcelos

Sistemas de Reputação Distribuídos e Armazenamento da Informação em Blockchain

Dissertação no âmbito do Mestrado em Segurança Informática, orientada pelo Professor Doutor Bruno Sousa e apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro, 2023

Acknowledgements

This work was supported by the project ARCADIAN-IoT project, which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101020259.

Abstract

Blockchain technology is being widely adopted in various industries and environments due to its unique characteristics that allow for integrity, immutability and traceability in a decentralized network. Cryptographic mechanisms are also used and provide non-repudiation, authenticity and confidentiality to all involved parties so that trust can be enforced without the need for a controlling authority.

When the General Data Protection Regulation (GDPR) was put into effect, it completely changed the paradigm of personal data storing and processing; and even though Blockchain characteristics might have a positive impact on bookkeeping and audit, there are some issues when planning to use this technology with personal data.

In this work, the conflicts between GDPR and Blockchain will be analysed and discussed. The available literature is also studied to find solutions for existing issues so that the main goal can be accomplished: to develop a Blockchain-based framework that can handle personal data and is compliant with the GDPR.

As a result of this investigation, we will propose BlockPrivGDPR platform, that is based on the found mechanisms, and also on new suggested ones, that will be designed and implemented using Hyperledger Fabric Blockchain.

This work is part of the ARCADIAN-IoT project, and it integrates its reputation system, where a permissioned blockchain is used to store the values used to calculate the involved entities' reputation score.

Keywords

Blockchain, General Data Protection Regulation, GDPR, ARCADIAN-IoT, Permissioned Blockchain, Off-chain storage

Resumo

A adoção de tecnologias Blockchain está a crescer em várias indústrias e ambientes, devido às suas características únicas que garantem integridade, imutabilidade e rastreabilidade numa rede descentralizada. São ainda utilizados mecanismos criptográficos que oferecem garantias de não-repúdio, autenticidade e confidencialidade. Tudo isto contribui para a garantia de confiança entre todas as partes envolvidas, sem a necessidade da existência de uma autoridade controladora.

Quando o Regulamento Geral da Proteção de Dados (RGPD) entrou em vigor, alterou completamente o paradigma de processamento e armazenamento de dados pessoais. Apesar da utilização da Blockchain poder representar um impacto muito positivo em termos de registo e auditorias, existem alguns problemas na utilização desta tecnologia para tratar dados pessoais.

Neste trabalho serão analisados e discutidos os conflitos entre Blockchain e o RGPD. De seguida, são procuradas soluções para as questões levantadas na literatura disponível, de forma ao principal objetivo ser atingido: o desenvolvimento de uma plataforma baseada em Blockchain que possa tratar dados pessoais e seja complacente com o RGPD.

Como resultado do trabalho será proposta a plataforma BlockPrivGDPR, que se baseia nos mecanismos encontrados, bem como novos mecanismos sugeridos, que será desenhada e implementada em Hyperledger Fabric.

Este trabalho faz parte do projeto ARCADIAN-IoT, e integra o seu sistema de reputação, onde uma Blockchain permissionada é utilizada para guardar os valores usados para o cálculo da reputação das entidades envolvidas.

Palavras-Chave

Blockchain, Regulamento Geral da Proteção de Dados, RGPD, ARCADIAN-IoT, Blockchain permissionada, Armazenamento off-chain

Contents

1 Introduction **1**
- 1.1 Objectives . . . 1
- 1.2 Contributions . . . 2
- 1.3 Structure . . . 2

2 Background and State of The Art **5**
- 2.1 Background . . . 5
  - 2.1.1 General Data Protection Regulation . . . 5
  - 2.1.2 Blockchain . . . 11
  - 2.1.3 GDPR vs Blockchain . . . 13
  - 2.1.4 Support and Related Tools . . . 16
- 2.2 State of the art . . . 17
  - 2.2.1 Blockchain vs GDPR . . . 17
  - 2.2.2 Blockchain Implementations . . . 22
  - 2.2.3 Summary . . . 25

3 BlockPrivGPDR high-level overview and requirements **27**
- 3.1 ARCADIAN-IoT . . . 27
- 3.2 BlockPrivGDPR High-Level Architecture . . . 28
- 3.3 Framing and requirements . . . 29
  - 3.3.1 Functional Requirements . . . 29
  - 3.3.2 Interactions and communication . . . 32

4 Research Objectives and Approach **33**
- 4.1 Research Methodology . . . 33
- 4.2 Objectives . . . 34
- 4.3 Approach . . . 34
- 4.4 Risk Analysis . . . 35

5 Analysis of Hyperledger Fabric **37**
- 5.1 Definitions . . . 37
- 5.2 Transactions . . . 38
  - 5.2.1 Transactions in Hyperledger Fabric . . . 38
  - 5.2.2 Consensus and data transmission . . . 39
  - 5.2.3 Privacy concerns and privacy-preserving mechanisms . . . 40

6 BlockPrivGDPR: Analysis, Architecture and Implementation **45**
- 6.1 Data privacy analysis . . . 45
- 6.2 BlockPrivGDPR: Practical Aspects . . . 46

6.2.1 Introduction . . . 46
6.2.2 Decision-making and compromises . . . 47
6.2.3 Architecture . . . 49
6.2.4 Gateway specification . . . 50
6.2.5 Smart Contract definition and considerations . . . 51
6.2.6 Other components . . . 54
6.2.7 Actions and outcomes . . . 56

7 Setup and Evaluation **59**
7.1 System setup . . . 59
7.1.1 Blockchain deployment and assumptions . . . 59
7.1.2 System and Blockchain monitoring . . . 60
7.1.3 Benchmarks . . . 61
7.1.4 Final system definition and configuration . . . 68
7.2 System Evaluation . . . 69
7.2.1 User Actions and responses . . . 69
7.2.2 ARCADIAN-IoT interactions and responses . . . 72
7.2.3 System showcase . . . 72
7.2.4 PDA interaction . . . 75
7.2.5 Non-functional evaluation summary . . . 77
7.2.6 GDPR compliance evaluation and discussion . . . 77

8 Conclusion **81**
8.1 Future Work . . . 82
8.2 Final Note . . . 83

Appendix A Article submitted to EAI SecureComm 2023 **93**

Appendix B User Consent - Draft **113**

Appendix C Joint Controllership Agreement - Draft **117**

Appendix D Messages specification **119**

Appendix E GDPR.EU GDPR compliance checklist **121**

Appendix F System Demonstration **123**
F.1 JCA Smart Contract interaction . . . 123
F.2 Asset Management Smart Contract interaction . . . 124

Acronyms

ABE Attribute Based Encryption.

AI Artificial Intelligence.

AMQP Advanced Message Queuing Protocol.

AMSC Asset Management Smart Contract.

CA Certification Authority.

DDoS Distributed Denial of Service.

DoS Denial of Service.

DPO Data Protection Officer.

EAI European Alliance for Innovation.

EPRS European Parliamentary Research Services.

EU European Union.

GDPR General Data Protection Regulation.

HE Homomorphic Encryption.

ID Identification number.

IoT Internet of Things.

JCA Joint Controllership Agreement.

JCA-SC Joint Controllership Agreement via Smart Contract.

JCASC Joint Controllership Agreement Smart Contract.

MITM Man-in-the-Middle.

MSP Message Service Provider.

NIST National Institute of Standards and Technology.

OS Operating System.

PDA Personal Data Analyser.

PII Personal Identifiable Information.

RDFa Resource Description Framework in Attributes.

RO Research Objective.

RPC Remote Procedure Call.

STOA Panel for the Future of Science and Technology.

TLS Transport Layer Security.

TX/s Transactions per second.

WP Working Party.

ZKP Zero Knowledge Proof.

List of Figures

2.1 Overall Analysis of permissioned Blockchains Graph [51] 23

3.1 General scheme of structure and components that constitute ARCADIAN-IoT project [5] 28
3.2 High-Level view of BlockPrivGDPR interactions 29

5.1 Sequence diagram of Hyperledger Fabric transaction [34] 39
5.2 Chaincode normal invoke payload 41
5.3 Normal asset data stored 41
5.4 Event data 42
5.5 Asset properties for chaincode invocation 42
5.6 Private asset data stored 42
5.7 Chaincode private invoke payload 43
5.8 Hybrid asset data stored 43

6.1 Practical scheme of the solution proposal 46
6.2 Practical scheme of the solution implemented 49
6.3 Architecture overview of implemented solution 50
6.4 High-Level Architecture of Reputation system and BlockPrivGDPR 51
6.5 General interaction scheme of the Gateway and PDA 54
6.6 Consent Creation Flow 56
6.7 User/Score Asset Handling Flow 57
6.8 Unknown Asset Creation Flow 57

7.1 Grafana Dashboard General System Information 60
7.2 Grafana Dashboard Specific Blockchain Statistics 61
7.3 Hyperledger Caliper report example 63
7.4 Bar chart of Create a car benchmark results 64
7.5 Bar chart of Change car owner benchmark results 65
7.6 Throughput (TX/s) of the Blockchain in a round of Hyperledger Caliper 66
7.7 Bar chart of Create Unknown asset benchmark results 67
7.8 Bar chart of Asset Management benchmark results 68
7.9 Filled user consent form 69
7.10 Data update form 70
7.11 Form request result 70
7.12 Right of Access form 71
7.13 Contact form 71
7.14 Gateway: creation of user consent and asset 73

7.15 Gateway: restricted data read 73
7.16 Gateway: user reputation score update 73
7.17 Gateway: consent blockage 74
7.18 Gateway: user reputation score update failure 74
7.19 Gateway: user consent unblock 74
7.20 Gateway: right of access request and response 74
7.21 Gateway: user consent revoke 75
7.22 Gateway: user data query failure 75
7.23 Gateway: PDA interaction 76
7.24 Gateway: Unknown asset querying 76
7.25 PDA: String analysis 76

F.1 Variable set for direct Blockchain interaction 123
F.2 Direct invoke for JCA consent via terminal 123
F.3 Error returned on duplicated asset creation 124
F.4 Direct logging assets query and response 124
F.5 AMSC execution failure as the organization hasn't provided JCA consent 125
F.6 Consent creation example 125
F.7 User/score asset creation example 125
F.8 Consent, restricted and private user details query 126
F.9 Right of access query example 126
F.10 Update Personal Data example 126
F.11 User/score asset fails to be created, as the user hasn't provided consent 127
F.12 User/score asset fails to be created, as the user has blocked processing 127
F.13 Member of organization 2 tries to query organization 1 user data . 127
F.14 Update score mvcc_read_conflict error 128
F.15 Org1 user fails trying to access Org2 private data 128

List of Tables

2.1 GDPR summary 10
2.2 Overall Comparison of security vulnerabilities of permissioned Blockchains, based on [46] 24
2.3 Overview of found solutions for Blockchain compliance 26

4.1 Risk Analysis Summary 35

7.1 Create a Car Hyperledger Caliper benchmark test results 63
7.2 Change Car Owner Hyperledger Caliper benchmark test results . 65
7.3 Simple query Hyperledger Caliper benchmark test results (average) 66
7.4 Create Unknown asset benchmark test results (not using Caliper) . 67
7.5 Asset Management benchmark test results (not using Caliper) . . . 68
7.6 Non-functional requirements fulfilment table 77
7.7 GDPR Rights 78
7.8 GDPR Principles 78

Chapter 1

Introduction

Since it was introduced in 2018, the General Data Protection Regulation (GDPR) has changed significantly the way that personal data is seen and handled by companies and countries. The fact that compliance is mandatory not only for businesses operating in the European Union (EU) but rather for any business that uses data from EU citizens makes it a global regulation [2].

To avoid non-compliance and enforce the regulations, the EU defined high fines that can go up to 20 million euros, or 4% of the company revenue [61], being examples of this enforcement a 746 million euros fine to Amazon, and a 405 million euros fine to Instagram [64].

Being Blockchain a relatively new technology with a compound annual growth rate of 56.3% [55], and with an expected growth rate of 63% over the next 6 years in the healthcare industry [55], it's safe to say that this technology is currently very relevant, and will become even more in the near future.

Blockchain, by definition, is not compliant with GDPR, due to its immutable and decentralized properties. In this work, the conflicts between the two will be identified and explored, with the main goal of finding solutions for all of them, not only from a theoretical but also from a practical perspective.

1.1 Objectives

The main goal of this work is to create a GDPR compliant platform to store and handle personal data in a Blockchain environment, , that can be used as part of a reputation system, in the ARCADIAN-IoT project [6].

In order to be able to create such a system, a careful examination of both Blockchain technologies and the GDPR has to be done, so the conflicts can be clearly identified.

After finding solutions for all conflicts, a general platform will be designed, implemented and evaluated, having GDPR and personal data in mind.

Support for additional software, like the Personal Data Analyser (PDA) [44], will also be developed, to increase the security of the platform and compliance with GDPR.

After all these steps, the final version of the system will be tested and evaluated.

1.2 Contributions

As for contributions, an article was submitted to European Alliance for Innovation (EAI) SecureComm 2023, an international conference on security and privacy in communication networks [14], (documented in Appendix A) that contains all the theoretical analysis of the GDPR compliance issues of Blockchain, and possible solutions, along with a Proof-of-Concept that describes in practice how to accomplish a Blockchain-based environment that is compliant with GDPR. This submission was rejected, and we are now working on the paper, based on the provided feedback, so we can repurpose it to a different end, namely, a more extensive article that summarize all the investigations, studies, practical implementation, and tests regarding the platform that will be created, to be submitted to a magazine.

The source code for the Blockchain-based solution is publicly released[1] as an open-source project under Apache Licence, Version 2.0 [21]. Due to project specificity, and privacy concerns regarding the ARCADIAN-IoT project, only the main components of the system will be of public access: the Gateway-invoker, and the Smart Contracts.

Contribution to a distributed reputation system that is transparent and more reliable by using Blockchain technology in a GDPR-compliant fashion, to be employed in the ARCADIAN-IoT project.

1.3 Structure

This work is divided as follows:

- **Chapter 2** - Information regarding all of the regulations and Blockchain studies, and the conflicts between them. An analysis of the literature is performed to understand and discuss possible solutions.
- **Chapter 3** - Presentation of the BlockPrivGDPR, as part of the ARCADIAN-IoT project. A high level architecture is presented, and the requirements are presented.
- **Chapter 4** - Description of the approach and research methodology used to accomplish the objectives of the work, along with the found risks and mitigation plans.

[1] Public repository is available at: `https://github.com/DansterPT/BlockPrivGDPR-public`

- **Chapter 5** - Theoretical and practical analysis, with special focus on privacy, of Hyperledger Fabric, the chosen Blockchain implementation for our system.
- **Chapter 6** - Presentation and description, of the developed system: BlockPrivGDPR. The whole decision-making process is also explained.
- **Chapter 7** - System setup and evaluation: on the first section we discuss how we setup the system, and present performance tests and benchmarks; on the second section the system behaviour is showcased, and an evaluation is performed.
- **Chapter 8** - Conclusions and summary of the key findings and accomplishments, along with future work suggestions.

Chapter 2

Background and State of The Art

2.1 Background

2.1.1 General Data Protection Regulation

The General Data Protection Regulation (GDPR) is a regulation created and approved by the European Union (EU), that was put into effect on the 25th of May 2018 [2]. The goal of this regulation is to impose obligations on how to handle, process and store the personal data of EU citizens, anywhere in the world. It describes how the whole process of collection, storing, and processing data should be done, the rights of the data subjects, and the fines and penalties that can arise when entities do not comply with it. Further analysis of the regulation will be done by looking into its articles one by one.

GDPR articles summary

This summary is based on the General Data Protection Regulation official release [61], with small exceptions when other references are stated. All chapters and articles references are to the mentioned document.

GDPR is divided into 11 chapters. Being the most relevant for this work chapters 2, 3 and 4, which will be the most explored ones (please note that some articles will not be fully covered, or not covered at all).

Before providing details about the mentioned chapters, it's relevant to explore **chapter 1 - General provisions** (articles 1 to 4). In this chapter, the scope and definitions of the regulation are defined, and to better understand the scope and context, some important definitions are transcribed from GDPR.EU website[2]:

- ***Personal data*** — *"Personal data is any information that relates to an individual who can be directly or indirectly identified. Names and email addresses are obviously personal data. Location information, ethnicity, gender, biometric data, religious beliefs, web cookies, and political opinions can also be personal data. Pseudony-*

mous data can also fall under the definition if it's relatively easy to ID someone from it."

- ***Data processing*** *— "Any action performed on data, whether automated or manual."*
- ***Data subject*** *— "The person whose data is processed."*
- ***Data controller*** *— "The person who decides why and how personal data will be processed."*
- ***Data processor*** *— "A third party that processes personal data on behalf of a data controller."*

Another important definitions are included in **article 4**:

- ***Pseudonymisation*** *- "The processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person."*
- ***Personal data breach*** *- "A breach of security leading to the accidental or unlawful destruction, loss, alteration, unauthorised disclosure of, or access to, personal data transmitted, stored or otherwise processed."*

Moving on to **chapter 2** (**articles 5 to 11**), the principles of GDPR are presented. In the first article of this chapter (**article 5**), the principles are presented:

1. **Lawfulness, fairness and transparency** (on the treatment of the collected data)
2. **Purpose limitation** (specific, explicit, and legitimate purposes are stipulated by the controller for the processing of the data)
3. **Data minimization** (only the necessary data is collected)
4. **Accuracy** (the data should be updated/rectified, or erased, on subject request)
5. **Storage limitation** (the data is only stored while required or needed)
6. **Integrity and confidentiality** (security measures are applied to guarantee the security of the personal data)
7. **Accountability** (the controller is responsible to ensure and demonstrate compliance)

On **article 6** the lawfulness of processing is discussed. For the processing to be lawful, there are six possibilities, being the first one the one that will be taken into effect in this case: *"the data subject has given consent to the processing of his or*

her personal data for one or more specific purposes". Additional considerations are documented in this article.

On **article 7** the conditions for consent are discussed. For consent to be valid, it has to be freely given by the data subject, and he/she must be properly informed of the purpose and the ways his/her data will be used *"in an intelligible and easily accessible form, using clear and plain language"*. The data controller has to be able to demonstrate that the subject gave the consent, and the data subject must have the possibility to withdraw the consent at any given moment, as easily as he/she provided it.

The remaining articles of this chapter are regarding special categories/types of personal data that will not be covered in this work, and therefore will not be thoroughly explained.

Chapter 3 enumerates and clarifies the rights of the data subjects.

The first 3 articles (**articles 12 to 14**) are about the **right to be informed**. These articles include the information that must be provided from the data controller to the data subject, be it obtained by the subject himself/herself or not, how the communication must occur, and also explains how transparency should be achieved.

Article 15 describes the **right of access**: the subject has the right to obtain from the controller everything regarding his/her data (purpose, categories of data used, to whom it has been shared, the period of time the data that will be stored, the source of the data, if there is automated decision-making on the processing); can at any time request the rectification or erasure of the data; can request a copy of the personal data that the controller has, in a commonly used electronic form (which the controller must provide without any costs for the first time).

The data subjects also have the **right to rectification** (**article 16**) of their data in the cases the data is either inaccurate or incomplete; the **right of erasure** (**article 17**) where it's stated that the data must be erased without undue delay (with some exceptions) if the subject requests it with a valid justification (e.g. the personal data is no longer necessary for the purpose it was collected).

The **right to restrict processing** is defined by **article 18**, where it is explicit that even if the data remains stored, it cannot be processed if this right is invoked. The subject can state that the data is inaccurate, the processing is unlawful (and the subject doesn't want the erasure of the data), or that the controller no longer needs the data for processing (but still needs it for legal reasons). There is an obligation from the data controller, to notify the subjects regarding the moment when the requests of the previous three rights are complied with (**article 19**).

On **article 20**, it's defined the **right to data portability** which obligates the controller to provide all the personal data about the requester in a machine-readable format, that can be sent to a third party (directly, if possible).

Article 21 determines the **right to object**, which states that the user must be able to object to the processing of its data, and the controller has to stop using it unless the controller proves that there are legitimate reasons to keep processing it (this

is not valid in some cases, like direct marketing).

The last right, which regards **automated decision-making, including profiling**, is present in **article 22**. In this right, it's stated that unless it is necessary, it's authorised by Union or Member state law, or there is explicit consent from the data subject, the subject has the right *"not to be subject to a decision based solely on automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly affects him or her"*.

Chapter 4 defines **controllers and processors**. In terms of the Blockchain, this is an important chapter, since there will be many copies of the data available, and there should not be any "grey area" to avoid any possible infraction of the regulation.

Article 24 defines the responsibility of the controller. The controller must implement the required technical and organisational measures that ensure compliance with GDPR. The compliance must be demonstrable, and the procedures should be systematically reviewed. **Article 25** complements the previous by adding that data protection should exist by design and default and that state-of-the-art technologies should be used by the controllers to ensure it.

In the case, there is more than one entity defining the purposes and how data will be processed, these entities will be considered Joint Controllers as specified in **article 26**.

Article 28 describes the processor role. Transcribing part of the first paragraph of this article: *"the controller shall use only processors providing sufficient guarantees to implement appropriate technical and organisational measures in such a manner that processing will meet the requirements of this Regulation"*. The next paragraphs define how the processor should act, and how the contract between the controller and the processor should be elaborated: the contract should be in writing and define the nature, purpose and duration of the processing. After the acceptance of the contract, the processor must only use the data on documented instructions, must ensure that any person that has access to the data will commit to confidentiality, must ensure the security of data and processing, and assists the controller as much as possible. The tenth and last paragraph of this article are very relevant for this study: *"...if a processor infringes this Regulation by determining the purposes and means of processing, the processor shall be considered to be a controller..."*.

Article 29 states that a processor can only process the data when instructed to do so by the data controller (the EU and/or Member State law can also instruct the processor to do so).

According to **article 30**¸ a record of the processing activities must be kept by the controller(s), and a record of all categories of processing activities by the processor(s) (there are some exceptions where the entities are not obligated to keep records).

Article 31 specifies that the parties involved in the data processing and that must cooperate with supervisory authorities, on request.

The **security of processing** is defined by **article 32** where it's stated that when

possible: personal data should be pseudonymised and encrypted; confidentiality, integrity, availability and resilience should be ensured; availability must be restored in a timely manner in the event of an accident; there should be a regular process put in place to test and evaluate the security measures.

The next two articles (**33 and 34**) explain how data breaches should be handled in regard to notifying the supervisory authority and the data subject.

When using new technologies, or a controller is dealing with large amounts of data, or sensitive data, a data protection impact assessment should be performed either by the controller or by the Data Protection Officer (DPO), as referred in **article 35**. Prior consultation of the supervisory authority to the assessment is explained in **article 36**.

Article 37 explains how and when a DPO should be designated:

- *"the processing is carried out by a public authority or body, except for courts acting in their judicial capacity"*
- *"the core activities of the controller or the processor consist of processing operations which, by virtue of their nature, their scope and/or their purposes, require regular and systematic monitoring of data subjects on a large scale"*
- *"the core activities of the controller or the processor consist of processing on a large scale of special categories of data pursuant to Article 9 or personal data relating to criminal convictions and offences referred to in Article 10"*

Articles 38 and **39** state the position and tasks of the DPO, that will not be explored, since the platform does not fit under the conditions to require a DPO.

Codes of conduct are defined on **articles 40** and **41**; and certification and certification bodies on **articles 42** and **43**.

Chapter 5 depicts the transference of personal data to countries outside the EU. This chapter will not be covered, as for this work, it will be considered that there will be no entities operating from outside the EU.

The remaining six chapters will also not be analysed since it's out of the scope of this study, however, a brief summary of what's present in those chapters is going to be provided:

- **Chapter 6** is about supervisory authorities, their tasks, competencies, and powers, and the rules and conditions that they must base their actions on.
- **Chapter 7** describes the cooperation and consistency required between all the supervisory authorities and European data protection board.
- **Chapter 8** contains information on how to take legal actions and complaints with the supervisory authorities, and the penalties that can be applied in the cases of non-compliance.

- **Chapter 9** presents the "exceptional cases", and the provisions to be taken regarding specific processing situations (e.g. data collected by religious associations).

- **Chapter 10** explains the exercise of the delegation and the committee procedure.

- **Chapter 11** contains the final provisions.

Table 2.1 summarizes the content of the GDPR divided into articles.

Chapter	Article(s)	Content
1	1 to 4	General provisions: scope, context, and definitions
2	5	Presentation of the GDPR principles
	6	Lawfulness of processing
	7	Conditions for consent
	8 to 11	Special categories and types of personal data
3	12 to 14	Right to be informed
	15	Right of access
	16	Right to rectification
	17	Right of erasure
	18	Right to restrict processing
	19	Data controller notification obligations
	20	Right to data portability
	21	Right to object
	22	Automated decision-making, including profiling
4	24, 25	Responsibilities of data controller and recommendations
	26	Joint Controllers definition
	27	Foreign controllers/processors representatives
	28, 29	Processor role definition
	30	Processing activities requirements
	31	Obligation of processing parties to cooperate with supervisory authorities
	32	Definition of security of processing
	33, 34	Handling of data breaches
	35,36	Data protection impact assessment, and prior consultation
	37 to 39	DPO definition
	40, 41	Codes of conduct definition and monitoring
	42,43	Certification and certification bodies
5	44 to 50	Transference of data to countries outside the EU
6	51 to 59	Independent supervisory authorities
7	60 to 76	Cooperation and consistency
8	77 to 84	Remedies, liability and penalties
9	85 to 91	Provisions on specific cases
10	92,93	Committee procedure and delegation
11	94 to 99	Final provisions

Table 2.1: GDPR summary

2.1.2 Blockchain

This subsection is based on National Institute of Standards and Technology (NIST) Internal Report 8202 [68].

Blockchain is a digital-distributed ledger, that does not rely on a secure/trusted third party (centralised authority) to guarantee trust, but rather uses cryptography mechanisms that prevent any tampering with the recorded data.

The transactions between the users are recorded in chained blocks: each block contains the hash value of the previous one, which guarantees chaining and chronological order, integrity, and immutability to all recorded transactions. The authenticity of the data is also assured using public-private sets of keys (digital signatures).

A block is divided in two parts: the block header that contains the information of the block itself (block number, a hash of the previous block, block size, etc); and the block data containing the transactions' data. For a block to be added to the Blockchain it's used a consensus mechanism, that depends on the implementation of the Blockchain.

Due to the fact that there is no centralized party that keeps the records, there are many copies of the ledger among the network, that are saved and kept updated by the "full nodes" (participants of the Blockchain that keeps a full copy of the chain).

In terms of access and participation, Blockchain can be of **public** or **private** access and can be **permissionless** (anyone can see and interact with it) or **permissioned** (only users with access can see and/or interact with it. Note: it's possible to have some user related restrictions and allow only to see some specific parts of the chain, e.g., the user own transactions).

Blockchain types

Permissionless Blockchains are the ones that correspond to the definition of Blockchain introduced by Satoshi Nakamoto in the paper about Bitcoin [49]. In this type of Blockchain, everything is public, meaning that anyone can join or leave the network, and everyone can access all the information written on the blocks [68]. There are, however, some protection/confidentiality mechanisms that protect the transactions' information [68]: Cryptographic Hash Functions (usually SHA-256) and asymmetric-key cryptography (public-key cryptography). These techniques are also used to guarantee integrity and non-repudiation (by using digital signatures).

Permissioned Blockchains where only authorized users can join and interact with the network, being this control done by a centralized or decentralized authority [68]. This means that all the users are identified and known to the mentioned authority, and there can be some level of trust between the users. Depending on the practical implementations, there can be different restrictions and access control mechanisms. The base cryptographic mechanisms are the same as

mentioned above [68].

Consensus Models

Depending on the type of Blockchain, the levels of trust are different, and therefore, different types of consensus mechanisms can be implemented, depending on it. Since in private/consortium **permissioned Blockchains**, the users have to be identified in order to join, there is some level of trust between the nodes, so, simpler consensus mechanisms can be used. One example is the Raft consensus algorithm, that consists in a leader-followers approach: the nodes participate in an election to elect a leader that will be receiving the transactions from the client; when a transaction is received, the leader executes it, and sends it to the followers for validation, if the majority of the followers agrees with the result, then the transaction is stored and committed [24].

For **permissionless Blockchains**, on the other hand, since there is no level of trust, more complex techniques are needed to avoid the control and publishing of malicious blocks in the chain. The two most common ones are [68]:

- **Proof of Work -** in this model, the nodes are required to solve a computational puzzle that is hard and slow to solve, but very fast to verify and validate. The idea is that the nodes add a nonce (random number) to the block data to generate a hash value that has a given number of zeros in the beginning. Depending on the number of nodes, and the time spent to find this solution ("mine a block"), there are mechanisms for the Blockchain to adjust the difficulty itself (increase or decrease the number of zeros in the beginning) of this computational challenge, which will assure that the time to generate a new block is stable (usually around ten minutes). Since the verification of the nonce and hash value is very fast, all nodes of the Blockchain can easily verify that the block is legitimate. To attract nodes, and compensate them for the computational effort, rewards are given to the nodes that "mine a block", in the form of cryptocurrency, which is a digital coin associated with a given Blockchain platform.
- **Proof of Stake -** The idea behind this model is that the bigger the investment of a given node in the Blockchain, the bigger the chance they want it to succeed. Having this in mind, a node will have has much probability to generate the next block as it has participated in the network (note: there are variations of this model, but the concept is always the same: to give more influence to the bigger stakeholders). In this model, there are no "mining fees", but rather a transaction fee charged to the nodes involved in the transactions recorded in the given block, that will be given to the block that generates the block.

Permissionless vs Permissioned

In terms of the types of Blockchains, the permissionless ones create a lot more challenges in terms of privacy, security and confidentiality:

- The fact anyone can join and interact with the network makes the content and information of the transaction open to everyone.

- Since there is no access control, it's not possible to limit which entities access (and process) the data.

- Even though there are cryptographic algorithms that ensure security and confidentiality at this point in time, it's not possible to guarantee that this will happen for the whole time that the network will be kept (but even if it could be, there's no guarantee that no one will store an offline copy of the whole network to try and decrypt it in the future). It's stated by the Article 29 Working Party [50] that when designing and planning the system, it should be ensured that the encryption techniques used will be enough for the whole period the data will be processed.

- In a permissionless Blockchain, the users are not identified, and therefore it's not viable to have contracts between the controllers and the unknown processors.

- Another disadvantage is the resource and energy consumption required by most used Consensus Models of this kind of Blockchain (e.g. Proof of Work consensus model described above).

2.1.3 GDPR vs Blockchain

Bearing in mind the rights and principles of the GDPR previously stated and explained, in this section the conflicts between the Blockchain principles and mechanisms and the GDPR will be explored.

Principles

Lawfulness, fairness and transparency

Regarding lawfulness and fairness, Blockchain does not bring any specific problem. As long as it is freely provided informed consent by the data subject, and it's specified all the uses of the subject's data, and all the entities using it (which can be an issue when using permissionless public Blockchains) [15]. All the legal grounds must be thoroughly explained, and all the participating entities must be named.

In terms of transparency, it can be hard to comply if there isn't a specific and simple channel of communication between the data subjects, and the data controller(s), so it must be ensured, that there is indeed a way of easy communication the users can use [15]. Another problem can be the assurance of which entities actually processed the data, and on which basis (purpose) the process has been done [56].

Purpose limitation

The data that is provided by a data subject has a defined purpose, and it must be only used for that purpose (even though there are some exceptions). There are some concerns regarding this matter: given the nature of Blockchain, if data is stored in the Blockchain itself, it is used for the calculation of the block hash value, which is then used on the next block, and therefore on the next block, and so on; meaning, that there is a "perpetual processing" of the data [15].

Since there are many intervening entities, the purpose should be clarified regarding each one, and access control mechanisms should be put in place [15]. The statement of the purpose should be clear and transparent.

Data minimization

Blockchain by definition only has the option to append data [68], which means that it will continue growing over time and keeping all the introduced data. Also, the fact there are multiple copies of the ledger is another concern for this principle [15].

Accuracy

Given the nature of Blockchain, it's possible to assure the immutability of the data, which is both a blessing and a curse for this specific topic. Because of this characteristic, we can assure the accuracy of the data provided, however, it's mandatory that the data should be rectified or erased without delay. Having this in mind, it's mandatory to have a mechanism that allows both rectification and erasure of earlier provided data [15].

Storage limitation

As previously stated in Data minimisation, the fact that Blockchains only allow append operations create some conflicts on this principle, being the main question: *When does the data become obsolete?* [15]. Even though anonymization techniques may help to comply with this principle, it is not a silver bullet, and additional mechanisms will have to be designed [15].

Integrity and confidentiality

Integrity is assured by the Blockchain, so there are no specific problems in this topic; however, confidentiality can be a problem. If the data can be accessed by unauthorized attackers, it will be a confidentiality issue. So, there should be mechanisms implemented that ensure the security and confidentiality of the data.

Encryption might help achieve some level of confidentiality, however, when using cryptography, it should be assured that the algorithms used will remain secure throughout the whole lifetime of the platform, which can be a challenge [15], especially when thinking about permissionless Blockchains, and that there are many distributed copies of the ledger that can or not be erased in due time (depending on the user's control).

When using permissioned Blockchains there is access control, that regulates who can access a given piece of data, but there can be "unsafe" records of the data stored with public access (like a plain hash value of the private data), which also raises concerns [15].

Accountability

The data controller is the identity responsible to specify and implement the technical and organizational measures to ensure that the processing of the data is compliant with GDPR, and it should be demonstrable [61].

In case, there is more than one entity that is defining the terms and purposes of the processing, there is not a data controller, but rather joint controllers [61]. These identities, usually have a Joint Controllership Agreement (JCA) where the purposes, the measures, and the means for processing are stated, and how the parts are involved.

Having the above in mind, specific problems can arise depending on Blockchain implementations and use cases. If a single data processor can define and demonstrate compliance within the whole Blockchain network, it could be the preferred solution, however, it may not be the case, given that can be many different entities and organisations participating in the distributed ledger, and it can be better to implement a Joint Controllership Agreement among all the involved parties [15].

One specific issue that arises from the use of a single data controller is that when a full node goes offline, the data processor won't be able to reach it, and therefore, it won't be possible to demonstrate its compliance, and the node won't receive updates regarding the data that it has, which can lead to accuracy issues.

Rights

The right to be informed/of access

It is fundamental that the data subjects have all the information needed to be fully informed regarding how their data is used and processed and by whom; and for that, specific mechanisms must be provided to the subject to request this information.

Even though there are no specific issues regarding the principles of Blockchain, there can be implementation-specific issues that should be considered: if there are joint controllers, a subject must be able to request this information for any of it, which can be hard to implement and guarantee, given the nature of the network.

The right to rectification

This right is not compatible with the original nature of the Blockchain, since it is an append-only ledger, and therefore immutable. So, a specific mechanism has to be implemented.

The right to erasure

Along with the right to rectification, the right to erasure is also not compatible with the concept of Blockchain.

The right to restrict processing

This right faces the same challenges of rectification and erasure, since it is not

possible by design to change the data stored in the Blockchain, it can be hard to restrict the processing. Besides the technical obstacles, there might also be governance obstacles, due to the number of participants, and, potentially, controllers [15].

The right of data portability

Blockchain itself doesn't seem to create any concerns for this right as long as there are mechanisms for the subjects to obtain their data in structured and readable format [15].

The right to object

If a subject objects to the processing of his/her data, the involved parties can no longer process the data, however, this right faces the same obstacles as the right to restrict processing.

Rights in relation to automated decision-making and profiling

In relation to automated decision-making, it can be stated that smart contracts pose an issue to this right [15]: *Smart contracts are simple programs stored on a Blockchain that run when predetermined conditions are met. They typically are used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss* [43].

As described by IBM, smart contracts are fully automated and need no human intervention to run, which can be seen as a challenge to this right, so it's important to analyse how to cope with this, and which safeguards must be introduced.

2.1.4 Support and Related Tools

In terms of support and tools that might be used to complete and expand this work, we use the Personal Data Analyser (PDA), which is an Artificial Intelligence (AI) driven solution that aims to detect Personal Identifiable Information (PII) on given input strings [58]. This tool can help to improve the security and privacy of the system, by detecting wrongly labelled personal data before it is stored as non-personal data, and it's available as an open-source project under Apache 2.0 Licence [44].

The reasons why we chose this specific software are the fact that it is open-source and can be modified to be integrated in our project, the fact that it was developed by a colleague of Coimbra University, and its high success rate in PII identification (average of 90% of accuracy) [58].

PDA integration is further explored in subsection 6.2.6, and demonstration in subsection 7.2.4.

2.2 State of the art

As detailed in the previous section, there are many GDPR compliance issues that arise from using Blockchain platforms. In this section, the state of the art, and related work regarding these themes will be explored.

2.2.1 Blockchain vs GDPR

In the Background section, a summary of the principles and rights was done, and some concerns were mentioned. In this subsection, an analysis of the techniques and mechanisms used to make the Blockchain platforms compliant with GDPR.

To avoid fragmentation and to better understand the used methods, the principles and rights were combined into bigger categories.

Lawfulness, fairness, transparency and the rights to be informed and of access

Consent is the most important thing when a user joins a network. It is extremely important that the conditions and data usages are explicit, detailed and accurate. Having this in mind, and assuming that this is correctly done, Smart Contracts can be used to manage consent.

Smart Contracts can be ideal for this management for two particular reasons: the use of machine (binary) language, makes it easier to understand since it is more objective and therefore removes ambiguity, and subjectiveness; All smart contract executions are logged in the Blockchain, creating a log that assures integrity, immutability and non-repudiation [23].

As contracts, smart contracts are also made by humans, which means that there can be errors, or software bugs, which can compromise the validity of the contract [23]. This should be taken into consideration, and there should be ways to update the software and re-ask subjects for consent if needed.

Given the fact that there can be different types of consent, and one can consent to more or less than another, [12] suggest the use of Resource Description Framework in Attributes (RDFa) ([57]) to manage and store the given consent by each user, which would be later consulted by the Controllers/Processors of the data to assure that the operation is lawful, given the user consent.

In terms of fairness, as mentioned in subsection 2.1.3, there has to be a direct channel of communication that the subjects can use to reach the controller, and for that, the entities must be identified [15], which is only objectively possible on permissioned Blockchains. In permissionless it would create technical and governance challenges [23].

In terms of transparency, the main issue found was regarding the identification of the entities that processed the data and on which basis [56]. This can be addressed by creating a log of all transactions regarding the data [47]: a smart con-

tract for logging all the interactions with the data can be put in place to create a log, that would guarantee the same characteristics stated above for smart contract execution, and would create the possibility to audit all actions, and to prove compliance. This audit could be done both in general (to all data processed), or to a particular subject, in case of a specific request or complaint.

With these mechanisms, we can also assure that the basis to comply with the right to be informed, and the right of access are put in place, the only thing that would be required would be to implement a way for the user to request and receive its data, and the information about who processed it, and when. For this, the above-mentioned channel of communication could be used, and a smart contract could be put in place to automatically process these requests and to gather and send the information in a machine-readable way to the subject.

Purpose limitation, and the rights to restrict processing and to object

When a subject gives consent, he/she must be able to withdraw or change the given consent. The use of RDFa mechanisms, discussed above could be a way to restrict (or narrow) the consent, and if expanded, could also be used for the users to exercise the right to restrict processing, and to object.

Storage limitation, accuracy and the rights to rectification and erasure

The append-only nature of Blockchain brings some concerns regarding the storage of the data, since it's not legally viable to store personal information directly in the chain, unless there are some additional mechanisms to edit the Blockchain, like a deletion consensus mechanism [15].An easier solution, and one that avoids the change of the principles of the Blockchain, is to keep the personal data off of the chain and store only an integrity proof of it on-chain, like a hash value. Nonetheless, there are some limitations regarding the hash use, as discussed on the integrity and confidentiality topic (section 2.2.1).

By keeping the data off-chain, it's easier to comply with GDPR, because the immutability of the records is bypassed, and it's possible to alter, and delete the records of the chain, and create a new block on the chain with the corresponding integrity proof recorded on chain (but we still have to guarantee the untraceability of the Blockchain recorded values, more on this on integrity and confidentiality).

There are some limitations of the off-chain mechanisms [23]: it can increase costs and complexity of the system; the multiple copies of the data can become desynchronized (more on this on data minimization); it can reduce the security level of the Blockchain (more on this on integrity and confidentiality).

Another solution proposed in the literature is the use of chameleon hash functions. Cryptographic ("regular") hash functions are by definition collision-resistant, which means that it's computationally infeasible to compute two different inputs that produce the same output. So, when the input changes, the output will also

change.

Chameleon hashes, however, allow changes in the input value, while the output remains the same: when the hash value is first calculated, a secret key (called *trapdoor key*) is also generated. This key can be used later to produce the same output value with different inputs [53].

In the model presented by H. Precht and J. Gómez [53], the blocks contain two security measures regarding the chameleon hashes:

- The equivalent to the block header (the Blockchain implementation used by the authors is simplified) contains the hash value and a checksum:
  - This checksum will change if the information inside the block changes, even though the hash value will remain the same, so it's easily noticeable and controllable when a block changes.
- Only the block owner has the trapdoor key:
  - This guarantees that only the owner of the block is able to change it.

In this case, the decision of the change in the block is not up to the block owner, but rather to an external entity (e.g., the owner of the data), so, the mentioned mechanism is not viable.

S. Han and S. Park suggest that the trapdoor key is only in the possession of a single trusted entity (e.g., the data controller), and all changes are controlled by them [23].

There are also some discussions regarding deletion consensus mechanisms, where there would be voting performed by all nodes, and if the majority agreed, the block would be deleted; however, due to the chained nature of the Blockchain, this would disrupt it [45].

Integrity and confidentiality

Integrity is guaranteed by Blockchain mechanisms, yet confidentiality is not. When using the mentioned off-chain techniques to allow compliance with other principles and rights, confidentiality must be kept in mind, as it is stated in Article 29 Data Protection Working Party [50] that:

- The sole use of hash functions is not enough to ensure the confidentiality of a record
- The use of salted-hash functions can hinder input derivation, but it may still be feasible (referring to short salt values, and also noting that the salt values are commonly not secret)
- The use of a keyed-hash function with a stored secret key may introduce enough entropy and difficulty for an attack to be impractical

In the case of a Blockchain it may be infeasible to keep a secret key between all the involved parties, however, it might be possible to assure enough entropy (and therefore impracticality of attacks) by using sufficiently big single-use salt values (that would have to be stored along with the personal data, for integrity validation, but otherwise kept secret), as described by the Spanish data protection authority [13][1].

The usage of Access Control and/or encryption mechanisms are often the ones chosen to introduce and ensure confidentiality [23]. In terms of access control, there are two main techniques: using smart contracts (that will use X.509 certificates [29]), and by using Attribute Based Encryption (ABE) Keys.

Regarding encryption to achieve anonymization, there are two proposed encryption techniques: Zero Knowledge Proof (ZKP), and Homomorphic Encryption (HE).

ZKP are algorithms that provide boolean outputs that are the result of a verification of a given input, so a statement can be proved without the disclosure of any sensitive data. The usage of these algorithms might be enough to anonymize it [56].

HE allows some computations (addition, and multiplication) to be performed on encrypted information that produces the same result as if the plain text information was used. By using this kind of mechanism, the data can be stored and used in an encrypted format, and only the owner of the encryption key can later decrypt it [1]. The usage of HE has been proposed on Blockchain systems where only simple operations are required on the private data, like electronic voting systems [67].

Data minimization

If the personal data is kept off-chain as previously suggested in subsection 2.2.1, it's easier to comply with this principle, as the original data can be deleted, and it is computationally infeasible to guess the data values based on the Blockchain-stored integrity proof (assuming single-use salts are used) [13].

Along with the erasure requests, there can be time limits regarding the storage of the data that can be defined when the data object is created, and the data is automatically deleted when the 'expiry date' is reached (this is possible by default in some Blockchain implementations, e.g. Hyperledger Fabric [32]).

Another mechanism that can be used is Blockchain pruning: where old blocks' content would be erased and only kept the block header [17]. In the case an off-chain storage method is used, the data concerning the Blockchain that would be erased would also have to be deleted; and this would be just a complementary mechanism, as it does not satisfy the right of erasure [23].

Regarding the synchronization of the data between the parties involved, it should

[1]Other data protection authorities provide descriptions, but they are not informative as the Spanish data protection authority documents

occur by default every time a new block is committed to the Blockchain [15], so even if a node stays temporarily offline, it should synchronize its data when the next block is committed. In the eventuality a node goes offline for a longer period of time than accepted, it can be implemented data self-destruction mechanisms like proposed by L. Zeng et al. [69] to cloud computing, that guarantee that possible non-compliant data is destructed.

The default mechanisms of Blockchain, however, do not guarantee the synchronization of the off-chain databases, but there have been proposed mechanisms to assure that this is done in a short period of time, e.g. Digital Data Converter [7].

Accountability

As prior discussed, it may be hard to enforce compliance in terms of accountability in a Blockchain system given the fact that the data is distributed, the nodes (which includes the data controller(s)) may not be available at all time, and that there are many parties involved in the data processing. [52] suggest the implementation of a Joint Controllership Agreement via Smart Contract (JCA-SC) that would be used to:

- Guarantee GDPR compliance - Since the data would have to be processed according to the JCA
- Guarantee that a valid JCA is in place - As a call to the JCA-SC would be made at every method of the processing contract
- Guarantee that at least one of the controllers is available - Since, in order for the JCA to be valid, there has to be at least one controller node online

This integration would make contract calls less efficient, however, the implementation is considered to be simple and straightforward in implementations that support smart contract calls inside other smart contracts (like Hyperledger Fabric), and there are many benefits in terms of guaranteeing compliance [52].

The right of data portability

In the case of this right, as long as the data and at least a controller are available, there are no conflicts between providing the data to the owner and Blockchain. The JCA-SC solution mentioned [52] could be enough to guarantee the availability of a data controller, that could send the data in a machine-readable format.

Rights in relation to automated decision-making and profiling

M. Finck [18] analyses how smart contracts interact with the rights in relation to automated decision-making and profiling (article 22 of GDPR) [18]. M Finck concludes that when considering the original meaning of smart contracts (totally automated, and run automatically when pre-conditions are met [43]), those

are not compatible with article 22; however, there can be ways to create GDPR-compatible smart contracts, namely the introduction of *"arbitration mechanisms"* [18] to it.

These mechanisms would allow humans to pause, alter, or even remove the software that composes the smart contract, and furthermore, the contracts could have methods to provide information on data usage to the subjects. These conditions would make the smart contracts compatible with the mentioned article [18].

Exemption of GDPR for Blockchain

The study of A. Mirchandani [48] analyses the interaction between Blockchains and GDPR, focusing on the permissioned ones. The author of the study suggests that the easier way to ensure compliance and to be able to use permissioned Blockchains that store personal data would be to exempt it from the GDPR. For this to happen, one of the solutions would be for the users to provide explicit consent not to exercise some of their rights, namely the right to be forgotten, the right to data portability and the right to rectification.

Another solution for this exemption would be [48]: to alter the language of GDPR to include an exception for permissioned Blockchains; to clarify the definition of data erasure under the GDPR; to classify plain hashed personal data as anonymized (instead of pseudonymized, which is still considered personal data).

2.2.2 Blockchain Implementations

Given the reasons presented in section 2.1.2, it would be much more difficult to use a permissionless Blockchain than a permissionless one, and consequently, only permissioned implementations will be analysed and compared.

J. Polge et al. performed a comparison of the five major implementations of permissioned and private Blockchains used in the industry: Hyperledger Fabric, Ethereum Geth, Quorum, MultiChain, and R3 Corda [51]. The mentioned comparison was divided into five different categories:

- **Community activity** – based on GitHub contributors and commits, and on Twitter followers and tweets
- **Adoption** – based on the industrial use cases found in Forbes: Blockchain 50 of 2019, that summarized the Blockchain platforms used by the biggest worldwide companies [10]
- **Privacy/Confidentiality** – based on how restrictive the implementation can be in terms of access control, and its security mechanisms
- **Scalability, Throughput and Latency** – based on the best results of real deployments found by the authors [51]

The compilation of results is provided by the figure 2.1:

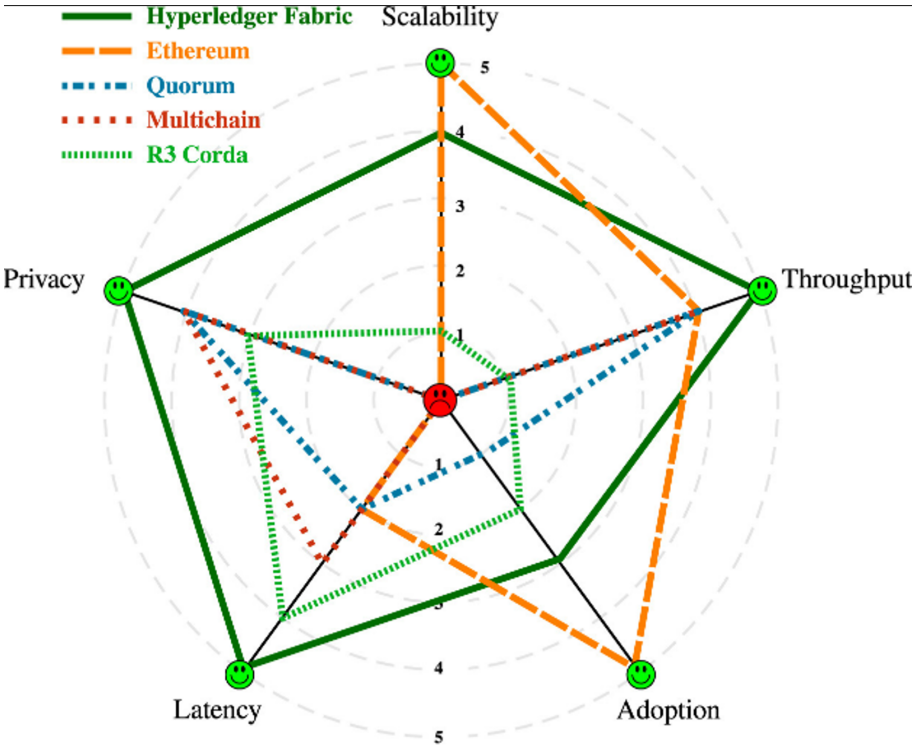

Figure 2.1: Overall Analysis of permissioned Blockchains Graph [51]

By the analysis of the figure 2.1 presented above, it's visible that Hyperledger Fabric had the best score (5 out of 5) in three of the five categories analysed, and was in the second position in the remaining two categories, where Ethereum Geth performed the best.

For our work, the most important feature is **privacy**, in which Hyperledger had the best score among all the tested implementations. Ethereum Geth scored the worst possible score in this category (0 out of 5). Quorum and Multichain both scored 4 out of 5.

The second most important matter would be **adoption**, since it can be translated into more tested use-cases, and more literature available. Here, Ethereum Geth was in the lead, and Fabric was in second place, followed R3 Corda. Quorum scored 1 out of 5, and Multichain 0.

In terms of **scalability**, **throughput**, and **latency** a mean was considered, and Hyperledger Fabric was again on the top with an average score of 4.67 out of 5, followed by Ethereum Geth with an average score of 3.67 out of 5, Multichain scored 2.33, and both Quorum and R3 Corda averaged a score of 2.0.

An evaluation in terms of **security**, that compares Ethereum, Quorum, Hyperledger Fabric, and R3 Corda, was performed by M. Lagarde [46]. The table 2.2 was generated based on the summaries provided by the author, and represents the types of vulnerabilities found for node/message sender authentication, and

	Ethereum		Quorum		Hyperledger		R3 Corda	
Attack/ Scheme	Node	Trans action	Node	Trans action	Node	Trans action	Node	Trans action
Brute force	-	-	-	-	-	IF	-	YES
Dictionary	-	-	-	-	-	IF	-	YES
Sniffing	-	-	-	-	-	IF	-	IF
Key compromise	YES	YES	YES	YES	NO	-	YES	YES
Replay	NO	NO	NO	NO	NO	IF	IF	IF
MITM	YES	-	IF	-	NO	IF	IF	IF
Session hijacking	YES	-	IF	-	NO	IF	NO	IF
Source non-repudiation	NO	NO	NO	NO	NO	NO	NO	NO
DDoS/DoS	YES	-	IF	-	IF	YES	IF	YES

Table 2.2: Overall Comparison of security vulnerabilities of permissioned Blockchains, based on [46]

transaction/Remote Procedure Call (RPC) authentication, in the latest version of the implementations available at the time (2019).

The Blockchain platforms were tested for the following attacks: Brute force, Dictionary, Sniffing, Key compromise, Replay, Man-in-the-Middle (MITM), Session hijacking, Source non-repudiation, and Distributed Denial of Service (DDoS)/Denial of Service (DoS).

The label is the following:

- **"-"** - No available data
- **"YES"** - Vulnerability found
- **"NO"** - No vulnerability found
- **"IF"** - Vulnerability found, but there are built-in mechanisms to remove it

Since the vulnerabilities marked as "IF" can be fixed by changes in the configuration, only the ones marked as "YES" will be considered for our analysis. Having this in mind, we can conclude by the table that in terms of security, Hyperledger Fabric is the implementation with less overall found vulnerabilities (1), followed by Quorum (2), and Ethereum (5) and Corda (5).

Regarding the default configurations of each implementation, it was found no additional vulnerabilities on using it on Ethereum, Fabric would introduce DDoS and DoS vulnerabilities, Corda and Quorum would introduce many vulnerabilities [46].

After the analysis of the different permissioned Blockchain platforms, our belief was that **Hyperledger Fabric** would be the most suitable for the practical implementation.

2.2.3 Summary

Table 2.3 summarize the found solutions and which principles/rights it concerns.

Solutions	**Principles/rights**
Smart contracts for consent [23]	- Lawfulness - Fairness - Purpose limitation - Right to restrict processing - Right to object
Smart contracts for consent with RDFa [12]	- Lawfulness - Fairness - Purpose limitation - Right to restrict processing - Right to object
Direct Channel of communication [15]	- Fairness - Right of access
Smart contract for logging all data interactions [47]	- Transparency - Right to be informed - Right of access - Lawfulness auditing - Fairness auditing - Purpose limitation auditing
Deletion consensus mechanism [45]	- Storage limitation - Accuracy - Right to rectification - Right to erasure
Chameleon hashes [53]	- Storage limitation - Accuracy - Right to rectification - Right to erasure
Off-chain storage with proof of integrity on-chain [23]	- Storage limitation - Accuracy - Right to rectification - Right to erasure - Integrity
Off-chain storage with proof of integrity on-chain, calculated after addition of a big single-use salt value [23] ; [13]	- Storage limitation - Accuracy - Right to rectification - Right to erasure - Integrity - Confidentiality

Access control mechanisms (X.509 certificates, or ABE keys) [23]	- Confidentiality
Encryption as a way to anonymize data (using ZKP or HE) [56]	- Confidentiality
Define expiry date on off-chain stored data [67]	- Data minimisation
Blockchain Pruning [17]	- Data minimization
Data self-destruction in case online synchronization is not possible for a long period of time, based on the idea of [69]	- Data minimization
Digital Data Converter to enforce synchronization of off-chain stored data [7]	- Data minimization - Accuracy
Joint Controllership Agreements Smart Contract (JCA-SC) [52]	- Accountability - Right of data portability
Addition of "arbitration mechanisms" to smart contracts [18]	- Rights in relation to automated decision making and profiling

Table 2.3: Overview of found solutions for Blockchain compliance

Chapter 3

BlockPrivGPDR high-level overview and requirements

3.1 ARCADIAN-IoT

ARCADIAN-IoT is the acronym for Autonomous trust, security and privacy management framework for Internet of Things (IoT) devices. It is a project funded by the EU's Horizon2020 programme [11]. The project consists of the creation of a framework for trust, security and privacy management for autonomous networks of devices, services and human users. This framework will also empower users in terms of privacy due to its transparent and decentralized nature [6]. It will be developed and tested with three realistic use cases, namely [6]:

- Emergency and vigilance using drones and IoT
- Monitoring of grid infrastructure
- Medical IoT devices for telemonitoring patients

The figure 3.1 below presents a general structure and components that are part of the framework:

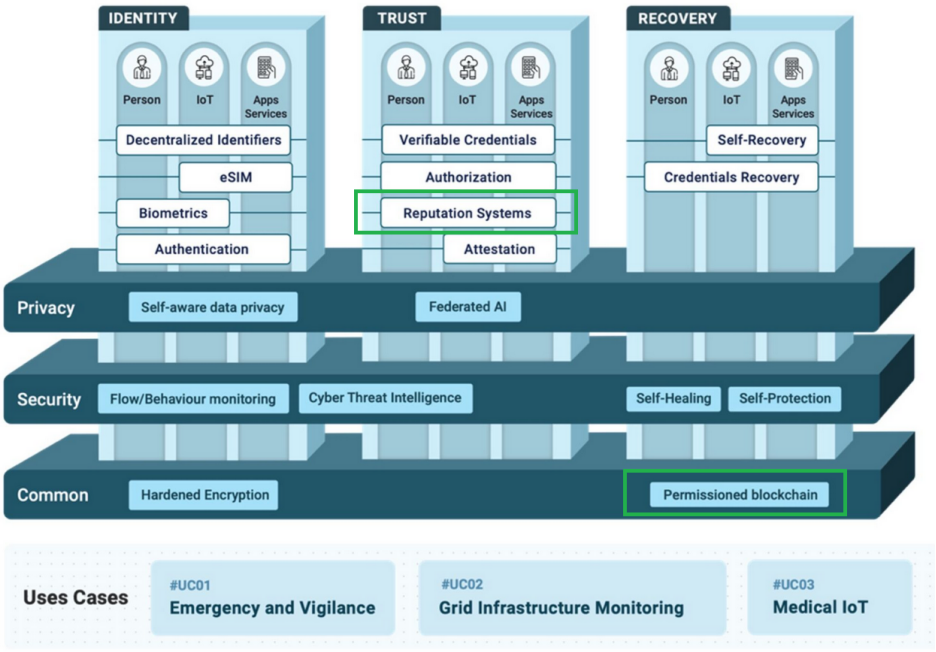

Figure 3.1: General scheme of structure and components that constitute ARCADIAN-IoT project [5]

This work is part of the Reputation System component, which is responsible to evaluate and classify the individual entities that are part of the network (humans, or not). The values/grades received by the reputation system, and used to calculate the grade of the entities, will be stored in the permissioned Blockchain, along with an identification and a timestamp. By storing the data in the Blockchain, it is assuring integrity, traceability, and immutability.

The reputation values and the identificators used are considered personal data in the light of the General Data Protection Regulation (GDPR), so, GDPR compliance is mandatory.

3.2 BlockPrivGDPR High-Level Architecture

Figure 3.2 depicts in a very high-level way how the different modules interact with BlockPrivGDPR. In the next subsections, framing, requirements, and how the components interact will be explained in a general way.

The BlockPrivGDPR components will be further shown and detailed in chapter 6.

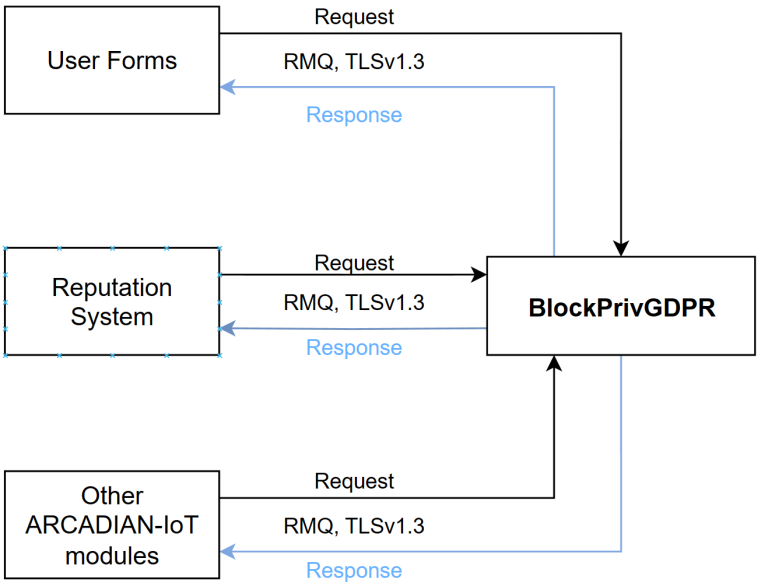

Figure 3.2: High-Level view of BlockPrivGDPR interactions

3.3 Framing and requirements

Even though BlockPrivGDPR was meant to communicate only with the Reputation System, as shown in Figure 3.2, it was designed and implemented in a way that it could interact with any component of the ARCADIAN-IoT environment, and also purposely made user forms that allow end users to perform system requests directly, in order to be able to exercise their GDPR rights without unnecessary delays.

3.3.1 Functional Requirements

The main requirement of the system is that it contains the required functions and components to comply with the GDPR, namely:

- The user has to have a mechanism to request all of its data, and the system should return it in a machine-readable format. Along with the data itself, it should also return the information regarding the purpose of the data, the storage time, the source of the data, and if automated decision-making is involved - Right of access, and right of data portability.
- The user needs to be able to update their personal data using a simple form - Right of rectification.
- The user needs to be able to restrict or object to the processing of their data, using a simple form, where the reason must be provided - Right to restrict processing, and right to object.

- Given the nature of Smart Contracts, and the fact that they perform automated actions, the system controller(s) must have an easy way (like a trigger) to stop its execution, and also to update them - Rights in relation to automated decision making and profiling.
- Prior to any action execution, user consent, and Joint Controllership Agreement (JCA) validity should be verified.
- All actions should produce a log.

Non-related to GDPR requirements are:

- The system must be able to validate received data and return an error if some of the input is invalid.
- The system shall receive messages with a predefined structure, and answer those messages with the response from chaincode execution (structure and message definition in Appendix D).

Non-functional Requirements

In terms of Non-functional Requirements, the following were defined:

- **GDPR Compliance -** Being this the main concern of the system, it must be assured that it contains all of the required features to comply with this regulation. Since the mechanisms and system requirements for compliance are the main object of this study, those were defined along the way.
- **Usability :**
  - Interactions within the multiple components must use RabbitMQ, and messages structure, action, and response shall be specified.
  - The end users should have a front end to easily interact with the system. At joining, the users must be able to read the Terms and Conditions to join, and to easily provide consent. After joining, users must be able to easily revoke consent, and exercise rights (data access, rectification, objection, and restriction).
- **Security :**
  - All of the in-transit data should be encrypted using Transport Layer Security (TLS) protocol version 1.2 or above.
  - No Personal or Personal Identifiable Information (PII) data shall be stored without ensuring GDPR compliance.
  - Access control mechanisms (using X.509 certificates) should be put in place to ensure that only allowed parties access the stored data.

  - The peer nodes shall not keep a backup of the ledger, and any transactions' history should be erased every 6 hours[1].

- **Availability –** It is expected that BlockPrivGDPR is available at all times. Target availability is around 99.9%, which translates in less than 1 hour of downtime per month.

- **Reliability :**

  - Gateway reliability - 100% of the received messages/requests must be processed and answered (with a *success* or *failure* message). If there is a peak of requests in a given moment, the Gateway must be able to queue the messages, so the processing is done safely.
  - Blockchain reliability - the transactions shall be processed in a synchronized fashion. 100% of the chaincode invoke shall have a response indicating its success or failure. It is expected that all of the failed invokes occur due to handled exceptions, and that there are no failures due to system limitations (e.g., no resources available to process a given transaction).

- **Recoverability -** If the system crashes, it's expected that only the Orderer service recovers automatically. Peer nodes will contact the Orderer to receive a real-time copy of the ledger.

- **Scalability -** The system will be tested to allow for four peers from two different organizations, and it should meet the expected performance metrics defined.

- **Performance:**

  - It is expected that requests take no more than 2 seconds to be processed (measured from the moment the request is received to the moment the response is sent)[2].
  - If the request requires the use of Personal Data Analyser (PDA), there are no performance expectations, in terms of processing time.
  - Since there is not yet a baseline of transactions per second defined by the project, and this is very hardware dependant, benchmarking tests shall be conducted to establish the safe number of transactions per second to be accepted by the system, and the limits of simultaneous request it can process. The minimum acceptable rate is of 100 transactions per second.

- **Capacity -** It is expected that the system will be processing an average of 1000 transactions per day[3].

[1] 6 hours was defined as the maximum retention period on the Consent Note (available on Appendix B)

[2] It was found that 53% of the users would leave a website if it takes more than 3 seconds to load [59]. Since the system will allow user interactions, we want to keep the total response time under this 3 seconds threshold. By keeping the system process time under 2 seconds, we still allow for 1 second of "travel" time, to allow for HTML page redirect/update.

[3] Estimation made for the purpose of dynamical erase of data

3.3.2 Interactions and communication

Regarding the interaction and communication of the system, the following design constraints were provided by the project team:

- The system should receive and send messages using RabbitMQ, which is an open-source message broker [65], and it should rely on an open source protocol for the messages, being the chosen one Advanced Message Queuing Protocol (AMQP).
- The messages and its responses must have a defined syntax, that shall be shared with the all of the involved teams.
- It is mandatory that the messages exchanged are properly encrypted. To comply with this, the system only accepts authenticated messages (with resource to X.509 certificates), and relies on TLS protocol version 1.3.

The messages are briefly discussed in subsection 6.2.4, being the full specification and syntax available on appendix D.

Chapter 4

Research Objectives and Approach

4.1 Research Methodology

Being this work about storing personal data, the first step was to understand how it could be done, and which steps would need to be taken to do it legally, and for that and as a starting point, a complete study of GDPR had to be done.

Knowing the regulations, the next stage was to understand how Blockchain works, both theoretically and in practise, so the conflicts between the two would be uncovered.

Having identified the conflicts and challenges of implementing a Blockchain that would be compliant, it was time to explore further literature on this issue, and for that, official documentation from the European Union, and official standards and recommendations were researched.

Afterwards, it was time to look for solutions, and for already implemented Blockchain systems that considered GDPR (either all of it, or some parts), and for that Google Scholar as a search engine was used. At first, more general keywords were used, namely "Blockchain" and "GDPR", and later on, more specific keywords were added to look for more specific problems (e.g., specific rights or principles from GDPR). With the obtained results, a filtering was applied, to consider top conferences and journals. Given the fact that both GDPR and Blockchain are recent, it was not necessary to apply temporal filters.

Subsequently, the next step, was to search the available literature for comparisons on permissioned Blockchain implementations, to try to find the most suitable one, and the same methodology described above was used, only adding "Comparison" and "Permissioned" to the keywords.

Then, with this knowledge, a practical analysis (with special focus on privacy) of the best candidate (Hyperledger Fabric) was done, in order to start preparing for the development. For this, the first step was to study the documentation, and to follow the tutorials available on [31].

Finally, some more practical experiences were conducted, based on the first propo-

sition for a solution, that are described in section 6.1.

Based on the results of the tests and experiences, the technical and theoretical knowledge, a slightly different approach than the initially proposed was followed (fully explained in section 6.2), which resulted in our final system, named BlockPrivGDPR.

4.2 Objectives

Being the main goal of this work to provide a Blockchain framework that is compliant with GDPR, which should be the practical output. To do it, smaller objectives, labelled as Research Objective (RO) must be accomplished:

RO1 - Extensive comprehension of Blockchain technology, types, and implementations

RO2 - Complete study of GDPR

RO3 - Understanding of GDPR compliance issues of Blockchain

RO4 - Selection of the best implementation of Blockchain for the practical implementation

RO5 - Design and coding of the Blockchain solution

RO6 - Integration of solution with other mechanisms that enhance security and performance (e.g., PDA)

RO7 - Solution configuration and testing assurance

4.3 Approach

The approach to accomplish the stated objectives will be discussed in this section.

RO1 - To accomplish extensive comprehension of Blockchain, the study will start on a broad document developed by NIST [68] that specifies and explains how this technology works, and what types of practical implementations and mechanisms exist. After a general knowledge and understanding, specific documentation for specific implementations will be studied (e.g., Hyperledger Fabric documentation [31]), as part of the **RO4**.

RO2 - To understand GDPR, the starting point will be EU GDPR website [2], and then, the whole GDPR document will be analysed [61], additional documents or articles might be consulted for further information, or clarification.

RO3 - As an introduction, and to get a global idea of all of the compliance issues and conflicts, the starting point will be the study of Dr Michèle Frink at the request of the Panel for the Future of Science and Technology (STOA) and managed by the European Parliamentary Research Services (EPRS) [15]. Then, literature on this matter will be researched.

RO4 - To achieve this, initially literature on Blockchain comparison will be searched and analysed, and then, documentation for the best candidates will be analysed to select the most promising platform.

RO5 - By analysing the literature on Blockchain implementations, the more suitable will be chosen and deeply examined. Then, based on all the theoretical studies, and the solutions found, a practical solution will be designed, coded, and tested.

RO6 - For this goal, research on the available tools that might help to improve and expand the developed system will be done, and the selected tools will ideally be integrated with the Blockchain platform.

RO7 - Finally, both performance and compliance tests were performed, in order to fine tune the system, and to assure GDPR compliance.

4.4 Risk Analysis

While developing this paper, and the described system, some risks were identified taking into consideration some hurdles found. Table 4.1 summarizes them. Further explanation will be given below.

ID	Condition/Concern	Consequence	Mitigation Plan
R1	Not fully understand Regulations, or misinterpret it	Negligence some required steps for GDPR compliance	Read further explanations of the GDPR available, and support interpretations on available literature
R2	Limited number of Fabric deployment examples/tutorials	Increased difficulty to deploy and configure Hyperledger Fabric	Focus on the official documentation and tutorials, that may be limited, but it's trustworthy and informative
R3	Very limited number of chaincode examples that focus on privacy	Increased difficulty to design and code chaincode	Focus on official documentation to understand nuances and available functions that may help with privacy
R4	Fail to implement GDPR-compliant functions in chaincode	Platform is not GDPR-compliant	Extensive testing of any implemented feature

Table 4.1: Risk Analysis Summary

As previously mentioned, table 4.1 summarizes the risks taken into consideration during the first part of the work. For easier reference, we identified each risk with an Identification number (ID) ***Rn***, being n the sequence number it was identified.

The first risk (**R1**) was identified right in the beginning of the GDPR study, due to the formal language that the regulations are written in, and due to its general scope, which resulted in a concern regarding a misconception or misunderstanding of the stated.

To overcome this concern, additional research was done on further clarification, from European Union (EU) related sources, e.g. [19], EU related projects, e.g. [2], and individual Working Party (WP) articles, opinions, and clarifications, e.g. [50].

When we started the practical tests, the first task was to deploy and configure a Hyperledger Fabric Blockchain instance for testing purposes, and **R2** was identified. Even though the official documentation [31] is very informative and contains many tutorials, those are limited, and most do not focus on privacy matters. It was also hard to find additional support on other sources, especially regarding the version that it's going to be used (2.2).

It was found that the focus should be on official documentation, and even though some information was missing on the tutorials (mostly regarding the privacy preserving mechanisms), it was documented somewhere else.

Along with **R2**, after a Blockchain instance was successfully deployed, many questions were raised regarding the design, coding and operation of chaincode. **R3** summarizes the faced issues: limited examples on chaincode, specially the ones focusing privacy preserving mechanisms. Technical limitations also contributed, as it was the first time we were using Blockchain-related libraries and the first time working with Golang programming language.

R4 was a concern that was present throughout the whole work, since there are no platforms or tools to objectively evaluate the developed work in terms of GDPR compliance.

To try to mitigate this, we tried to analyse GDPR from all angles, and based our compliance evaluations on the documentations and guidelines we found. Subsection 7.2.6 summarizes the results regarding GDPR compliance.

Chapter 5

Analysis of Hyperledger Fabric

In this chapter, an analysis of Hyperledger Fabric will be performed. We will start by analysing the documentation to depict how this Blockchain implementation works, and its specifications.

After a more theoretical presentation on how Hyperledger Fabric works, a brief practical experimentation was performed to illustrate some key privacy features. On this chapter, only a generic overview will be presented, as the system implementation will only be specified on the following chapters.

It's important to mention that the version of Hyperledger Fabric used for this demonstration is **version 2.2**. So, all the mentioned features and mechanisms are about this version. It's also very relevant, that the **versions prior to 2.0 do not contain some of the privacy features** used and explained. For more information on versioning, and the privacy changes, it's recommended to check the official documentation, available on [27]. Regarding the version used for the implementation of the system, it is Hyperledger Fabric 2.5, as important upgrades were released during the period of implementation (more details are presented in chapter 6).

5.1 Definitions

On subsection 2.1.2, a general presentation and exploration of Blockchain was performed, however, to better understand the practical analysis, it is important to clarify some terms beforehand. Some are explained below, based on the glossary of [30]:

- **Chaincode -** Smart contract for Hyperledger Fabric
- **Peers -** Participants of the Blockchain. The major types of peers are:
  - **Committing peer -** Every participant is a committing peer, as they receive new blocks that are committed to their copy of the ledger (after validation)

  - **Endorsing peer -** Peers that execute chaincode and send the output to the client application for the validation of transactions

- **Ordering service** (or orderer) - Group of nodes that order transactions into blocks, and then send them to the peers for validation and commit
- **Message Service Provider (MSP)** - Abstract component that provides the credentials required to authenticate transactions to the clients and peers

In the case of **transaction** definition, it's more broad than the one presented before. Since the one provided in the documentation not only explains transactions but also provides a good frame for it, it will be quoted from [30]: *"Transactions are created when a chaincode is invoked from a client application to read or write data from the ledger. Fabric application clients submit transaction proposals to endorsing peers for execution and endorsement, gather the signed (endorsed) responses from those endorsing peers, and then package the results and endorsements into a transaction that is submitted to the ordering service. The ordering service orders and places transactions in a block that is broadcasted to the peers that validate and commit the transactions to the ledger and update world state."*

5.2 Transactions

5.2.1 Transactions in Hyperledger Fabric

Before reasoning with private data and chaincode calls, we must understand how contract calls and transactions work on Hyperledger Fabric [34]:

1. A client/peer initiates a transaction and sends a request to execute some function.
2. The endorsing peers verify:
   (a) The transaction proposal is well-formed
   (b) It has not been submitted in the past (to protect from replay attacks)
   (c) The signature is valid
   (d) The initiator has the required permission to execute that function
3. If all verifications are valid, the endorsers use the input values sent by the client, and execute the chaincode; sending the outputs (and their signatures) to the ledger as "proposal responses" (no updates on the Blockchain are made yet).
4. The proposal responses are inspected: the application verifies if the "proposal responses" are the same. The application inspects query responses and verifies if all endorsement policies were fulfilled, and if needed, submits the transaction to the ordering service.

5. The application will broadcast the transaction proposal to the ordering service, for a block to be generated. The transaction will contain:

   (a) The read/write sets

   (b) The endorsing peers' signatures

   (c) The Channel ID

6. The transactions are validated to ensure the endorsement policy is fulfilled, and no changes occurred to the ledger state for read set variables. Transactions in the block are marked as valid or invalid.

7. Ledger is updated with the new block.

A scheme of the transaction diagram is presented on figure 5.1:

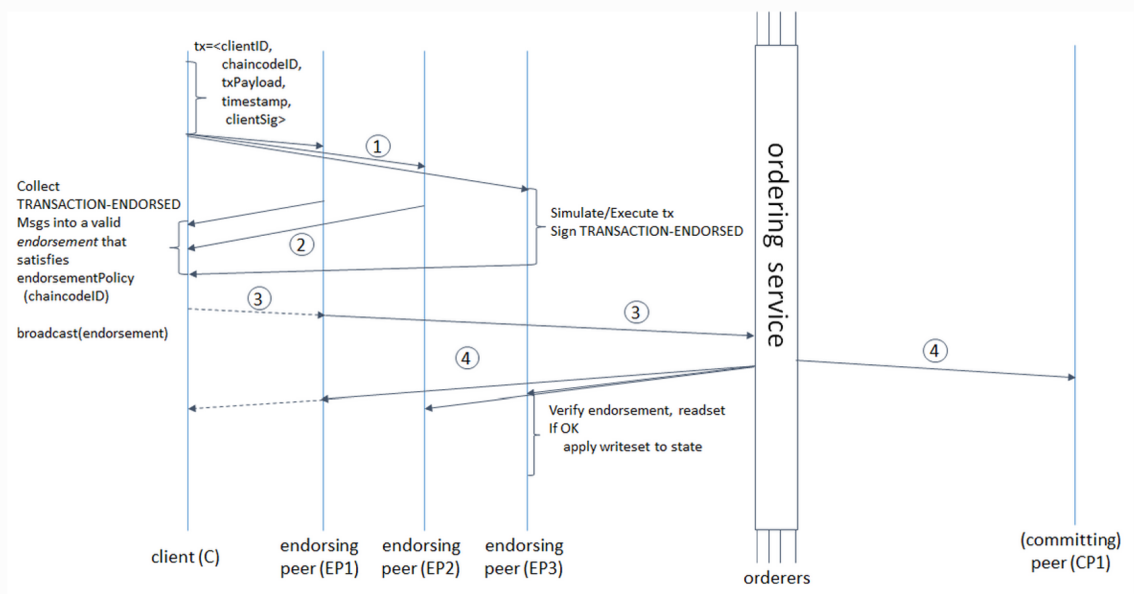

Figure 5.1: Sequence diagram of Hyperledger Fabric transaction [34]

5.2.2 Consensus and data transmission

Hyperledger Fabric supports multiple consensus mechanisms, however, the default and recommended one of the explored version is Raft, that was briefly explained in subsection 2.1.2. This algorithm follows a leader-followers approach, and is *crash fault tolerant*, which means that it can resist the loss of a given number of nodes, without losing integrity [40].

After a block is validated and committed, and as not all of the peer nodes need to be connected to the orderer [40], the peers can cascade the block information to other peers of their organization, using the gossip protocol. The gossip protocol is a protocol for data dissemination, and has three primary functions: peer detection; disseminate ledger data to outdated peers; disseminate ledger data (peer-to-peer) to newcomers.

5.2.3 Privacy concerns and privacy-preserving mechanisms

From the reading of the previous subsections and the definitions provided in section 5.1, an initial privacy concern can be identified: the transactions are validated by the endorsing peers who have access to the input values (and therefore the output values) of a given chaincode invoke.

In order to control and restrict the peers who can perform this operation, it's possible to define a custom **Endorsement policy**. In these policies, it's possible to define some privacy-related rules, like the restriction of endorsement to peers of the same organization as the requesting client. The mentioned policies can also define other types of rules [25].

For a better understanding of what information is stored on the Blockchain when a client invokes a chaincode, a practical analysis was done. For that, the tutorials found in the official documentation [31] were used to deploy the Blockchain, and for the testing, a slightly modified version of Asset Transfer Events chaincode [35] was used. This smart contract has the following functions:

- **Create Asset -** creates and stores an asset with the following properties: ID, Colour, Size, Owner, Value.
- **Read Assets -** get one or multiple asset properties
- **Transfer Assets -** transfer one asset from one to another, for a given agreed price

The chaincode was modified to store to create and store a reduced version of the asset on the Blockchain and to send an event with the full details of the asset, to all the authorized peers in the network[1].

First, an asset was created using the function Create Asset, and then the chain was analysed to check what was indeed stored, regarding the invoke. In figure 5.2 we can observe the payload of the transaction proposal, and it contains all the input values sent by the requesting client (Note: the values are stored in base64. On the left of the picture, the values are in base64, and on the right, the values are decoded to UTF-8):

[1]This version of the chaincode can be found on folder *PoC_chaincode_events* on the public repository of the project (`https://github.com/DansterPT/BlockPrivGDPR-public`)

```
"chaincode_proposal_payload": {
  "TransientMap": {},
  "input": {
    "chaincode_spec": {
      "chaincode_id": {
        "name": "basic",
        "path": "",
        "version": ""
      },
      "input": {
        "args": [
          "Q3JlYXRlQXNzZXQ=",
          "YXNzZXQxNjY2NTU1MTg4NDg5",
          "Ymx1ZQ==",
          "MTA=",
          "U2Ft",
          "MTAw"
        ],
        "decorations": {},
        "is_init": false
      },
      "timeout": 0,
      "type": "UNDEFINED"
```

```
"chaincode_proposal_payload": {
  "TransientMap": {},
  "input": {
    "chaincode_spec": {
      "chaincode_id": {
        "name": "basic",
        "path": "",
        "version": ""
      },
      "input": {
        "args": [
          "CreateAsset",
          "asset1666555188489",
          "blue",
          "10",
          "Sam",
          "100"
        ],
        "decorations": {},
        "is_init": false
      },
      "timeout": 0,
      "type": "UNDEFINED"
```


Figure 5.2: Chaincode normal invoke payload

As mentioned before, given the chaincode modification, only part of the asset is stored on the chain itself (ID, Owner, Hash), a full version is sent as an event.

Figure 5.3 demonstrates the asset data stored in the Blockchain (above in base 64, and below decoded to UTF-8):

```
"collection_hashed_rwset": [],
"namespace": "basic",
"rwset": {
  "metadata_writes": [],
  "range_queries_info": [],
  "reads": [],
  "writes": [
    {
      "is_delete": false,
      "key": "asset1666555188489",
      "value": "eyJJRCI6ImFzc2V0MTY2NjU1NTE4ODQ4OSIsIk93bmVyIjoiU2FtIiwiSGFzaCI6Ik5vdFlldEltcGxlbWVudGVkIn0="

"collection_hashed_rwset": [],
"namespace": "basic",
"rwset": {
  "metadata_writes": [],
  "range_queries_info": [],
  "reads": [],
  "writes": [
    {
      "is_delete": false,
      "key": "asset1666555188489",
      "value": "{"ID":"asset1666555188489","Owner":"Sam","Hash":"NotYetImplemented"}"
```


Figure 5.3: Normal asset data stored

Figure 5.4 shows not only the event data sent, but also that events' data are recorded in the blockchain (above in base 64, and below decoded to UTF-8):

Figure 5.4: Event data

From this experience, we can conclude that:

1. Standard invocation of chaincode will record the inputs on the chain, which means that it cannot be used, even if the assets are stored somewhere else

2. Events are recorded in the Blockchain, and therefore are not suitable as a privacy mechanism

Having this in mind, we will now experiment with the **private mechanisms** of Hyperledger Fabric.

To do it, a different chaincode will be used: Asset Transfer Private Data [36]. The functions of this smart contract are similar to the one previously used, however, it doesn't send events, and the data is stored implicitly (in access controlled databases, that are generated and controlled by the MSP).

To illustrate what's stored on-chain using the described mechanisms, an asset was created with the properties shown in figure 5.5:

```
export ASSET_PROPERTIES=$(echo -n "{\"objectType\":\"asset\",\"assetID\":\"asse
t1\",\"color\":\"green\",\"size\":20,\"appraisedValue\":100}" | base64 | tr -d \
\n)
```

Figure 5.5: Asset properties for chaincode invocation

Figure 5.6 shows that the properties of the asset are no longer stored on the chain itself, but rather a salted hash value is stored: before the calculation of the hash value, Hyperledger Fabric generates a salt value that is included as a property of the asset [26]. The goal of this randomly generated number is to add entropy to the hash value and to try to avoid "logical" brute-force attacks.

Figure 5.6: Private asset data stored

Even if the properties of the asset are not stored, if the standard invocation is used, the input values will be stored on the chain. To avoid this, the invoke must be using the **"transient"** flag. By using this flag, the input values will no longer be stored on the chain, as demonstrated in figure 5.7. Note that the input values will still be sent to the endorsing peers; so, an endorsement policy must be used to enforce restrictions on the sharing of the data.

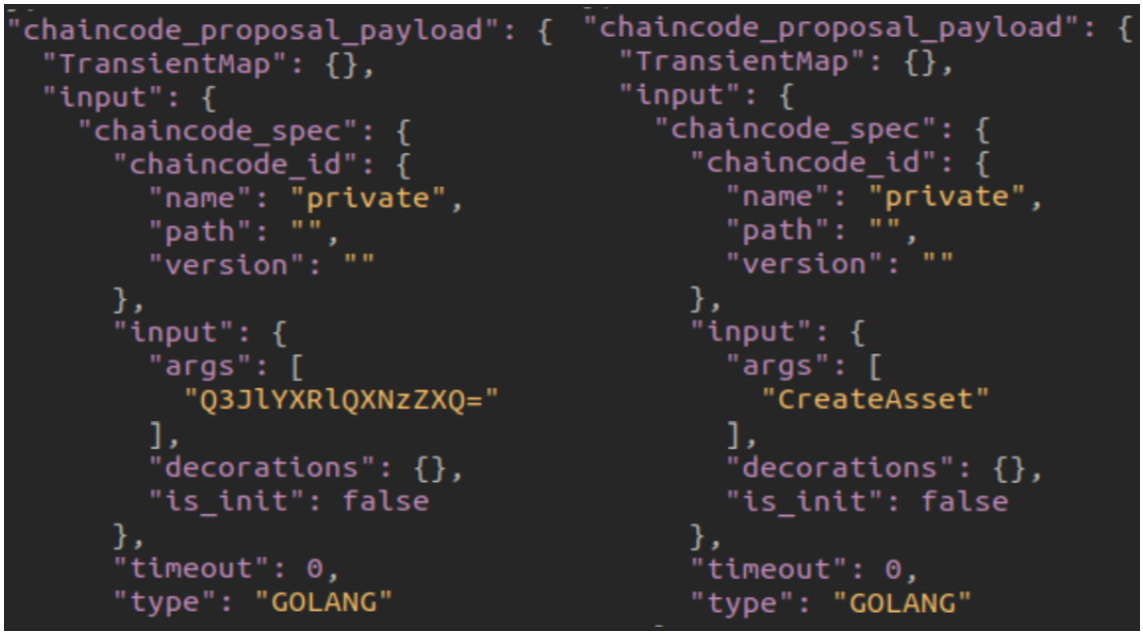

Figure 5.7: Chaincode private invoke payload

It is also possible to create hybrid assets (some properties are public, and others private), as shown in figure 5.8, the public values are stored in plain text, and salted hashes are stored regarding the private data that will be stored implicitly.

Figure 5.8: Hybrid asset data stored

Chapter 6

BlockPrivGDPR: Analysis, Architecture and Implementation

In this chapter we discuss the whole process of implementing BlockPrivGDPR: in section 6.1 a possible solution is presented based on the mechanisms explored in chapter 2. This solution was the first conjecture done regarding a General Data Protection Regulation (GDPR) compliant system.

Section 6.2 explains the whole decision process that led to the actual implemented system, and then by breaking down each component, all of the modules of the system are described.

6.1 Data privacy analysis

Based on the literature, and the found solutions and mechanisms that were described in chapter 2, we aggregated the ones that we believe that together would not only cover all of the principles and rights of GDPR, but also produce a GDPR compliant platform:

- Smart Contracts with Resource Description Framework in Attributes (RDFa) to manage the consent of each user
- Direct Channel of communication between the user and controllers
- Interactions logging Smart Contract
- Off-chain storage with on-chain integrity proof (with single-use salt)
- Expiry date on off-chain data
- Data self-destruction if nodes are offline for too long
- Digital Data Converter to enforce synchronization
- Joint Controllership Agreement as Smart Contract

- Addition of “arbitration mechanisms" to Smart Contracts

Having identified the mechanisms to be used in the system proposal, figure 6.1 depicts how the components could be deployed and interact with each other. The blue rectangle represents the Blockchain, which contains the Smart Contracts, the nodes, and the ledger. Outside the Blockchain, we have the off-chain databases, the users, the controllers, and mechanisms to control the off-chain databases.

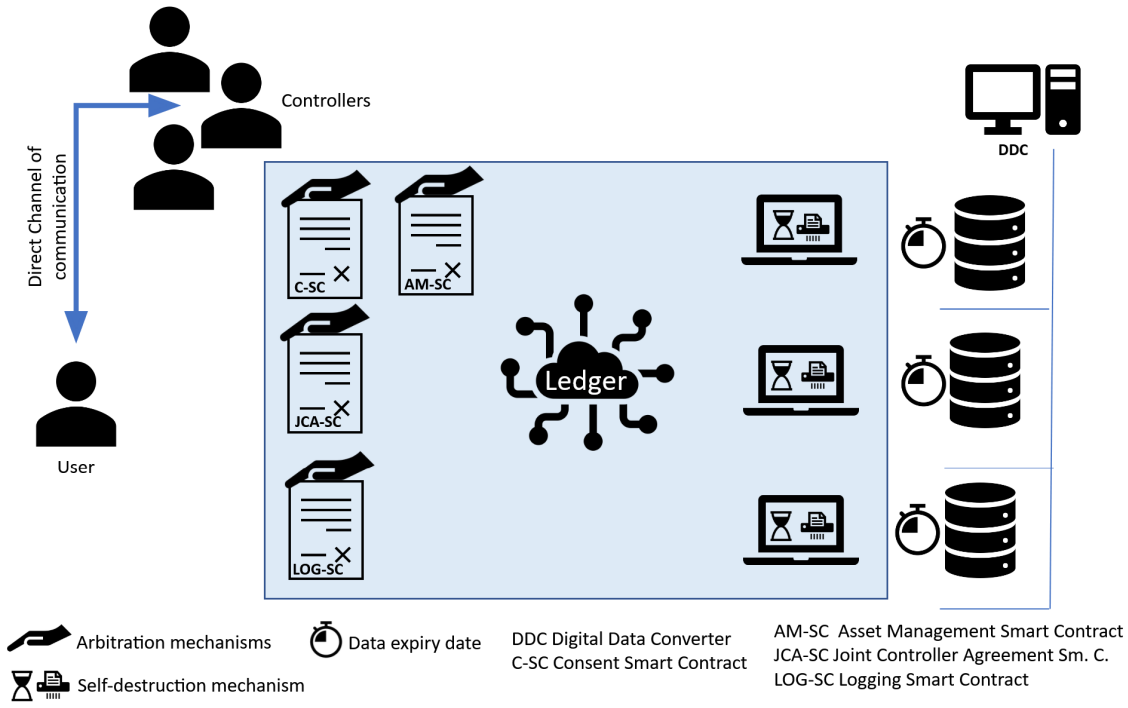

Figure 6.1: Practical scheme of the solution proposal

In appendix A a Proof of Concept based on this proposition is presented.

6.2 BlockPrivGDPR: Practical Aspects

In this section, we provide an overview of the practical implementation of the system: the decisions and compromises made, and the mechanisms actually adopted and developed.

6.2.1 Introduction

Before entering details regarding the implemented version of the system itself, it’s important to note that unlike the analysis performed in chapter 5, the final system uses Hyperledger Fabric version 2.5. This version was released during the development of this work, and contains a critical improvement on the private data deletion mechanism, as it includes the erasure of the private data history from both state and peers databases [28].

To provide a full picture of the system, we will start by explaining our decision process and the compromises that were made, which resulted in an end product that was slightly different from what was initially thought and proposed, but equivalent in terms of GDPR compliance (subsection 6.2.2). Then, in subsection 6.2.3 an overview of the architecture of the system is presented, along with a relation with the mechanisms and the GDPR principles and rights. A component breakdown will be done on subsections 6.2.4,6.2.5, and 6.2.6 with main focus on the two main components of the system: the Gateway, and the Smart Contracts, respectively. Finally, an overview of the possible actions and outcomes is provided in subsection 6.2.7, even though, those will only be shown and demonstrated in chapter 7.

6.2.2 Decision-making and compromises

In section 6.1 a possible practical solution was proposed, based on the state-of-the-art analysis, and on the mechanisms found in the analysed literature. Even though those mechanisms served as a base for our final platform, the final product was somewhat different. We will now enumerate all of the mechanisms, and how they compare to the initial ones.

To manage consent, we decided not to implement RDFa. Even though the system could benefit from it, due to technical reasons, and since a binary consent would suffice for our use cases, we decided to implement only a **Consent Smart Contract**.

The **Direct Channel of Communication** was indispensable. To make it easier for the users to use it, we implemented a form that will automatically email all of the involved data controllers (figure 7.13). Follow-up conversations will occur via direct email.

For logging purposes, the initial concept was also maintained, and an **Interactions Logging Smart Contract** was designed and implemented. This will be in-depth described in subsection 6.2.5, however, a conceptual note is that we decided to omit the record of some information, namely the ID of the asset on which the logged action occurred. From a functional point of view, this omission is a compromise, since it limits the traceability of the actions; however, from a privacy point of view, it is a feature [15].

To avoid the overhead and complexity of using an external database, BlockPrivGDPR relies on **Access Controlled Databases** natively supported by Hyperledger Fabric, as besides the main/public state database that can be accessed by any peer/node, there is the possibility to have access controlled databases, that enforce access control through the use of X.509 certificates [33]. These databases can be configured to be completely private, or with restricted access.

This decision was made based on a set of considerations:

1. From a technical perspective, it would be extremely complex to support an external database with all of the needed functions and mechanisms to

assure all of the GDPR principles.

2. From a control perspective, the inclusion of a foreign element that is not natively supported by Hyperledger Fabric, contributes to more obstacles.

3. From a setup and support perspective, the addition of all of the needed components would result in more needs from the parties that wish to join the system, and in an increased difficulty to support the system over time.

4. From a compliance perspective, the higher the number of components, the higher the number of possible points of failure.

5. Finally, from a testing perspective, a much more complex testing setup would need to be designed and put in place.

Even though we strongly believe it would be possible to implement our system using off-chain mechanisms, the lack of time and specific knowledge has made us steer in a different direction. This decision brought some benefits to our system and implementation, which will be discussed in the next mechanisms' presentation.

Another mechanism that is highly important for the system, is the definition of an **Expiry Date for the Private Data**. Since the data will be fully controlled by the Blockchain system, it will rely on the BlockToLive functionality [32], which will be further explained in subsection 6.2.5.

By relying on Fabric databases, the nodes with access to the private databases keep a temporary history of the blocks received in the form of a log file, that is used by the LevelDB. This is kept by default in Hyperledger Fabric, and it's not possible to deactivate. The goal of this history/Backup is to keep a ledger copy, and to diminish the number of queries to the ledger and to recover faster in case a node goes offline. As this poses a compliance threat, we decided to include an automated routine mechanism with the goal to **Delete the Node History/Backup Data**, that will not only periodically delete this log file, but also enforce all data deletion in case of the node being offline for too long (more details in subsection 6.2.6). It's important to note, that the due to the endorsing policies in place, there are no transmission of personal/private information from one organization to another: a peer only received personal data from its own organization.

For this scenario, we don't require an additional mechanism to enforce the node synchronization, as we can **Rely on Hyperledger Fabric mechanisms to keep the nodes data updated**.

The **Joint Controllership Agreement via Smart Contract (JCA-SC)** and the **"Arbitration Mechanisms"** were kept as initially thought of.

Figure 6.2 represents visually the implemented mechanisms and how those connect with the GDPR principles and rights.

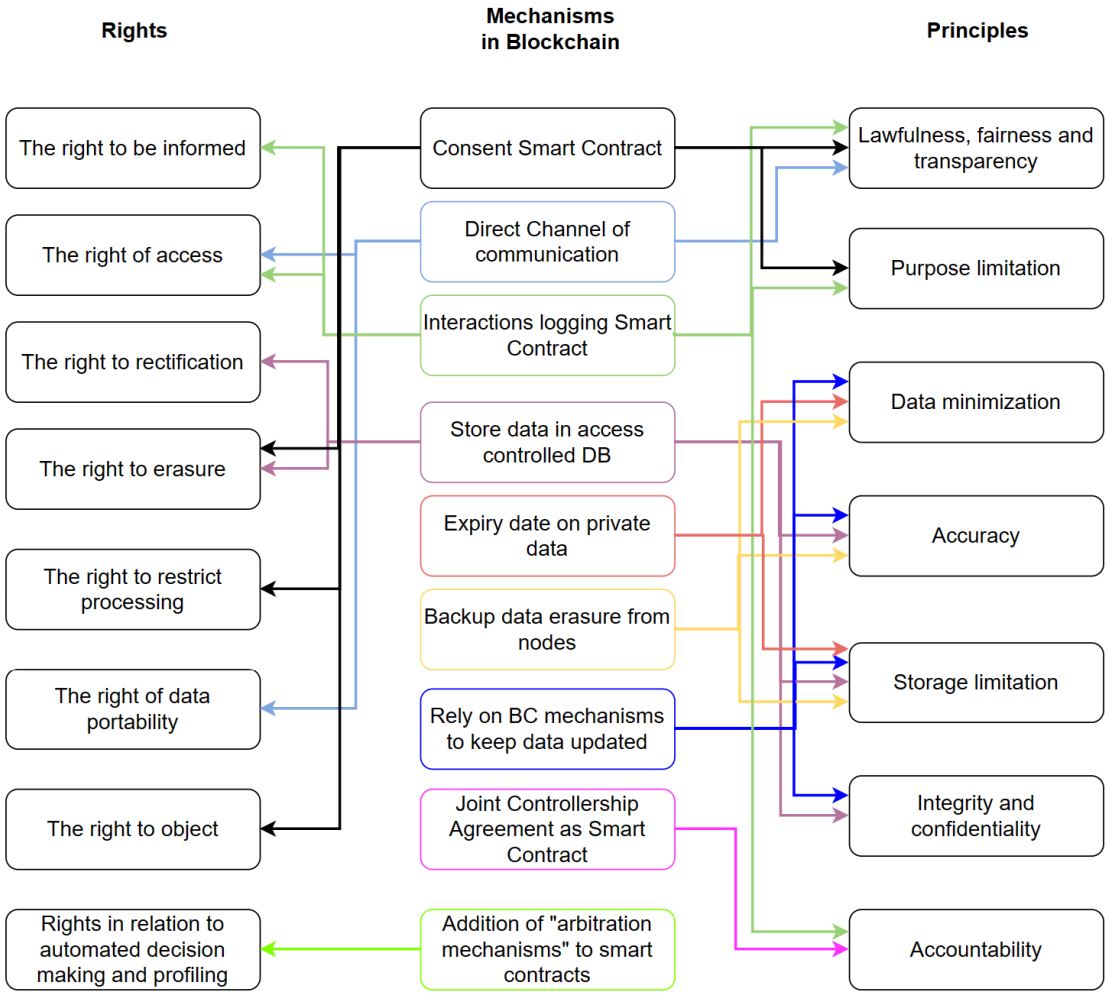

Figure 6.2: Practical scheme of the solution implemented

6.2.3 Architecture

BlockPrivGDPR has two main components that sustain the system: the Gateway, and the Smart Contracts/chaincodes deployed on the Blockchain. The Gateway is responsible for handling all of the communications between the different parts of the system and also performs some logical control operations, and some non-deterministic actions; the Smart Contracts contain all of the logic behind how each operation must occur.

In figure 6.3 the whole architecture is represented in some level of detail, where all of the interacting components, and how those interactions occur are shown. In the next subsections, both the gateway and chaincodes will be described and explained, along with the remaining components.

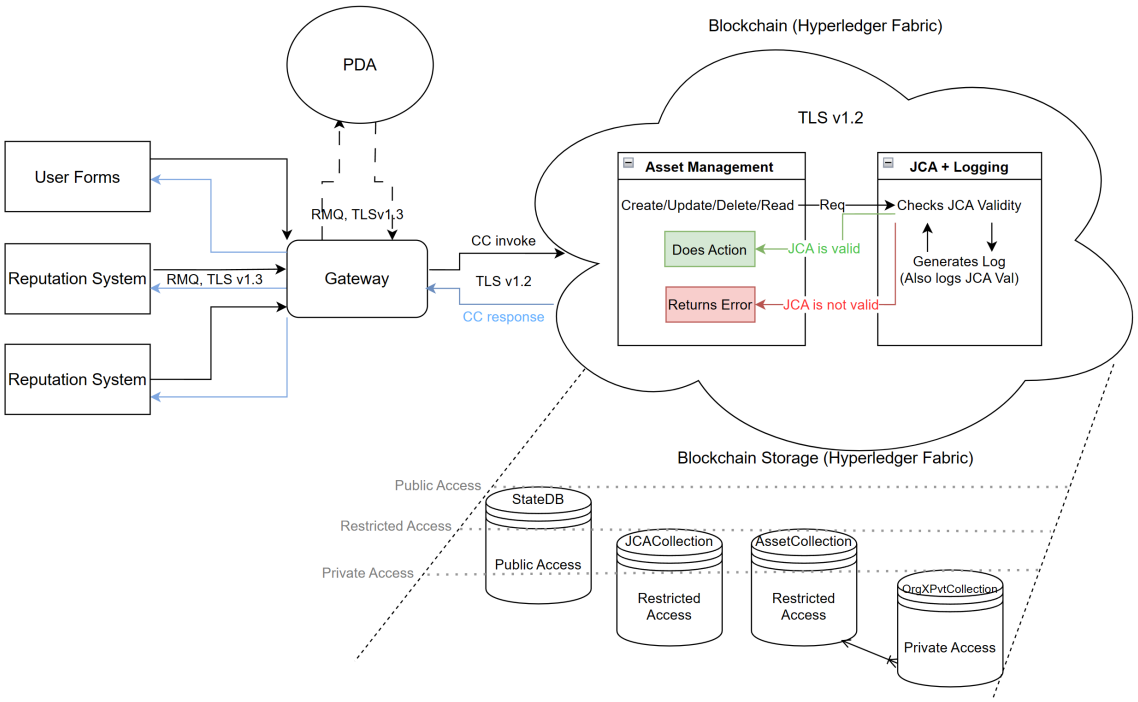

Figure 6.3: Architecture overview of implemented solution

6.2.4 Gateway specification

The Gateway module is responsible for handling all inbound messages, invoking the required methods of the implemented chaincodes, and responding to the sender of the message. There are also some logical operations that are handled by this component, due to some restrictions on the Blockchain level, namely non-deterministic operations (salt generation), and sequential control (for logging purposes).

Figure 6.4 presents a simplified version of the BlockPrivGDPR, where the main goal is to understand that the system is designed in a way that all of the interactions are done via Gateway[1].

[1]JCA Smart Contract interactions are not included in the Gateway, as those are intended for direct invoke from the Data Controllers. The process is explained in Appendix F

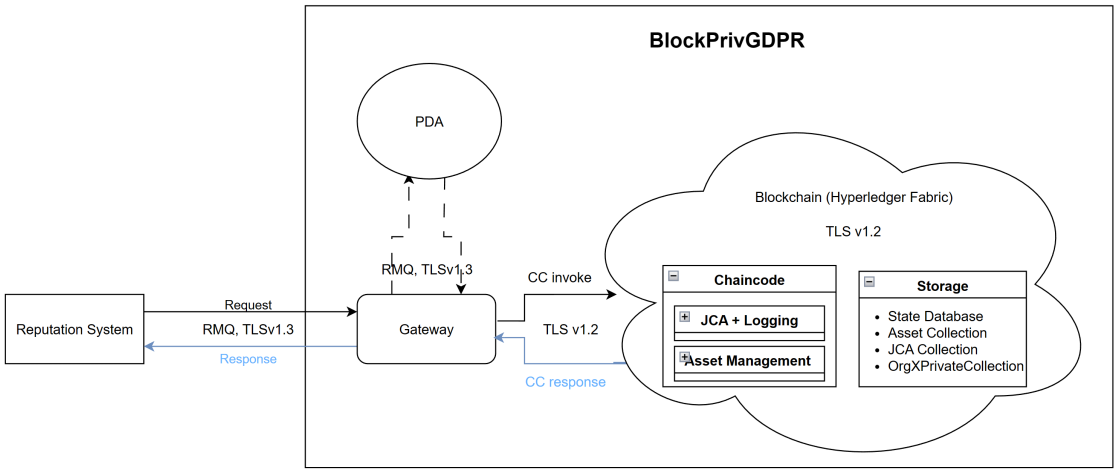

Figure 6.4: High-Level Architecture of Reputation system and BlockPrivGDPR

Being the Gateway a complex model, it was divided into two different components, the **receiver** and the **invoker**:

- The **receiver** is the module that connects to RabbitMQ queues and processes the received messages. It also interacts with Personal Data Analyser (PDA) when an *Unknown* asset is created. This module also controls the asset IDs used for logging, to ensure it is sequential, and implements the required benchmarking functions that will be mentioned in subsection 7.1.3.
- The **invoker** module is the one that connects to the Blockchain and invokes the implemented chaincodes. Since the Blockchain requires that operations are deterministic, the non-deterministic operations (e.g., salt generation) are handled by this component.

Regarding the accepted messages, and their structure, those are presented in appendix D, as those are part of a very technical aspect of the system. In a brief way, the messages accepted cover all of the actions that can be performed either by the users or the ARCADIAN-IoT components.

6.2.5 Smart Contract definition and considerations

The first step to properly define the databases where the assets are stored is to define and configure the database collections. The collections are the way we use to define the access policies of each database, and also some restrictions regarding the assets. One of those restrictions is for how long should the asset be stored in the Blockchain, and for that we use the BlockToLive property: this property defines how many new blocks must be created without asset interaction, for the asset to be automatically purged [32]. Since the requirements are for an average of 1000 blocks per day (subsection 3.3.1), we estimate that every week there will be 7000 new blocks, and defined the BlockToLive property accordingly: the JCACollection assets should exist during the whole time of the project (3 years),

while the user related asset should be erased if there's no interaction with it for a 6 months period.

As previously shown in the system architecture (figure 6.3), we have three different access controlled databases:

- **JCACollection**: Accessible by the two defined organizations. Only the members of those organizations are going to be able to validate and receive transactions. *BlockToLive*: 1092000[2].
- **AssetCollection**: Accessible by the two defined organizations. Only the members of those organizations are going to be able to validate and receive transactions. *BlockToLive*: 168000[3].
- **Org*X*MSPPrivateCollection**: Each organization has its own private collection, and only it can access the assets stored. Only the members of the corresponding organization are going to be able to validate and receive transactions. *BlockToLive*: 168000.

As for the **Asset Management Smart Contract (AMSC)**, we defined three different objects/assets:

- **AssetConsent** – the object saving the user consent (ID, ConsentFlag, BlockedFlag).
- **Asset** – the object with the user-restricted properties (ID, reputation, owner X.509).
- **AssetPrivateDetails** – the object containing the private properties of the user (ID, Name, Email).

In terms of functions, we present the most relevant ones, and a brief summary of their intended purpose:

- **CreateAssetConsent** – allows the creation of the consent object. It receives as input parameters the ID and the value for the consent flag. The blocked flag is set to false by default.
- **ReadAssetConsent** – queries a consent object using the ID as input.
- **UpdateAssetConsent** – allows the update of the consent object. It receives the ID and the value for the consent flag as input parameters. If the user changes their consent to false, it will call the function DeleteAsset to delete the user object, if it exists. If a user blocks their consent, the blocked flag will be set to true, however, no changes to the user/score asset are done.

[2] 7000 *(blocks per week)* x 52 *(weeks per year)* x 3 *(years)*
[3] 7000 *(blocks per week)* x 4 *(weeks per month)* x 6 *(months)*

- **CreateAsset** – allows the creation of a user object. The parameters are sent using the transient flag (so the input values are not recorded on the chain). It calls the ReadAssetConsent function to check if consent has been provided by the user. The field “Owner" of the asset, is populated with the information regarding the X.509 certificate of the peer that invokes it.
- **DeleteAsset** – allows the deletion of a user object. Like the previous function, it has no input parameters, as the ID is sent using the transient flag.
- **ReadAsset** – queries the restricted data of the user object. Receives the ID as an input parameter.
- **ReadAssetPrivateDetails** – queries the private data of the user object. Receives the ID and the name of the private collection[4] the object is stored in as input parameters.
- **UpdateAssetScore** – updates the reputation score of a given user by receiving the ID, and the new reputation score.
- **UpdateAssetPersonal** – updates user personal data of a given user by receiving a given ID, name, and email.

Regarding the **Joint Controllership Agreement Smart Contract (JCASC)**, two objects are defined:

- **JCAConsent** – the object saving the organization consent (OrgID, ReadFlag, ConsentFlag).
- **LoggingAsset** – A logging object, that allows to store all interactions of the peers with the data (LogID, action, OrgID, JCA validity, peer X509).

This smart contract, presents the following functionalities:

- **CreateJCAConsent** - the function to be invoked to create a JCA Consent object, required by each joining organization to be able to use the AMSC functionalities ("to process data").
- **CheckJCAConsent** - queries the JCA Consent object, and calls the CreateLogging function. This is the function called by every method of AMSC prior to action completion.
- **CreateLogging** - creates a logging object, based on the action of the AMSC, and the response of the checkJCAConsent.
- **ReadLogging** - queries logging objects by ID.
- **ReadLoggingByRange** - queries an array of logging objects based on the provided IDs range.

[4]The private collection name is inserted by the Gateway invoker.

6.2.6 Other components

In this section, the remaining components of the BlockPrivGDPR system are briefly described, namely the **PDA** and the PDA interface, the **erasure script**, the **weekly control script**, the **user interaction forms**, and the **arbitration mechanisms**.

Personal Data Analyser

The PDA is an Artificial Intelligence (AI) driven solution that aims to detect Personal Identifiable Information (PII) on given input strings [58]. This tool can help improve the security and privacy of the system, by detecting wrongly labelled personal data before it is stored as non-personal data, and it's available as an open-source project under Apache 2.0 Licence [44].

For our system, beside the user and reputation information, and since it is part of a wide network, we wanted to give the possibility to use BlockPrivGDPR to store assets in the Blockchain that are not fully labelled, meaning that anything could potentially be stored in them.

Since we are allowing this kind of storing, we need to assure the privacy of the sensible or personal data, and for that, we use the PDA. The PDA was developed in a previous project and is coded in a different language (Python), and uses a specific protocol [44] that we do not have access to, we needed to develop a small program that created the required bridge between our Gateway, and the PDA, that will be referred to as *PDA interface*.

To maintain the communication standard, the PDA interface communicates with the Gateway via RabbitMQ, and to avoid changing the core of the PDA, the information is sent via text file. The Gateway will write the information to a file, it will then send a message to the PDA interface with the name of the file, and the PDA interface will call the main module with the file as argument. After the processing and evaluation of the strings sent, a message with the result of each analysed string is sent back to the gateway, that will use to store the asset attributes in restricted or private databases, accordingly.

Figure 6.5 illustrates in a very general way how the interaction between the Gateway and PDA happens.

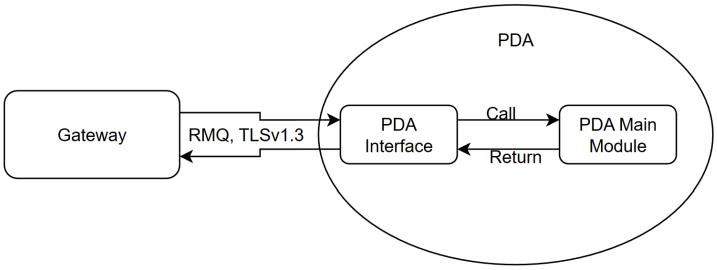

Figure 6.5: General interaction scheme of the Gateway and PDA

Erasure Script

The erasure script is a simple Linux Script set to run every six hours[5] in each of the participant nodes of the Blockchain. The goal of this script is to delete the Blockchain history log files generated when the nodes receive new block information, as explained in subsection 6.2.2. Besides the mentioned operation, this script will also perform a ping to the Blockchain orderer, and in case this operation fails four consecutive files, it will delete all peer data.

As this log files are also likely related with LevelDB operation, in large networks (with more peers and/or organizations), it is possible that this erasure creates can create some synchronization issues, or data loss. We tested the behaviour of the system with erasure enabled, and we did not find any issues. However, it will be suggested as part of the future work to do further testing to this tool. We could not find any specific documentation or official guideline regarding these files.

To schedule the periodical run times, we use *cron* utility included with Linux Operating System (OS) [8].

Weekly Control Script

The erasure script is a simple Linux Script, set to run every Friday, using cron utility, that has the goal to enforce the weekly average of produced blocks is met, so the system can assure the dynamic personal data erasure. To do so, the script runs a .go file that obtains the number of Blockchain generated blocks in the past 7 days, and in case it is under 7000, it will request the creation of 7000-<*number of actual created blocks*> of a specific mock object of the Blockchain, by sending the requests to the Gateway.

User Interaction Forms

The User Interaction Forms are very simple HTML forms that allow users to interact easily with BlockPrivGDPR. These forms allow users to provide consent about their data processing, update their data and consent, request all their data, and send a direct email to all of the data controllers. After the introduction of the data by the user, a message is sent via RabbitMQ to the Gateway, and once a response arrives, the form is redirected to a new page where the result of the request is provided. The forms are shown and detailed in subsection 7.2.1.

Arbitration mechanisms

As per our analysis, for the Smart Contracts to be compliant with the article 22 of the GDPR (regarding automated decision making and profiling), the system

[5]The six hours time period was defined as being the maximum retention period of the node backup data, under the consent note provided to the user

would need to have means for humans to pause, alter, or remove the automated software running [18].

In order to change a running chaincode, we can use the built-in Hyperledger Fabric mechanisms, that allow for new versions of the chaincode to be proposed, and if accepted and validated by all of the participating organizations, it will be installed onto the channel, and the new version will be executed on chaincode invoke [37]. To stop, or delete the chaincodes, as those run on independent docker containers [37], those can be stopped, or killed respectively, causing further invokes to fail. The number of running containers will depend on the endorsement policy of the chaincode.

6.2.7 Actions and outcomes

In order to explain visually how the system works, we created diagrams demonstrating the flow of possible actions. Before each one, we will briefly describe it, and provide some remarks regarding the possible outcomes.

Figures 6.6 portrays the consent creation flow. When a consent creation message arrives to the Gateway, it will invoke the corresponding method on the AMSC (1.), here, before any operation is done, the JCASC is invoked, to check if the organization trying to do the operation has a valid JCA in place (2.). Independently of the JCA validity, a log of the action is created and stored in the JCACollection database (3.,4.,5.). Then, the JCASC returns the validity boolean response to the Create Consent function (6.). If the response is negative, no further operations are performed, and the failure response is returned to the Gateway (8.); however, if the JCA is in place, and there is no consent for that user already, the consent asset will be stored on the StateDB (7.) and the response is returned to the gateway (8.).

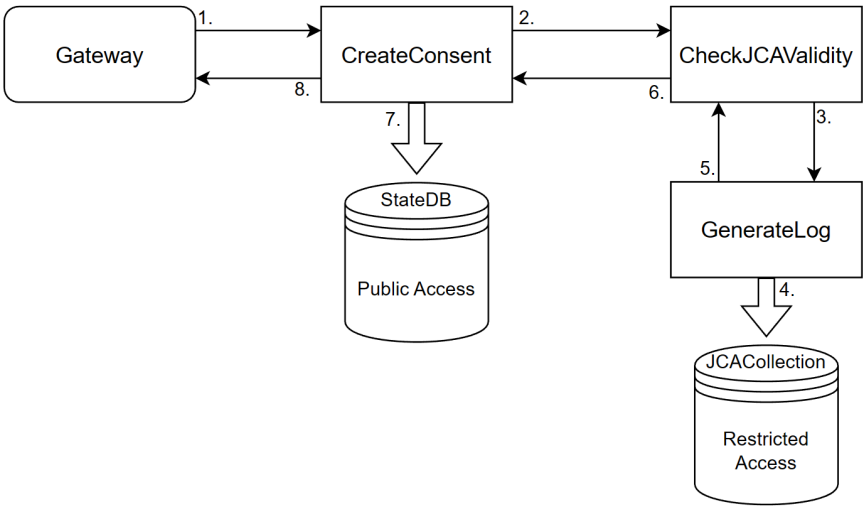

Figure 6.6: Consent Creation Flow

Figure 6.7 pictures how a User/Score asset is handled. It is very similar to the consent creation flow, with two major differences: the user consent is also veri-

fied (steps 7., 8.), and in case it doesn't exist, or its status doesn't allow for data processing, the operation is not performed (step 9. is skipped); the other difference is regarding the storing/update of the asset, as half of it is stored in the private organization database, and the remaining stored on the Asset Collection database (step 9.).

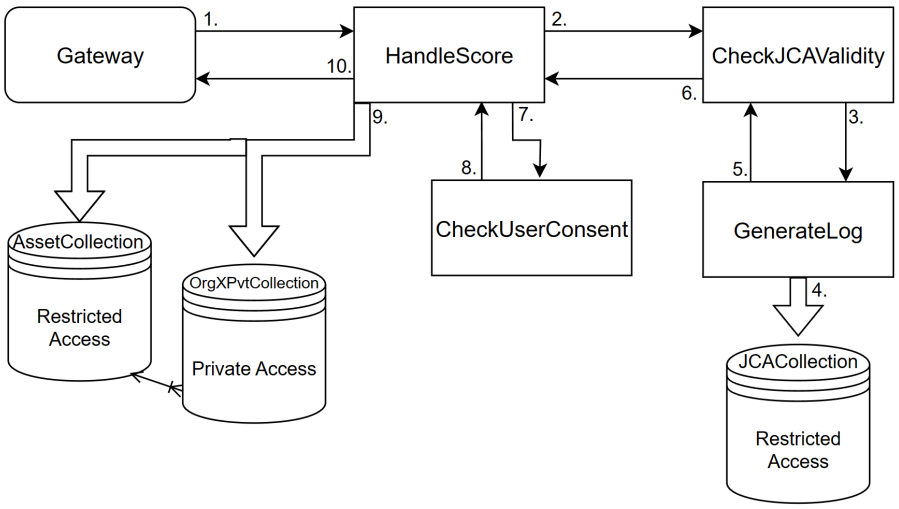

Figure 6.7: User/Score Asset Handling Flow

Finally, the *Unknown* asset creation flow is depicted in figure 6.8. The main difference between this flow and the first presented, is that prior to the chaincode invoke, the PDA is consulted in order to increase the confidence of lawfully storing the data as private/restricted (steps 1., 2.).

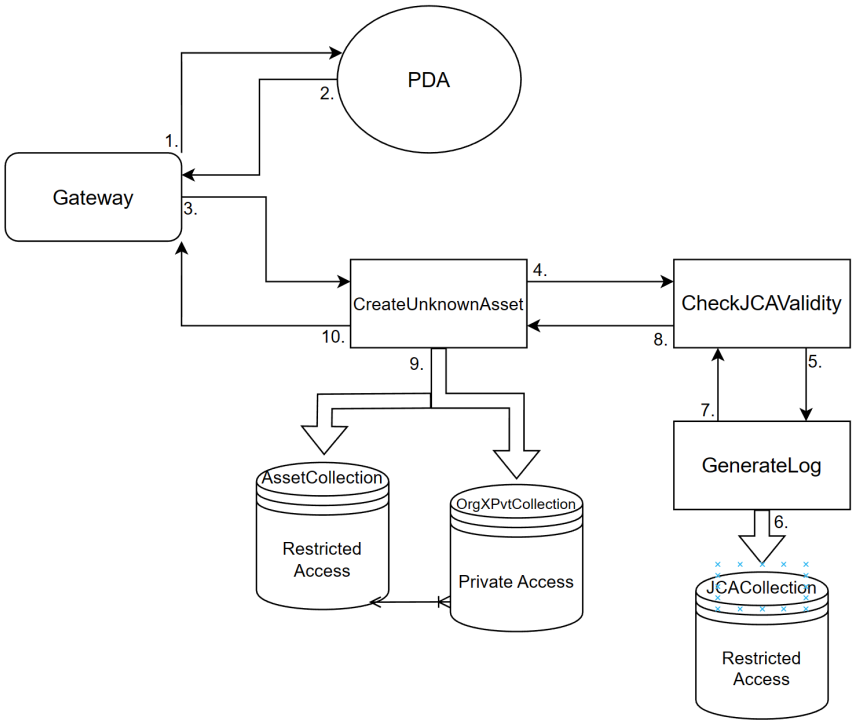

Figure 6.8: Unknown Asset Creation Flow

Chapter 7

Setup and Evaluation

7.1 System setup

In this section, we will discuss how the Blockchain and the BlockPrivGDPR system were setup and configured. Performance and benchmark test will be performed and discussed, that will help to further optimize the configurations.

7.1.1 Blockchain deployment and assumptions

For Blockchain deployment, we used the provided scripts by Hyperledger Fabric official tutorials, launched along with Certification Authorities. Two different organizations (*org1*, *org2*) are created, and the peers (with corresponding X.509 certificates) are launched as Docker containers. The default LevelDB database is used, for security reasons.

As the Blockchain deployment and configuration is not the prime focus, and the official scripts provided all we needed for testing purposes (namely server and client side Transport Layer Security (TLS) authentication with a Certification Authority (CA) for each organization and orderer, with automated certificate generation, and multiple organization setup) [38][33], we only adjusted some of the configurations according to the security recommendations of M. Lagarde [46], discussed on subsection 2.2.2. Also, some further performance related configuration changes that will be discussed below (subsection 7.1.4).

Finally, after the procedure of Blockchain deployment, we proceed to deploy both Joint Controllership Agreement Smart Contract (JCASC) and Asset Management Smart Contract (AMSC): to do so, we have to load the chaincode definition to the Blockchain, and both organizations need to approve it. After approval, the chaincode containers are instantiated, and those become available for invoke.

Once everything is set, both organizations should officialize their consent to the

Joint Controllership Agreement (JCA). As the JCASC is meant to be interacted with by the Data Controllers only, this chaincode interaction doesn't occur via the Gateway. To do so, the administrators of the organization, shall do an invoke to the CreateJCAConsent using their X.509 certificate and private key, so a JCA-Consent object is created that can later be used to verify that the organization is part of the JCA (demonstrated in appendix section F.1).

7.1.2 System and Blockchain monitoring

For the monitoring of the system, we rely on the combination of *Prometheus* and *Grafana OSS*, that allows us to visually control what's going on our Blockchain system.

Prometheus is an open-source project that is used to collect and store metrics of a given system as time series data, with optional key-value pairs. It is built using an HTTP pull model, and generally implemented for system monitoring and alerting [54].

Grafana OSS is a data visualization and monitoring open-source solution, that allows for the creation of dashboards that support multiple data presentation mediums (e.g., charts, tables, timelines) [22]. Grafana supports integration with many different data sources, being Prometheus one of them.

As per deployment of both pieces of software, we relied on the sample provided by Hyperledger Fabric [42], that includes a configuration for both services under a docker container. Then, we added some further components to the sample dashboards, based on the metrics reference available on the official documentation [39], that would allow us to visualize some further information, namely the rate of Transactions per second (TX/s), and the number of completed and requested transactions.

Our configured dashboard allows for Blockchain monitoring, and also for general server monitoring. In figure 7.1 some general information of the computer resources are shown: on the left-hand side a timeline of CPU and RAM usage is presented; in the middle, the number of docker containers running is shown; on the right side we can monitor the current disk occupation and memory usage. Figure 7.2 demonstrates some Blockchain related time-based graphs, where we can easily check for timing and analytic metrics.

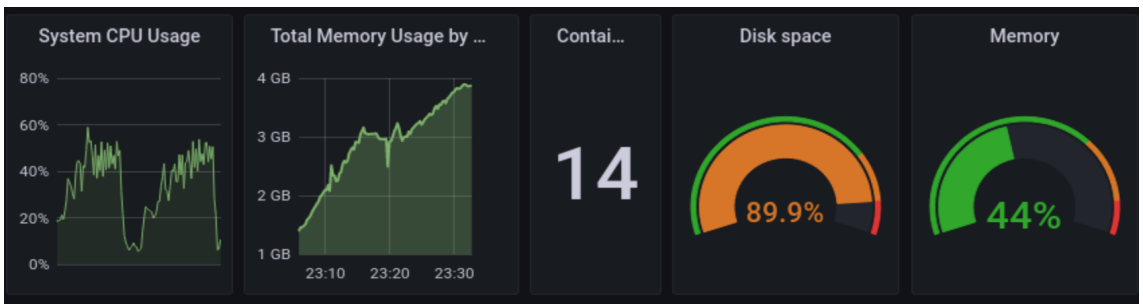

Figure 7.1: Grafana Dashboard General System Information

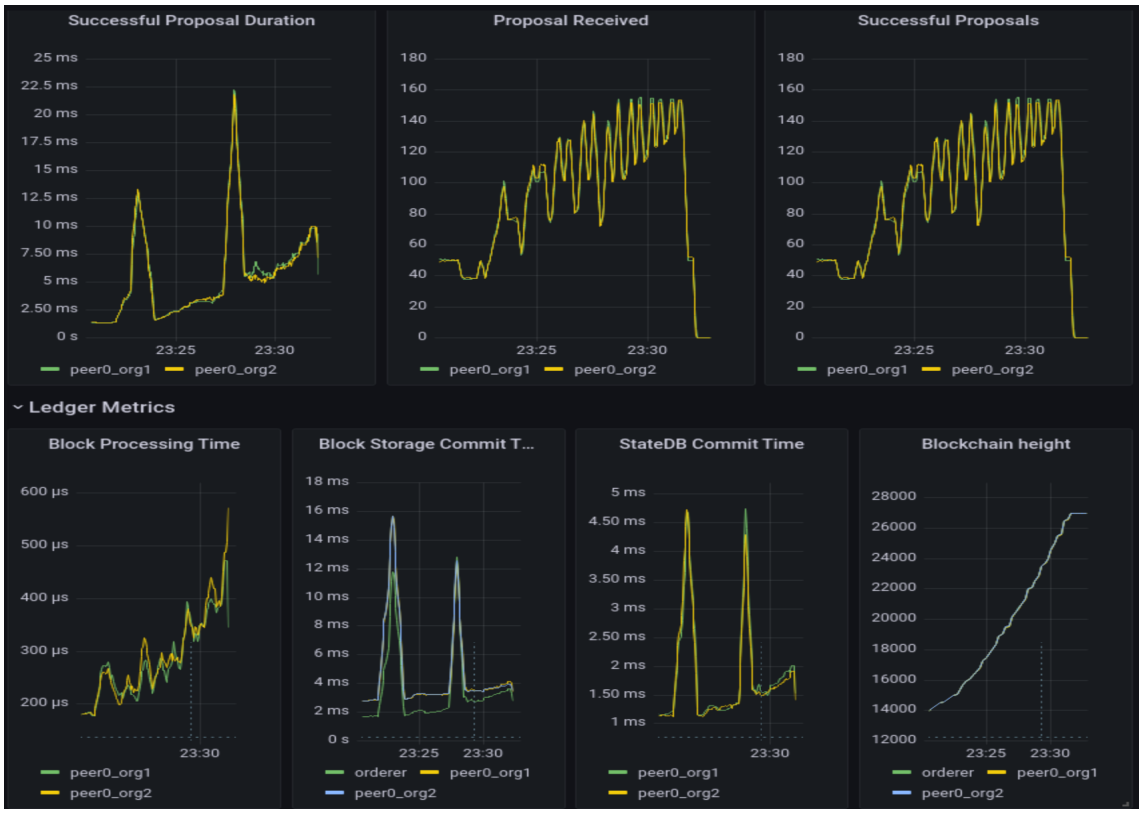

Figure 7.2: Grafana Dashboard Specific Blockchain Statistics

7.1.3 Benchmarks

To test and deploy the designed system, we used an Ubuntu Virtual Machine with the following resources:

- CPU - AMD Ryzen 7 6800H (8 vCPUs)
- RAM - 8 GB DDR5-4800MHz
- Disk - SSD NVMe 40 GB

For a baseline test, we used Hyperledger Caliper, which is a blockchain performance benchmark framework [9] that supports multiple permissioned Blockchain implementations, being Hyperledger Fabric one of them. To make this baseline test as standard and comparable as possible, instead of using our own chaincodes, we used *Fabcar* chaincode, available at Hyperledger Fabric Samples GitHub repository [16].

The reason why we chose this chaincode, is related to the number of references in the literature that use it to perform benchmarks to the Blockchain system using Hyperledger Caliper [63], or that use *Fabcar* as a base for the developed smart contracts [60]. We also found this specific chaincode being used for test and development of new benchmark tools [4]. It's important to note that even though there are some tests in the literature that use the same setup as ours, those are

not comparable, since the tests are very hardware dependant, and the hardware is different, or not disclosed.

For this baseline test, we tested three different functions of the Fabcar chaincode:

- **CreateCar -** Function that creates a car asset in the State Database
- **ChangeCarOwner -** Function that queries a car by ID, changes its owner, and updates it in the State Database
- **QueryCar -** Function that queries the State Database and returns the car asset if it exists

The goal of this test would be to obtain the average latency of the system, the throughput (number of TX/s it could handle), and the reliability (all of the submitted transactions should be valid, so any failed transaction is related with the system itself).

Due to the nature of the testing, we split it into two tests:

1. Asset Creation and Update:
    - For this test a fixed number of transactions was defined: 10000 creation requests, and 5000 update requests[1]
    - A fixed rate for the transactions was set (starting at 100 TX/s, and ending at 650 TX/s [2])
2. Asset Query:
    - For this test, we defined the transaction time limit (starting at 10 seconds and ending at 100 seconds)
    - Hyperledger Caliper would then try to perform as many queries per second as possible

Besides the mentioned configuration parameters, another parameter that impacts the test is the number of *workers* for Caliper. The number of workers is the number of simultaneous processes that run in parallel to create the desired number of requests per second. This parameter was mostly set at 5, but for some of the tests it was increased, to try to reach the desired sent transactions rate. Below on the results presentation, the name of the test will indicate how many workers were used by Caliper.

[1] In order to perform a good benchmark, we need to submit multiple transactions simultaneously [41]. We also need to send a higher number of transactions than the maximum number of transactions a block can hold. To understand system stability under test, the system needs to be evaluated generating multiple blocks. The values chosen are a compromise between the best practises stated, the time to perform such tests, and also the amount of generated output (as we have limited storage space for the Blockchain itself)

[2] These values are based on previously done sample tests, that provided some insight of the system capabilities

After running the tests, Caliper generates an HTML report that contains a table with a summary of the performance metrics, and also the details of the test, as shown in figure 7.3.

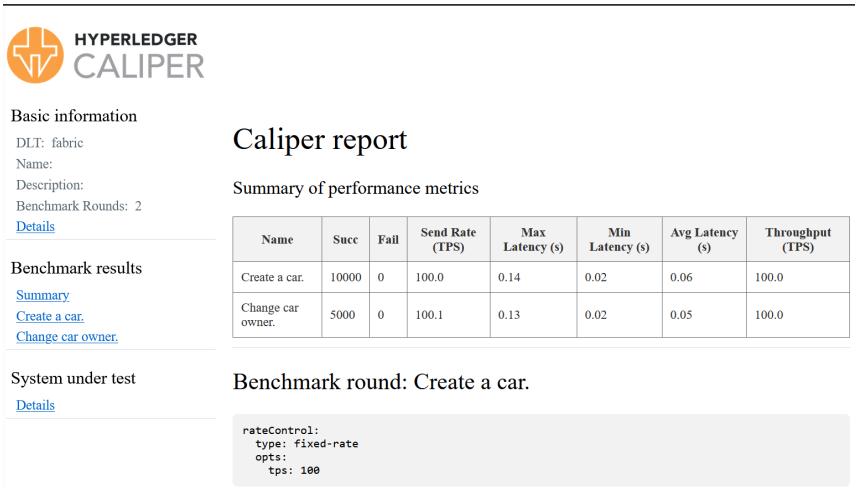

Figure 7.3: Hyperledger Caliper report example

It's important to note that in this test, the transactions are **asynchronous**, and the latency is the time difference between the time the request was submitted and the time the reply is received [20]. Another important remark is regarding the test name, it always follows the same structure: 'T' + '<target of TX/s to send>' (+ '_<number of workers>'). If the number of worker is omitted, it means that the test is using the standard of 5 workers. Lastly, we are running Caliper and Fabric on the same virtual machine: even though they are running in different Docker containers, both are sharing the same hardware.

The test results are presented below in table and chart format, where a brief analysis will be done after each pair is presented. All tests were performed five times, and the presented results are the average values, along with the standard deviation of the throughput and latency.

Create a car						
Test	**Send (TX/s)**	**T.put (TX/s)**	**St.Dev**	**Lat (s)**	**St.Dev**	**Reliability**
T100	100,00	100,00	0,00	0,06	0,00	100%
T200	200,07	200,00	0,00	0,05	0,02	100%
T300	300,10	277,82	6,50	0,05	0,02	100%
T350	349,98	325,60	16,78	0,05	0,02	100%
T400	400,12	371,06	17,43	0,04	0,00	100%
T450	450,04	403,28	12,66	0,12	0,14	99,94%
T500	499,52	450,20	8,00	0,08	0,05	99,99%
T550	517,72	458,86	51,71	0,10	0,07	99,97%
T600_5	588,86	496,56	36,38	0,16	0,13	99,95%
T600_6	508,40	489,30	67,81	0,22	0,07	99,94%
T650_6	556,06	484,96	30,40	0,40	0,28	99,93%

Table 7.1: Create a Car Hyperledger Caliper benchmark test results

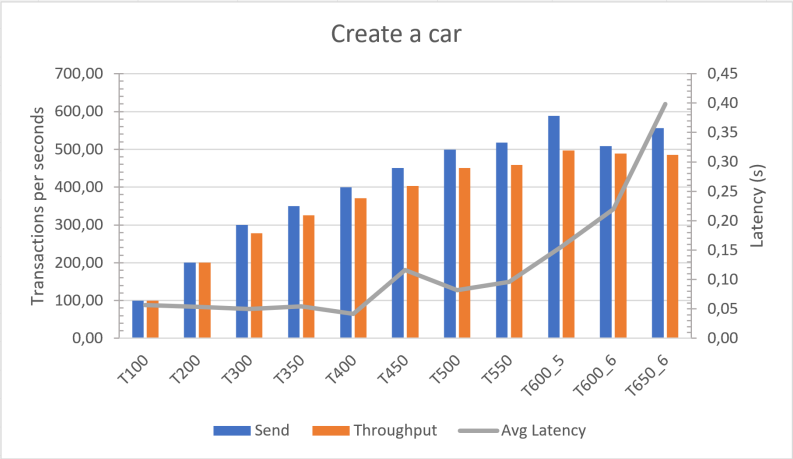

Figure 7.4: Bar chart of Create a car benchmark results

In terms of testing our higher target was to have a send rate of 650 TX/s, however, due to system limitations, even with a bigger number of workers, we couldn't even reach a stable rate of 600 TX/s. Starting by the transactions per second sent *versus* the actual throughput, we can see that starting at 300 TX/s the Blockchain is not able to keep up with the sent rate, even though, the throughput, keeps growing until the T550 test. We can see that on the last five tests, the results are in the [450;500] interval, which indicates that this would be the threshold for this kind of transaction. In terms of standard deviation, the bigger values are found in the last tests, where the system is less stable.

The average latency starts to be more noticeable from test T550, where we also see a bigger value for the standard deviation, which confirms that the system is more unstable when is trying to handle a bigger rate of transactions. Test T450 is worth mention as an outlier, with bigger than expected average latency, and the second bigger standard deviation value, this has occurred due to a higher average latency result on one of the tests, as one of the transactions timed out.

In terms of reliability, starting at test T450, we start to encounter failed transactions, and the reliability shows a decreasing tendency with the TX/s rates increase.

Change Car Owner						
Test	**Send (TX/s)**	**T.put (TX/s)**	**St.Dev**	**Lat (s)**	**St.Dev**	**Reliability**
T100	100,10	100,00	0,00	0,06	0,01	100%
T200	199,97	199,80	0,08	0,05	0,00	100%
T300	299,92	293,12	12,76	0,04	0,00	100%
T350	349,64	315,86	25,27	0,05	0,02	100%
T400	400,00	340,42	20,52	0,07	0,06	100%
T450	449,74	378,14	5,62	0,06	0,02	100%
T500	497,96	402,92	23,59	0,06	0,02	99,99%
T550	549,06	435,20	27,31	0,08	0,04	99,99%
T600_5	594,98	494,16	45,65	0,11	0,04	99,93%
T600_6	541,57	443,67	38,06	0,22	0,15	99,95%
T650_6	577,00	468,84	30,53	0,58	0,26	99,96%

Table 7.2: Change Car Owner Hyperledger Caliper benchmark test results

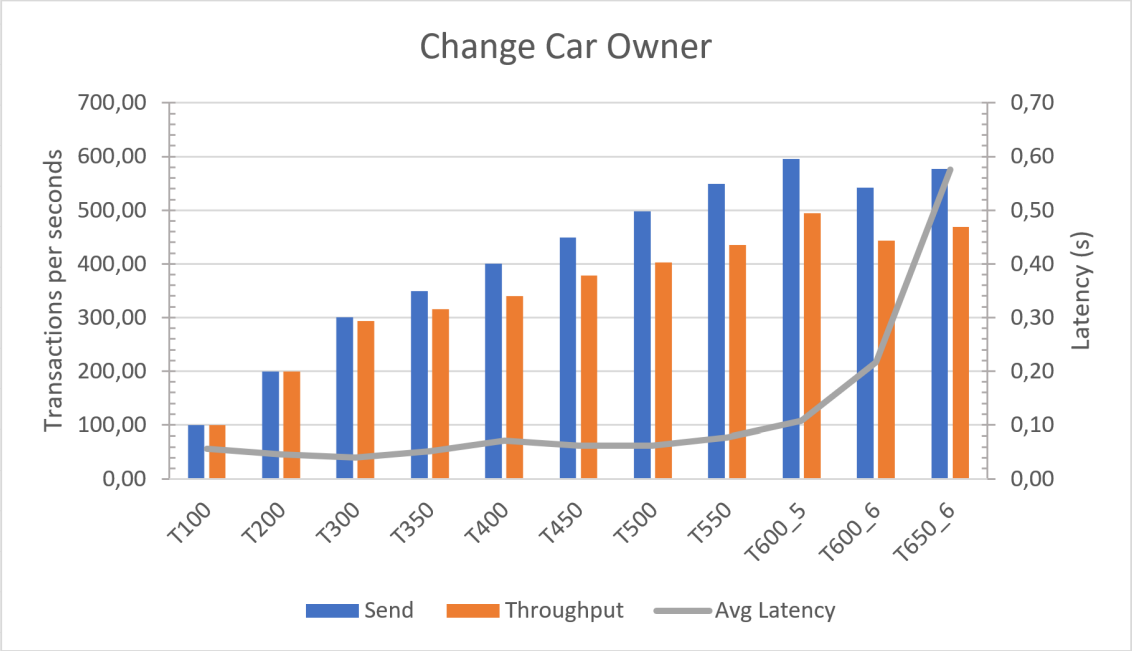

Figure 7.5: Bar chart of Change car owner benchmark results

The results of this test are very similar to the first one, and we can observe the same tendencies in all of the parameters analysed, and in this test no outliers are worth mentioning. Comparing both tests, on the first one we reached a higher, but very similar, maximum average throughput (496,56 *vs.* 494,16 TX/s), and the second test shown a higher average latency (0,40 *vs.* 0.58 seconds).

Beside the presented analytical results, another thing that we kept in mind was the system stability: we added a panel to the previously presented Grafana dashboard, that allowed us to visualize the rate of transactions the system was handling at each second. When the system is stable, we can see a constant rate of transactions being processed, however, when it starts to reach its maximum capacity, we can see that the transaction rate becomes unstable. Figure 7.6 presents

the timeline of transactions rate being handled by the Blockchain during part of a round of Hyperledger Caliper tests, and we can see that with the increase in the rate of the requests, the processing becomes more unstable.

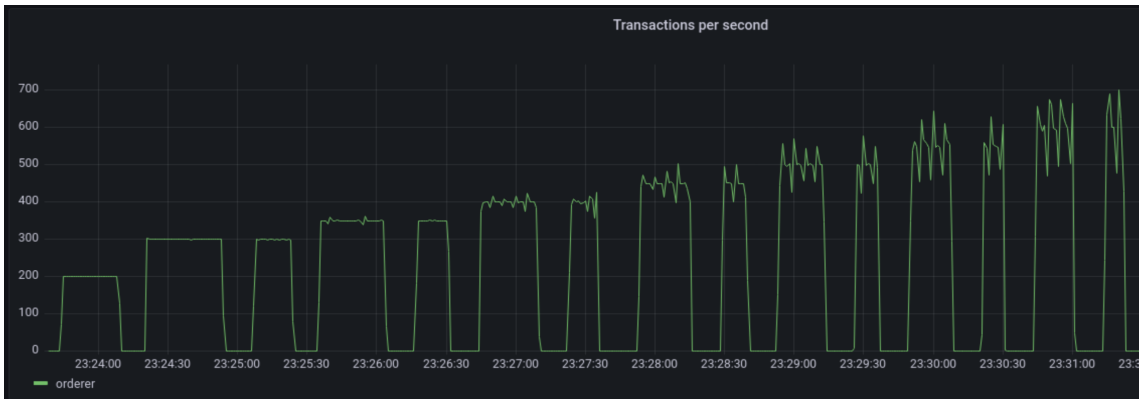

Figure 7.6: Throughput (TX/s) of the Blockchain in a round of Hyperledger Caliper

Regarding the querying tests, the results are quite different from the shown above: since the queries are simply the fetching of a value from the database, those translate into requests and not into transactions, which means that the overhead of generating and validating a transaction is not present, neither is block generation. Having this in mind, the throughput was always very close to the sent rate in all of the generated tests, and no significant difference was found between the tests time. Table 7.3 summarizes the querying tests done.

Asset Query		
Average Sent Rate	**Average Throughput**	**Average Latency**
984,96	984,78	0,01

Table 7.3: Simple query Hyperledger Caliper benchmark test results (average)

Based on these tests, we could reason that our Blockchain system is able to manage a transaction rate of up to 400 TX/s without compromising in terms of performance and reliability; however, these tests do not reflect our actual chaincodes.

So, after establishing a baseline of the performance using Caliper, we added some performance metrics to our modules, in an effort to measure some performance indicators of the AMSC, and the performance limitations of our system. Unlike Caliper, all of AMSC transactions are synchronous, and the time measurements are made on the Gateway module and not directly on the Blockchain, which will increase the latency; however, all of the chaincode invokes are made via Gateway on our system, and therefore we are measuring the perceived latency.

Two important remarks are: in our tests we are able to reach the desired 650 TX/s send rate, as we are using goroutines (which are more efficient and faster than Java threads [62]); and we are only able to evaluate the throughput within seconds precision, meaning that we always had to round up to the next second

when calculating the throughput, which can translate in a slightly higher actual rate than the calculated one.

As presented with the first batch of tests, both a table and a chart will be presented with each test results, and a brief commentary done after each pair:

Create unknown Asset (not Caliper)				
Test Name	**Send (TX/s)**	**Throughput (TX/s)**	**Avg Latency (s)**	**Reliability**
T100	100	99,74	0,26	100%
T200	200	198,45	0,39	100%
T300	300	295,19	0,55	100%
T350	350	331,13	0,63	100%
T400	400	386,77	0,86	100%
T450	450	434,78	4,75	100%
T500	500	476,19	7,37	100%
T550	550	500	8,08	100%
T600	600	500	8,82	100%
T650	650	454,5	10,29	100%

Table 7.4: Create Unknown asset benchmark test results (not using Caliper)

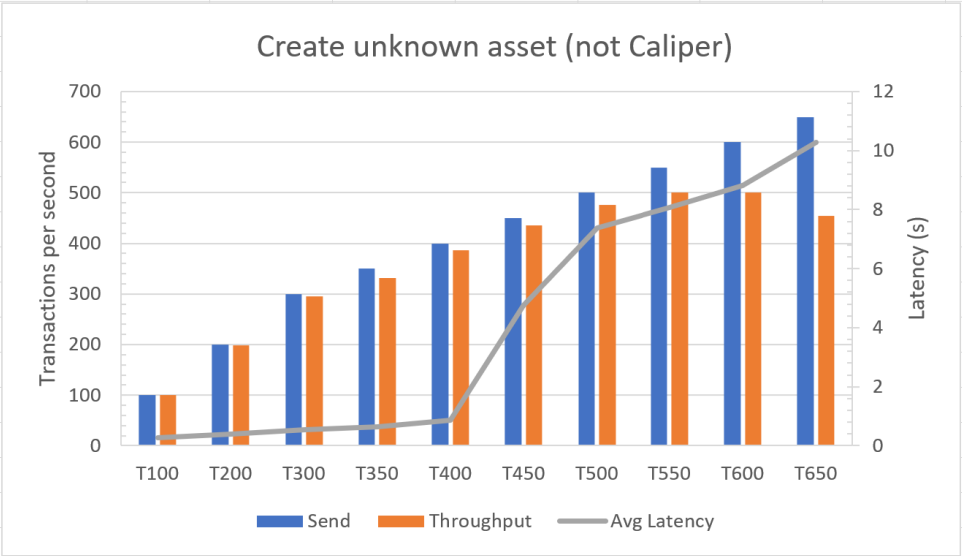

Figure 7.7: Bar chart of Create Unknown asset benchmark results

The first set of tests done was to Create Unknown Asset function of AMSC (Note: the Personal Data Analyser (PDA) call was disabled to do the testing). Starting by throughput analysis, we can see that the Blockchain system is able to closely keep up with the sent rate until the 500 TX/s. Regarding reliability, we had no execution failure of any kind even on the top rates. In terms of latency, we have a much higher latency overall when comparing to Caliper benchmarks. We will consider the target latency to be less than one second, which is attainable with sent rates of up to 400TX/s.

Asset Management (not Caliper)							
Function	**Test**	**Snd(TX/S)**	**T.put(TX/s)**	**St.Dev**	**Lat(s)**	**St.Dev**	**Reliab**
CreateConsent	T300	300	294,12	0,00	0,40	0,03	100%
	T350	350	326,16	5,07	0,55	0,15	100%
	T400	400	389,75	7,25	0,53	0,01	100%
CreateScore	T300	300	291,32	3,96	0,69	0,17	100%
	T350	350	337,41	10,48	1,92	0,95	100%
	T400	400	379,87	6,72	2,58	0,30	100%
UpdtConsent False	T300	300	291,32	3,96	0,45	0,02	100%
	T350	350	329,75	5,07	0,50	0,03	100%
	T400	400	389,75	7,25	0,73	0,15	100%

Table 7.5: Asset Management benchmark test results (not using Caliper)

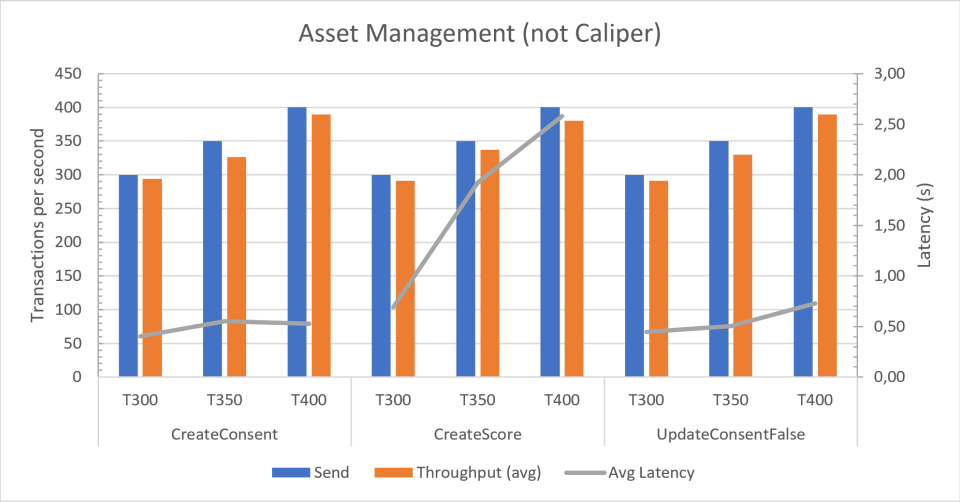

Figure 7.8: Bar chart of Asset Management benchmark results

Based on both Caliper and non-Caliper Benchmarks, we concluded that at best we would be able to manage a rate of 400 TX/s, so for our last test, we limited the sent rate to the [300;400] interval, and tested three different functions of AMSC: consent creation, user/score creation, and update consent to false (that beside the consent asset update, also deletes the user/score assets regarding that user ID).

From the analysis, we can see that both consent creation and update have very similar results to each other and to Unknown Asset Creation. However, in terms of latency, the user/score asset surpasses the 1-second latency threshold with rates higher than 300 TX/s due to its increased complexity.

7.1.4 Final system definition and configuration

Based on the tests performed on the previous subsection (7.1.3), we were able to decide on how to configure the system to assure the performance and reliability non-functional requirements defined on subsubsection 3.3.1: *100% of reliability, less than 2 seconds of latency, and at least a steady rate of 100 TX/s of throughput.*

On the Gateway side, we limited the number of simultaneous message handling to 300; while on the Blockchain itself, we limited it to 400 transactions per second. With this configuration, and since all of the BlockPrivGDPR transaction will be coming from the Gateway module, we will be limiting the volume of transactions to a rate that is compliant with the performance and reliability requirements in all of the performed tests, however, we keep a gap of 100TX/s in the Blockchain configuration to allow (eventual) transactions outside of BlockPrivGDPR to occur.

7.2 System Evaluation

In this section, we do a walkthrough of the system to show how the system responds, and how the interactions take place.

7.2.1 User Actions and responses

This subsection presents the user action forms, along with the possible responses to the users. All of the forms are based on templates freely provided by w3docs [66].

When a user wishes to joint the network, he/she must provide informed consent. To do so, the consent form should be used, where the user can read the consent and JCA information, and provide his/her personal data. Figure 7.9 shows the filled form, and the full draft documents regarding consent and JCA can be found in appendix B and appendix C respectively.

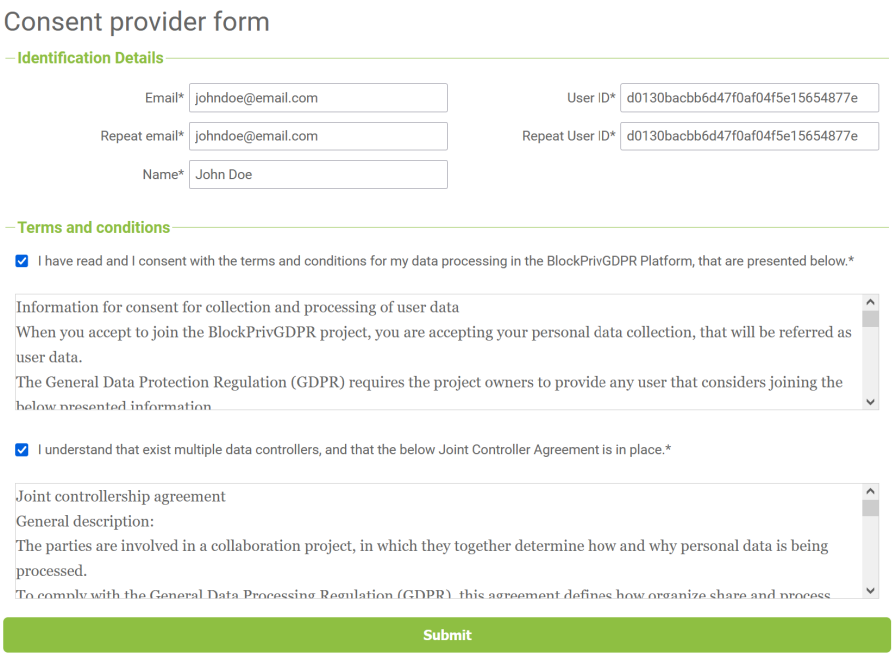

Figure 7.9: Filled user consent form

If a user needs to exercise the right to rectify his/her data, desire to object or restrict the processing of his/her data based on a justification, or wants to proceed with consent withdrawal and data erasure, he/she can use the data update form, presented on figure 7.10.

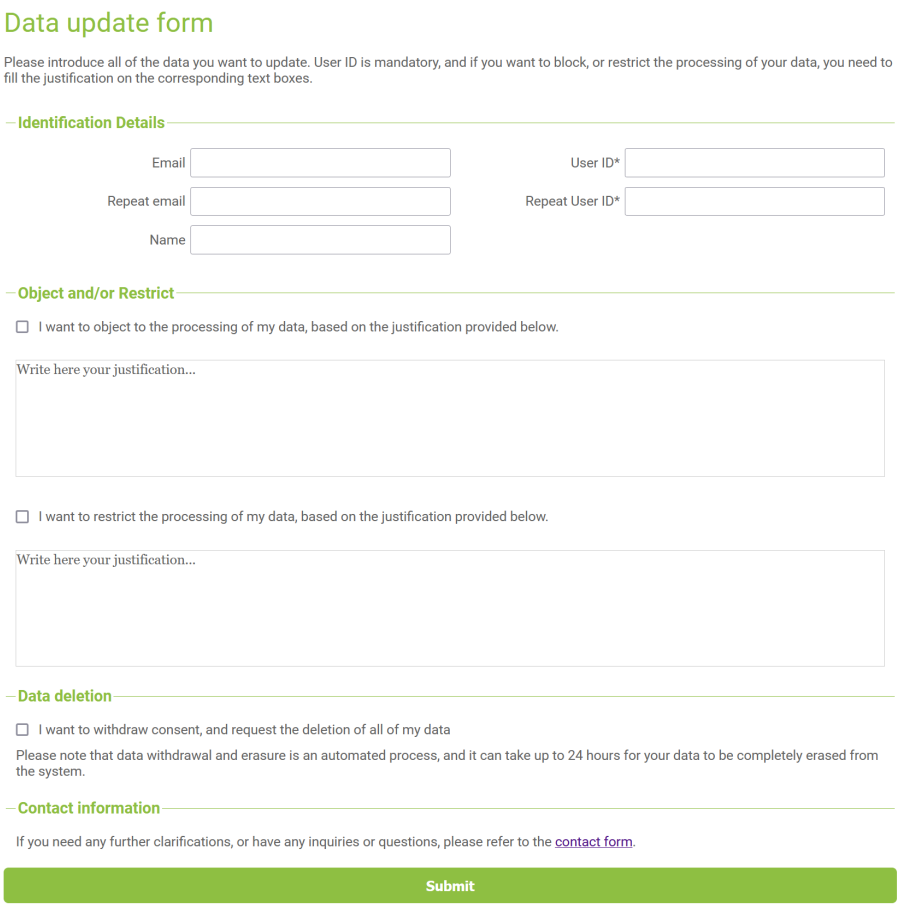

Figure 7.10: Data update form

When the user submits the form information, a message with his/her request is sent to the system, and once a response is generated, the page will be redirected to a new one that will show the result of the operation. If the operation succeeds, a success message is shown, however if it fails, a failure message that includes the failure reason is shown. Figure 7.11 depicts both cases.

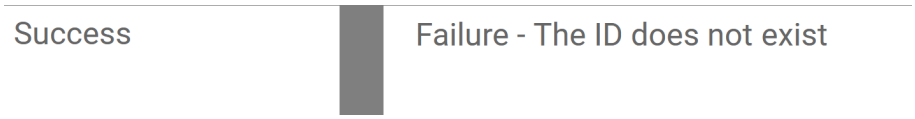

Figure 7.11: Form request result

The right of access and data portability can also be exercised in an instant automated way, via the right of access form (figure 7.12).

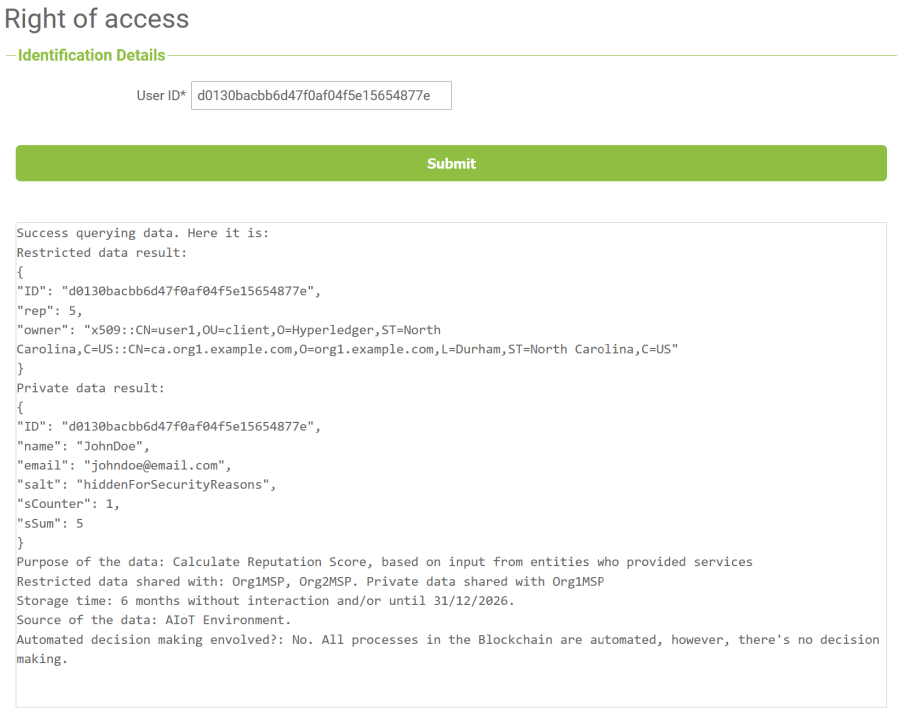

Figure 7.12: Right of Access form

A contact form is also available for the users (figure 7.13). When a user submits the filled information, an email is automatically sent to the email contacts of the organizations that are part of the Joint Controllership Agreement via Smart Contract (JCA-SC).

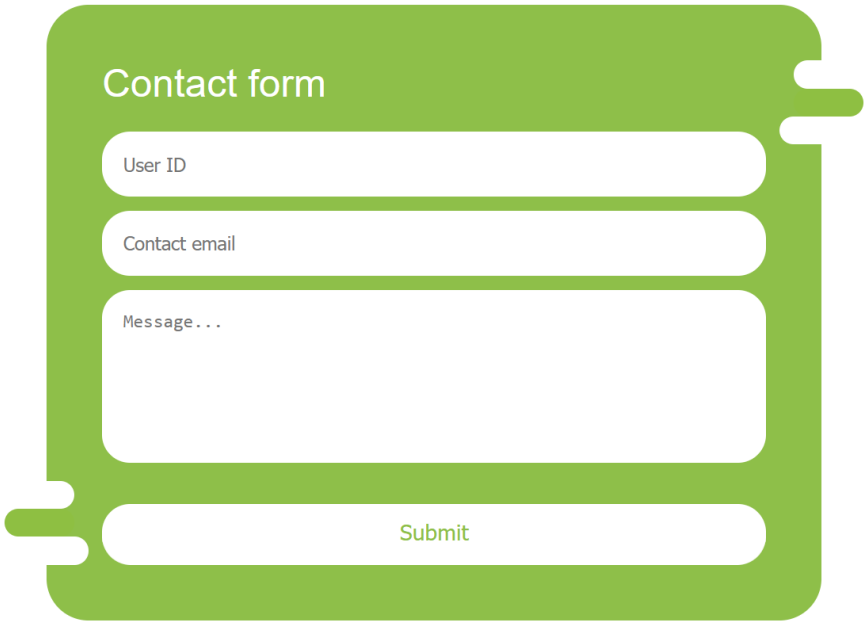

Figure 7.13: Contact form

7.2.2 ARCADIAN-IoT interactions and responses

As previously shown and explained, the ARCADIAN-IoT components are required to have a communication module via RabbitMQ to interact with the remaining modules. So, no further specific development had to be done, beside what was already covered to support this interaction. A full specification of each message (both requests and responses) syntax was done, and is included in this work as appendix D.

For the Reputation System, which is the component that will interact the most with our system, the mainly used functions will be reputation score update, and retrieval, which will occur via AMSC, and have the flow described in figure 6.7. Some failure messages will be described in subsection 7.2.3.

For the remaining components, as there are no specified interactions, we designed the *Unknown* asset, which includes PDA analysis (demonstrated in subsection 7.2.4). The flow of this asset handling is specified in figure 6.8.

7.2.3 System showcase

To summarize the system, and highlight how the system responds to the most common messages exchanged, we decided to create a comprehensive user story that demonstrates an entire life cycle of a user/score asset.

The goal is to showcase the different interactions to the reader in a way that is logical and sequential, and not too extensive. An important note is that even though not all of the requests/responses are covered in this subsection, all of them were covered, and the results are presented in Appendix F.

The Gateway outputs will be shown, as it covers both the message that lead to the action, and the response of the chaincode invoke that will be returned.

User registration

When a user wants to request any service provided by the companies involved in the project, he/she must provide consent and personal information via the Consent Provider Form (figure 7.9). When the form is submitted, two messages are sent to our Gateway: one to create the user consent asset, and another to create the user/score asset, as shown in figure 7.14.

```
2023/08/23 00:33:08 Received a message: createConsent-id:d0130bacbb6d47f0af04f5e15654877e,consentFlag:true
2023/08/23 00:33:08 Handle as consent creation
2023/08/23 00:33:08 components size: 2
2023/08/23 00:33:08 Size is OK
Submit Transaction: CreateAssetConsent, creates new asset consent with ID, consentFlag arguments
*** Transaction committed successfully
2023/08/23 00:33:10 Response sent...
Final time (ms) :  2024
2023/08/23 00:33:19 Received a message: score-id:d0130bacbb6d47f0af04f5e15654877e,score:0,name:JohnDoe,email:johndoe@email.com
2023/08/23 00:33:19 Handle as score
2023/08/23 00:33:19 Size is OK
Submit Transaction: CreateAsset, creates new asset consent with ID, Rep, Name, Email, Salt arguments
*** Transaction committed successfully
2023/08/23 00:33:19 Response sent...
Final time (ms) :  2032
```

Figure 7.14: Gateway: creation of user consent and asset

At this point, the user reputation score will be zero (figure 7.15), but all of the user data will be recorded in the controlled databases, and can be used by the authorized entity, to whom the user wants services from.

Service Request

When the user requests the desired service, before accepting, the company responsible for the service provision can query the Blockchain to see if the user has provided consent, and the value of his/her reputation:

```
2023/08/23 00:53:02 Received a message: read-id:d0130bacbb6d47f0af04f5e15654877e
2023/08/23 00:53:02 Handle as asset read
2023/08/23 00:53:02 Size is OK
Evaluate Transaction: ReadAsset, function returns asset attributes
*** Result:{
 "ID": "d0130bacbb6d47f0af04f5e15654877e",
 "rep": 0,
 "owner": "x509::CN=user1,OU=client,O=Hyperledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O=org1.example.com,L=Durham,ST=North Carolina,C=US"
 }
2023/08/23 00:53:02 Response sent...
Final time (ms) :  4
```

Figure 7.15: Gateway: restricted data read

After the service is concluded, the company shall provide a reputation score to the user, regarding the service experience. To do so, an update message will be sent to the Gateway:

```
2023/08/23 00:55:03 Received a message: updateScore-id:d0130bacbb6d47f0af04f5e15654877e,score:10
2023/08/23 00:55:03 Handle as updateScore
2023/08/23 00:55:03 Size is OK
Submit Transaction: UpdateAsset, updates an existing asset consent with ID, rep arguments
*** Transaction committed successfully
2023/08/23 00:55:05 Response sent...
Final time (ms) :  2025
```

Figure 7.16: Gateway: user reputation score update

User blocks processing and updates personal information

If a user looses access to the provided email contact, he/she can temporarily block the data processing while the situation is not fixed, using the Data Update Form (figure 7.10).

```
2023/08/23 00:56:36 Received a message: updateConsent-id:d0130bacbb6d47f0af04f5e15654877e,consentFlag:true,blocked:true
2023/08/23 00:56:36 Handle as consent update
2023/08/23 00:56:36 Size is OK
Submit Transaction: UpdateAssetConsent, updates an existing asset consent with ID, consentFlag arguments
*** Transaction committed successfully
2023/08/23 00:56:38 Response sent...
Final time (ms) :  2018
```

Figure 7.17: Gateway: consent blockage

If during this time, an intervening entity wants to process the user data, it won't be able to, as depicted in figure 7.18

```
2023/08/23 00:57:32 Received a message: updateScore-id:d0130bacbb6d47f0af04f5e15654877e,score:5
2023/08/23 00:57:32 Handle as updateScore
2023/08/23 00:57:32 Size is OK
Submit Transaction: UpdateAsset, updates an existing asset consent with ID, rep arguments
failed to submit transaction: %!w(*client.EndorseError=&{0xc00042e3d8})Endorse error with gRPC status Aborted: rpc error: code = Aborted desc = failed to endorse
transaction, see attached details for more info
Error from endpoint: peer0.org1.example.com:7051, mspId: Org1MSP, message: chaincode response 500, Processing is blocked (RoO, RoR)
2023/08/23 00:57:32 Response sent...
Final time (ms) :  5
```

Figure 7.18: Gateway: user reputation score update failure

If the user creates a new email address, and wishes to update it, and unblock the processing, using the form, an update message is sent to BlockPrivGDPR.

```
2023/08/23 00:59:00 Received a message: updateConsent-id:d0130bacbb6d47f0af04f5e15654877e,consentFlag:true,blocked:false
2023/08/23 00:59:00 Handle as consent update
2023/08/23 00:59:00 Size is OK
Submit Transaction: UpdateAssetConsent, updates an existing asset consent with ID, consentFlag arguments
*** Transaction committed successfully
2023/08/23 00:59:02 Response sent...
Final time (ms) :  2019
```

Figure 7.19: Gateway: user consent unblock

User leaves the system

After some services requests, if the user decides that he/she no longer wants to participate in the ARCADIAN-IoT project, he/she can request their data to store it for eventual future use.

```
2023/08/23 01:00:39 Received a message: rightOfAccess-id:d0130bacbb6d47f0af04f5e15654877e
2023/08/23 01:00:39 Handle as right of access
2023/08/23 01:00:39 Size is OK
Evaluate Transactions: ReadAssetPrivateDetails + ReadAsset, function returns asset attributes
Success querying data. Here it is:
Restricted data result:%s
 {
 "ID": "d0130bacbb6d47f0af04f5e15654877e",
 "rep": 10,
 "owner": "x509::CN=user1,OU=client,O=Hyperledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O=org1.example.com,L=Durham,ST=North Carolina,C=US"
 }
Private data result:%s
 {
 "ID": "d0130bacbb6d47f0af04f5e15654877e",
 "name": "JohnDoe",
 "email": "johndoe@email.com",
 "salt": "hiddenForSecurityReasons",
 "sCounter": 1,
 "sSum": 10
 }
Purpose of the data: Calculate Reputation Score, based on input from entities who provided services
Restricted data shared with: Org1MSP, Org2MSP. Private data shared with Org1MSP
Storage time: 6 months without interaction and/or until 31/12/2026.
Source of the data: AIoT Environment.
Automated decision making envolved?: No. All processes in the Blockchain are automated, however, there's no decision making.
2023/08/23 01:00:39 Response sent...
Final time (ms) :  9
```

Figure 7.20: Gateway: right of access request and response

Then, he/she uses the Data Update Form to request consent withdrawal and data deletion.

```
2023/08/23 01:05:19 Received a message: updateConsent-id:d0130bacbb6d47f0af04f5e15654877e,consentFlag:false,blocked:false
2023/08/23 01:05:19 Handle as consent update
2023/08/23 01:05:19 Size is OK
Submit Transaction: UpdateAssetConsent, updates an existing asset consent with ID, consentFlag arguments
*** Transaction committed successfully
2023/08/23 01:05:21 Response sent...
Final time (ms) :  2016
```

Figure 7.21: Gateway: user consent revoke

Once the request is fully processed, no further personal records or reputation information are kept in BlockPrivGDPR.

```
2023/08/23 01:05:34 Received a message: read-id:d0130bacbb6d47f0af04f5e15654877e
2023/08/23 01:05:34 Handle as asset read
2023/08/23 01:05:34 Size is OK
Evaluate Transaction: ReadAsset, function returns asset attributes
*** BAD Result:
*** The asset either doesn't exist, or you don't have permission to access it.
2023/08/23 01:05:34 Response sent...
Final time (ms) :  3
2023/08/23 01:05:44 Received a message: readPvt-id:d0130bacbb6d47f0af04f5e15654877e
2023/08/23 01:05:44 Handle as asset private read
2023/08/23 01:05:44 Size is OK
Evaluate Transaction: ReadAssetPrivateDetails, function returns asset attributes
*** BAD Result:
*** The asset either doesn't exist, or you don't have permission to access it.
2023/08/23 01:05:44 Response sent...
Final time (ms) :  4
```

Figure 7.22: Gateway: user data query failure

7.2.4 PDA interaction

When an *Unknown* asset creation is requested by some authorized entity, a request to the PDA is made to check for possible Personal Identifiable Information (PII).

As previously mentioned, and as specified in Appendix D, the message to request the creation of this kind of assets, contains 6 fields: the id, two entries for restricted storage, and three entries for private storage. Even though, the separation of restricted and private values should be previously done, the Gateway sends all values to the PDA for analysis, and the asset creation is based on the output.

Figure 7.23 depicts the Gateway output when a message to create an unknown asset arrives: the message is sent with asset ID, one allegedly restricted values, and two allegedly private values (when 0 is sent, it means the absence of value, as these values do not have a mandatory number of parameters). The Gateway writes the three parameters in file "Test_3.txt" and sends a message to the PDA. PDA then answers with two arrays of evaluation [3], and based on the answer, the parameters are considered restricted or private. If the response corresponding to the parameter is *'O'*, it means that the value is most like not a PII, and therefore treated as restricted, if it is different than *'O'* (e.g. *'per'* - person name, *'geo'* - geographical place), it means that it is most likely a PII, and therefore handled as private data.

[3]The PDA uses two different types of word processing, but based on our observations and tests, the first response array was the one that provided the most accurate answers, and therefore, only the first response array is considered.

```
2023/09/02 19:16:51 Received a message: createOther-id:u229,unk1:0,unk2:Roberts,pvt1:Engineer,pvt2
:Romania,pvt3:0
2023/09/02 19:16:51 Handle as other/unknown
2023/09/02 19:16:51 unkVal: u229
2023/09/02 19:16:51 unkVal: 0
2023/09/02 19:16:51 unkVal: Roberts
2023/09/02 19:16:51 unkVal: Engineer
2023/09/02 19:16:51 unkVal: Romania
2023/09/02 19:16:51 unkVal: 0
2023/09/02 19:16:51  [x] Sent to PDA Test_3.txt
2023/09/02 19:16:51 Received a message: (['per', 'per', 'geo'], ['org', 'org', 'eve'])
2023/09/02 19:16:51 1 - pvt
2023/09/02 19:16:51 2 - pvt
2023/09/02 19:16:51 3 - pvt
Submit Transaction: CreateUnknownAsset, creates new unknown asset (max 2 restricted, 3 private)
*** Transaction committed successfully
2023/09/02 19:16:54 Response sent...
Final time (ms) :  2062
```

Figure 7.23: Gateway: PDA interaction

As all of the parameters were considered as private data, the asset was created accordingly. Figure 7.24 shows how the asset was stored, with no stored data besides the id stored in the restricted database, and the remaining parameters stored in the private database.

```
2023/09/02 19:23:19 Received a message: readUnk-id:u229
2023/09/02 19:23:19 Handle as unknown asset read
2023/09/02 19:23:19 Size is OK
Evaluate Transaction: ReadUnknownAsset, function returns asset attributes
*** Result:{
 "ID": "u229",
 "unk1": "0",
 "unk2": "0",
 "owner": "x509::CN=user1,OU=client,O=Hyperledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O
=org1.example.com,L=Durham,ST=North Carolina,C=US"
 }
2023/09/02 19:23:19 Response sent...
Final time (ms) :  3
2023/09/02 19:23:30 Received a message: readUnkPvt-id:u229
2023/09/02 19:23:30 Handle as unknown asset private read
2023/09/02 19:23:30 Size is OK
Evaluate Transaction: ReadUnknownAssetPrivateDetails, function returns asset attributes
*** Result:{
 "ID": "u229",
 "unkPvt1": "Roberts",
 "unkPvt2": "Engineer",
 "unkPvt3": "Romania",
 "salt": "hiddenForSecurityReasons",
 }
2023/09/02 19:23:30 Response sent...
Final time (ms) :  3
```

Figure 7.24: Gateway: Unknown asset querying

Lastly, figure 7.25 portrays the PDA output when receives a request from the Gateway. In this case, the strings sent to evaluation were *'nothing'* and *'test'*, and the response to both was *'O'*, as those were not considered to be PII, meaning that those were stored on the restricted database.

```
 [x] Received 'Test_3.txt'
answering...
Input path is: %s Test_3.txt
File extension: %s .txt
Starting loading PII.
PII loading finished.
2023-09-02 19:37:16,405 - INFO - TXT file processing started.
DOC CONTENTS
nothing
test

2023-09-02 19:37:16,405 - DEBUG - Predicting entity...
2023-09-02 19:37:16,406 - DEBUG - Predicting entity...
ANALYSIS FINISHED!!!!
_[x] Sent bytearray(b"([\'O\', \'O\'], [\'O\', \'O\'])")
```

Figure 7.25: PDA: String analysis

7.2.5 Non-functional evaluation summary

Table 7.6 summarizes how we were able to fulfil the Non-functional requirements specified in subsection 3.3.1.

Sub-requirement	Achievement summary
Usability	
RMQ messages specification	All messages are specified in appendix D
Interaction forms	All forms designed with ease of use in mind, as shown in subsection 7.2.1
Security	
TLS encryption	TLS v1.3 in-transit encrypted; TLS v1.2 inside Blockchain
PII control	Private storage of known PII; PDA use for unknown PII detection
Backup erasure	Erasure Script
Availability	
99,9% time	Not Evaluated
Reliability	
100% on Gateway	Benhckmarks-based configuration
100% on Blockchain	Benhckmarks-based configuration
Recoverability	
Orderer recoverability	Erasure Script to assure backup deletion from nodes, so only orderer can be relied upon
Performance	
Latency	Benhckmarks-based assurance
no limit on PDA	Not Evaluated
Throughput	Benchmarks-based assurance: 300TX/s + 400TX/s assurance
Capacity	
1000 TX/day	Not Evaluated

Table 7.6: Non-functional requirements fulfilment table

7.2.6 GDPR compliance evaluation and discussion

In order to visually summarize the compliance regarding each one of the GDPR rights and principles, we compiled the tables 7.7 and 7.8 using the following colour scheme (further considerations are presented below):

- Green - We believe that our system is fully compliant
- Blue - There are some doubt as to whether full compliance is achieved
- Yellow - The system is only partially compliant
- Red - The system is not compliant

GDPR Rights	Mechanisms
The right to be informed	-Informed consent required to join the network -Right of access form -Contact form to request further information -JCA information available when providing consent -Interactions logging SC can provide additional information on data processing
The right of access	-Right of access form -Contact form to request further information
The right to rectification	-Data update form to request personal data correction
The right to erasure	-Data update form to request data deletion
The right to restrict processing	-Data update form to restrict data processing
The right to data portability	-All data can be requested in a machine-readable format via Right of access form
The right to object	-Data update form to restrict data processing
Rights in relation to automated decision-making and profiling	-All automated mechanisms (Smart contracts) can be stopped, updated, and erased

Table 7.7: GDPR Rights

GDPR Principles	Mechanisms
Lawfulness, fairness and transparency	-Consent Smart Contract -Interactions logging Smart Contract -Direct channel of communication (Contact form) -Consent and JCA information available for the user
Purpose limitation	-Consent Smart Contract -JCA Smart Contract
Data minimization	-Only required data is collected
Accuracy	-Node update whenever a new block is generated -Backup node data erasure -AMSC assures automated (and near instant) data update
Storage limitation	-Access controlled databases -Backup node data erasure -Data erased after 6 months of unuse
Integrity and confidentiality	-Access controlled databases -TLS v1.2 for secure communication inside Blockchain -TLS v1.3 for RabbitMQ communications -Blockchain-integrated mechanisms -Single use salts -Backup node data erasure
Accountability	-Interactions logging Smart Contract -JCA Smart Contract

Table 7.8: GDPR Principles

By analysing the tables above, it can be seen that we consider our system fully compliant with the GDPR **rights**, and with almost all of the GDPR **principles**.

We identified the principle **Storage limitation** as one where we have some concerns regarding full compliance, due to possible retention of old data, which can happen for two main reasons:

1. Synchronization delays - where a given node retains old data, because it is not receiving updates from the ledger (e.g., looses internet connection).
2. Backup outdated data - as previously discussed, the nodes keep a log of the received transactions that in some cases might contain outdated data (e.g. a user updates his/her data shortly after he/she provides it). While the erasure script is not executed, the outdated data will remain stored, even though it will not be considered for data querying/processing.

Even though this can happen, we put in place technical measures to help prevent this (erasure script[4]), and ones that allow us to define the maximum retention periods (24 hours for the node update, and 6 hours for backup data erasure). We inform the users of these limitations, and state the maximum retention times on the user consent information (Appendix B), and we also state that each organization is responsible to ensure the correct script execution in the JCA (Appendix C).

As per the **Integrity and Confidentiality** principle, we identified an issue that makes us believe that BlockPrivGDPR is not fully compliant on this principle. This is related with the backup log files: even though we implemented the technical and legal measures synthesized above, the fact that in the backup data there are some data stored in plain-text leads us to believe that it might contribute to a possible data breach, where some information might be illegally accessed.

To try to mitigate this risk, we included a clause on the JCA where it's stated that each organization is responsible to assure that no illegal access occurs to these log files. However, the file access is not handled via BlockPrivGDPR, but rather on the Operating System (OS) level which might not be fully controllable nor auditable.

An important consideration on this matter is that the files are only accessible by default by users with administrator privileges on the machines, which automatically restricts the number of users who can access these files. So, with very restrict guidelines, and very explicit clauses on the JCA we could be set for full compliance on this principle.

As a final evaluation, we filled in the checklist provided by GDPR.EU website [3], that we included as Appendix E. In short, we were able to tick all the boxes, with the exception of the ones that are directed to the organizations running the system, authority notifications and data transference to entities outside the European Union (EU).

Overall, we consider that we were very close to achieving full GDPR compliance. Further suggestions for system improvement and completion are provided in section 8.1.

[4]The Erasure Script is fully explained in subsection 6.2.6, where some further concerns are described

Chapter 8

Conclusion

When we started this work, we had the main challenge of uncovering all of the conflicts between Blockchain systems and the General Data Protection Regulation (GDPR), so we could design and implement a platform that relies on the Blockchain technology and is GDPR-compliant.

To achieve this goal, we started by analysing the literature, where we did not find a proposal that would meet our requirements, but rather many pieces that we reasoned we could merge, in order to create the desired system.

After analysing, comparing, and consider multiple permissioned Blockchain implementations, we decided to use Hyperledger Fabric, as it was the most promising one in terms of privacy. A full documentation study, and a lot of practical testing was performed, to understand in detail how this Blockchain, and its privacy-preserving mechanisms work.

Based on our studies, analyses, and practical tests, and starting from our initial proposition of merging multiple partial-solutions found in the literature, we were able to define the final mechanisms to implement, and how they would interact, in order to create our system: BlockPrivGDPR.

BlockPrivGDPR is a system that is composed by three main modules: the Blockchain, the Gateway, and the User interaction forms.

In the Blockchain we have two smart contracts deployed with multiple functions that allow for full data control and management, while directly ensuring Lawfulness, Transparency, and Accountability. We also used Hyperledger Fabric access controlled databases, that allowed us to use multiple databases with different types of access restrictions.

The Gateway module is composed by two different parts: the receiver and the invoker. The receiver is responsible to handle all communications in and out of the system, being it from the User forms, the Reputation System, or other component of ARCADIAN-IoT network, relying on TLS v1.3 protocol to keep the messages confidential and secure. The invoker is responsible to invoke the deployed chaincodes, and also to perform the non-deterministic operations beforehand (e.g., salt generation).

Finally, the User interactions form provide an easy and direct way for the users to interact with the system, and to obtain all of the information they need or want regarding their data. These forms allow the user to exercise all of their GDPR-related rights.

Beside the mentioned components, some other were integrated into the system with specific privacy related functions.

After a performance, and functional analysis, that helped us fine tune the system configurations, we performed a GDPR compliance evaluation, where we found that our system is nearly fully compliant with the GDPR, as we raised some confidentiality-related concerns, due to the possible temporary store of some personal data in plain-text: even though this storage is on access controlled directories (where only system administrators can access), there is a script implemented that erases any of these files every 6 hours, the user is alerted for this in the consent note, and we included a special clause for the handling of these files in the JCA, as this storage happens on the OS level, we could not implement a technical measure that would fully assure that no access violation would occur.

As per our final results, we believe that we accomplished our goal, and produced not only a Blockchain-based system that covers all of the GDPR rights and principles, but also provide insight on the various mechanisms that can be used to achieve compliance in a system that is by definition non-compliant with GDPR.

8.1 Future Work

As per future work, we have some specific suggestions, and some more broad.

Starting with the PDA integration, we believe it would be highly beneficial to improve the PDA interface in order to send the arguments for analysis directly instead of via file.

The second suggestion is related with the weekly control script. It would be beneficial if the BlockToLive values were controlled by the Blockchain, or if different methods were used to improve the method of dynamic data erasure.

The third suggestion is related with the confidentiality concern we described. For our system to be fully compliant, a definitive solution needs to found for the log files mentioned of the Ledger copy of the peers. It is also necessary to do a more complex testing of the impacts of the Erasure Script in the Blockchain system, specially in bigger networks than the one we used, as stated in subsection 6.2.6

The final suggestion would be to use our system as a base to deploy one that would rely on a fully external database, and fully implement the off-chain storage that was discussed in the beginning of this work

8.2 Final Note

As a final note, it's important to highlight that all of the evaluations are technical, and based on our interpretation of GDPR and use cases. Anyone wanting to implement BlockPrivGDPR should be aware of the exceptions applied to some special categories of personal data, and seek legal advice before using it.

The presented explanations, interpretations, and conclusions are not legal advice in any way.

References

[1] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)*, 51(4):1–35, 2018.

[2] Proton AG. What is gdpr, the eu's new data protection law?, May 2022. URL `https://gdpr.eu/what-is-gdpr/`.

[3] Proton AG. Gdpr checklist for data controllers, 2023. URL `https://gdpr.eu/checklist/`.

[4] Ifteher Alom, Md Sadek Ferdous, and Mohammad Jabed Morshed Chowdhury. Blockmeter: An application agnostic performance measurement framework for private blockchain platforms, 2022.

[5] ARCADIAN-IoT Consortium. Arcadian-iot - framework, Aug 2022. URL `https://www.arcadian-iot.eu/arcadian-iot-framework/`.

[6] ARCADIAN-IoT Consortium. Arcadian-iot - homepage, Aug 2022. URL `https://www.arcadian-iot.eu/`.

[7] Sujit Biswas, Kashif Sharif, Fan Li, Zohaib Latif, Salil S. Kanhere, and Saraju P. Mohanty. Interoperability and synchronization management of blockchain-based decentralized e-health systems. *IEEE Transactions on Engineering Management*, 67(4):1363–1376, 2020. doi: 10.1109/TEM.2020.2989779.

[8] David Both. How i use cron in linux, 2020. URL `https://opensource.com/article/17/11/how-use-cron-linux`.

[9] Hyperldger Caliper, 2023. URL `https://hyperledger.github.io/caliper/`.

[10] Michael del Castillo. Blockchain 50: Billion dollar babies, Dec 2022. URL `https://www.forbes.com/sites/michaeldelcastillo/2019/04/16/blockchain-50-billion-dollar-babies/?sh=4fe7bd5857cc`.

[11] European Commission. Horizon 2020, 2014. URL `https://research-and-innovation.ec.europa.eu/funding/funding-opportunities/funding-programmes-and-open-calls/horizon-2020_en`.

[12] Maryam Davari and Elisa Bertino. Access control model extensions to support data privacy protection based on gdpr. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 4017–4024. IEEE, 2019.

[13] Agencia Española de Protección de Datos. Introduction to the hash function as personal data pseudonymisation tecnhnique, October 2019. URL https://edps.europa.eu/sites/default/files/publication/19-10-30_aepd-edps_paper_hash_final_en.pdf.

[14] EAI. 19th eai international conference on security and privacy in communication networks, 2023. URL https://securecomm.eai-conferences.org/2023/.

[15] EPRS and M Finck. *Blockchain and the general data protection regulation : can distributed ledgers be squared with European data protection law?* Publications Office of the European Union, 2019. doi: doi/10.2861/535.

[16] Hyperledger Fabric. Github - hyperledger fabric-samples - fabcar - go, 2021. URL https://github.com/hyperledger/fabric-samples/tree/d1b3253cc56e1438667261e2c4931794af49cb64/chaincode/fabcar/go.

[17] Simon Farshid, Andreas Reitz, and Peter Roßbach. Design of a forgetting blockchain: A possible way to accomplish gdpr compatibility. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.

[18] Michèle Finck. Smart contracts as a form of solely automated processing under the gdpr. *Max Planck Institute for Innovation and Competition Research Paper*, 19-01, 2019.

[19] Directorate-General for Communication of the European Commission. Data protection, 2022. URL https://commission.europa.eu/law/law-topic/data-protection_en.

[20] Hyperledger Foundation. Hyperledger blockchain performance metrics white paper, 2018. URL https://www.hyperledger.org/learn/publications/blockchain-performance-metrics.

[21] The Apache Software Foundation. Apache license, version 2.0, january 2004. URL https://www.apache.org/licenses/LICENSE-2.0.

[22] Grafana. Grafana oss: Query, visualize, alerting observability platform, 2023. URL https://grafana.com/oss/grafana/?plcmt=footer.

[23] Sejin Han and Sooyong Park. A gap between blockchain and general data protection regulation: A systematic review. *IEEE Access*, 10:103888–103905, 2022. doi: 10.1109/ACCESS.2022.3210110.

[24] Parikshit Hooda. Raft consensus algorithm, Aug 2022. URL https://www.geeksforgeeks.org/raft-consensus-algorithm/.

[25] Hyperledger. Endorsement policies, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.2/endorsement-policies.html.

[26] Hyperledger. Secured asset transfer in fabric, 2022. URL https://hyperledger-fabric.readthedocs.io/en/latest/secured_asset_transfer/secured_private_asset_transfer_tutorial.html.

[27] Hyperledger. What's new in hyperledger fabric v2.x, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatsnew.html.

[28] hyperledger. What's New, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.5/whatsnew.html.

[29] Hyperledger. Certificates management guide, 2022. URL https://hyperledger-fabric.readthedocs.io/en/latest/certs_management.html.

[30] Hyperledger. Glossary, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.2/glossary.html.

[31] Hyperledger. Introduction, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatis.html.

[32] Hyperledger. Using private data in fabric, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.5/private_data_tutorial.html#pd-purge.

[33] Hyperledger. Private Data, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.5/private-data/private-data.html.

[34] Hyperledger. Transaction flow, 2022. URL https://hyperledger-fabric.readthedocs.io/en/release-2.2/txflow.html.

[35] Hyperledger. Fabric-samples/asset-transfer-events at main, 2022. URL https://github.com/hyperledger/fabric-samples/tree/main/asset-transfer-events.

[36] Hyperledger. Fabric-samples/asset-transfer-private-data, 2022. URL https://github.com/hyperledger/fabric-samples/tree/main/asset-transfer-private-data.

[37] Hyperledger. Fabric chaincode lifecycle, 2023. URL https://hyperledger-fabric.readthedocs.io/en/latest/chaincode_lifecycle.html.

[38] Hyperledger. Using the fabric test network with certificate authorities, 2023. URL https://hyperledger-fabric.readthedocs.io/en/release-2.5/test_network.html#bring-up-the-network-with-certificate-authorities.

[39] Hyperledger. Metrics reference, 2023. URL https://hyperledger-fabric.readthedocs.io/en/latest/metrics_reference.html.

[40] Hyperledger. The ordering service, 2023. URL https://hyperledger-fabric.readthedocs.io/en/release-2.5/orderer/ordering_service.html.

[41] Hyperledger. Performance considerations, 2023. URL https://hyperledger-fabric.readthedocs.io/en/latest/performance.html.

[42] Hyperledger. Prometheus-grafana hyperledger fabric repository, 2023. URL https://github.com/hyperledger/fabric-samples/tree/main/test-network/prometheus-grafana.

[43] IBM. What are smart contracts on blockchain?, 2022. URL https://www.ibm.com/topics/smart-contracts.

[44] Jibe Company. Personal data analyser (pda), 2021. URL https://bitbucket.org/jibecompany/poseidon-pda/src/master/.

[45] Esmeralda Kadena and Peter Holicza. Security issues in the blockchain(ed) world. In *2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 000211–000216, 2018. doi: 10.1109/CINTI.2018.8928212.

[46] Marie-Jeanne Lagarde. Security assessment of authentication and authorization mechanisms in ethereum, quorum, hyperledger fabric and corda. *Ecole Polytechnique Federale De lausannne*, pages 1–99, 2019.

[47] Driss EL MAJDOUBI, Hanan EL BAKKALI, and Souad SADKI. Towards smart blockchain-based system for privacy and security in a smart city environment. In *2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech)*, pages 1–7, 2020. doi: 10.1109/CloudTech49835.2020.9365905.

[48] Anisha Mirchandani. The gdpr-blockchain paradox: exempting permissioned blockchains from the gdpr. *Fordham Intell. Prop. Media & Ent. LJ*, 29: 1201, 2018.

[49] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.

[50] Article 29 Data Protection Working Party. Opinion 05/2014 on anonymisation techniques, 2014. URL https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf.

[51] Julien Polge, Jérémy Robert, and Yves Le Traon. Permissioned blockchain frameworks in the industry: A comparison. *Ict Express*, 7(2):229–233, 2021.

[52] Hauke Precht and Jorge Marx Gómez. Towards gdpr enforcing blockchain systems. In *International Conference on Wirtschaftsinformatik*, pages 440–446. Springer, 2021.

[53] Hauke Precht and Jorge Marx Gómez. Redactable blockchain-leveraging chameleon hash functions for a gdpr compliant blockchain. In *Konferenzband zum Scientific Track der Blockchain Autumn School 2020*, pages 66–70. Hochschule Mittweida, 2020.

[54] Prometheus. Overview: Prometheus, 2023. URL https://prometheus.io/docs/introduction/overview/.

[55] Daniel Ruby. 72+ blockchain statistics 2022 - how many people own bitcoin?, Nov 2022. URL https://www.demandsage.com/blockchain-statistics/.

[56] Steffen Schwalm. *DIN SPEC 4997 Privacy by Blockchain Design: A standardised model for processing personal data using blockchain technology*. DIN SPEC, 04 2020.

[57] Applied Testing Shane McCarron, Technology, Creative Commons Ben Adida, Sidewinder Labs Mark Birbeck, Kellogg Associates Gregg Kellogg, W3C Ivan Herman, and CWI Steven Pemberton. Html+rdfa 1.1 - second edition, 2015. URL https://www.w3.org/TR/html-rdfa/.

[58] Paulo Silva, Carolina Gonçalves, Nuno Antunes, Marilia Curado, and Bogdan Walek. Privacy risk assessment and privacy-preserving data monitoring. *Expert Systems with Applications*, 200:116867, 2022. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2022.116867. URL https://www.sciencedirect.com/science/article/pii/S0957417422003153.

[59] Betsy Stringam and John Gerdes. Service gap in hotel website load performance. *International Hospitality Review*, 33(1):16–29, 2019.

[60] Isti Surjandari, Harman Yusuf, Enrico Laoh, and Rayi Maulida. Designing a permissioned blockchain network for the halal industry using hyperledger fabric with multiple channels and the raft consensus mechanism. *Journal of Big Data*, 8(1), 2021. doi: 10.1186/s40537-020-00405-7.

[61] The European Parliament and the Council of European Union. Regulation (eu) 2016/679 of the european parliament and of the council, 2016. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679.

[62] Naohiro Togashi and Vitaly Klyuev. Concurrency in go and java: Performance analysis. In *2014 4th IEEE International Conference on Information Science and Technology*, pages 213–216, 2014. doi: 10.1109/ICIST.2014.6920368.

[63] Sadok Ben Toumia, Christian Berger, and Hans P. Reiser. Evaluating blockchain application requirements and their satisfaction in hyperledger fabric, 2021.

[64] Pete Townshend. The 5 biggest gdpr fines and why they were issued, Oct 2022. URL https://smartframe.io/blog/the-5-biggest-gdpr-fines-and-why-they-were-issued/.

[65] VMware. Rabbitmq official website, 2023. URL https://www.rabbitmq.com/.

[66] w3docs, 2023. URL https://www.w3docs.com/learn-html/html-form-templates.html.

[67] Baocheng Wang, Jiawei Sun, Yunhua He, Dandan Pang, and Ningxiao Lu. Large-scale election based on blockchain. *Procedia Computer Science*, 129:234–237, 2018. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2018.03.063. URL https://www.sciencedirect.com/science/article/pii/S1877050918302874. 2017 International Conference on identification, information and knowledge in the Internet of Things.

[68] Dylan Yaga, Peter Mell, Nik Roby, and Karen Scarfone. Blockchain technology overview. *arXiv preprint arXiv:1906.11078*, 2019. URL `https://doi.org/10.6028/nist.ir.8202`.

[69] Lingfang Zeng, Zhan Shi, Shengjie Xu, and Dan Feng. Safevanish: An improved data self-destruction for protecting data privacy. In *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, pages 521–528, 2010. doi: 10.1109/CloudCom.2010.21.

Appendices

Appendix A

Article submitted to EAI SecureComm 2023

BlockPrivGDPR: Privacy and GDPR Compliance in Blockchain

Daniel Vasconcelos[1] and Bruno Sousa[1]

University of Coimbra, CISUC, DEI
uc2017261340@student.uc.pt, bmsousa@dei.uc.pt

Abstract. Blockchain has unique characteristics for integrity, immutability and traceability in decentralized networks, potentiating its adoption. Despite guaranteeing trust without relying on a centralized authority, it has privacy issues when considering regulations like GDPR. Blockchain partially supports some GDPR principles, like lawfulness, fairness and integrity. Nonetheless, there are conflicts with other principles like transparency and data minimization. BlockPrivGDPR is a mechanism aiming to solve the privacy issues related to data storage in Blockchains. BlockPrivGDPR contributes with a comprehensive analysis of GDPR compliance and with a Proof of Concept to validate technical solutions towards full GDPR compliance. The findings of BlockPrivGPDR are that GDPR compliance in Blockchain requires several actors performing distinct configuration tasks.

Keywords: Blockchain · GDPR · Privacy Preserving · Off-Chain · Proof of Integrity · Hyperledger Fabric

1 Introduction

General Data Protection Regulation (GDPR) introduced in 2018, significantly altered how companies handle personal data. Technologies like Blockchain offer unique benefits but also present challenges in full compliance with the GDPR, particularly when it is related to storing and processing personal information. Blockchain is a digital distributed ledger, that does not rely on a centralized authority to guarantee trust but rather uses cryptographic mechanisms to guarantee information integrity and trustworthiness [1].

GDPR is a European Union (EU) regulation that aims to protect and preserve the data privacy of the EU citizens' personal data, by providing specific guidelines on what is required for organizations to keep and process users' personal data [2]. Several studies [3, 4] consider the issues of GPDR compliance in Blockchains. Despite the relevance, such works mainly focus on a limited set of rights or principles of GDPR.

We propose BlockPrivGDPR as a mechanism to solve the privacy issues related to data storage in Blockchains, through 1) extensive analysis of GDPR rights and principles and their implications in the data storage and processing in Blockchains; 2) Proof of Concept that establishes the foundations to validate

technical solutions towards full GDPR compliance.BlockPrivGDPR contributes to advancing the state of the art in privacy compliance, by identifying the conflicts between GDPR and the Blockchain technology, seeking solutions to provide full compliance in terms of data processing and storing. Using a specific use case, BlockPrivGDPR builds a proof of concept covering all the principles and rights of GDPR. BlockPrivGDPR proposes technical and organizational mechanisms for data processing in a GDPR-compliant fashion.

The remainder of this article is organized as follows: Section 2 introduces the related background, Section 3 analyzes the compatibility issues of Blockchain in terms of GDPR compliance. Section 4 overviews technical aspects and related work, Section 5 documents the high level overview of the proposed mechanism and Section 6 details the practical aspects. Section 7 concludes the paper.

2 Background

This section provides a background of Blockchain and GDPR.

2.1 Blockchain

Blockchain is a decentralized ledger, with no central authority, that is used to record and store transactions between users, nodes/peers [1].

The transactions between users are recorded in chained blocks: each block contains the hash value of the previous one, which guarantees chaining and chronological order, integrity, and immutability to all the recorded transactions. Data Authenticity is assured using public-private sets of keys (digital signatures).

A block is divided in two parts: a) the **block header** containing block information like block number, hash of the previous block, block size, b) and the **block data** including the data of transactions. To add blocks to the chain, a consensus mechanism is used, where the participants agree and validate the committed transactions. As there is no centralized party to keep the records, the "full nodes", besides participating in the network, also store a copy of the ledger and the chain.

Blockchains can be **public** or **private** when considering the access criteria. Regarding the participation perspective, Blockchains can be **permissionless**, where anyone can see and interact with it, or **permissioned**, where only the users with explicit access can see and/or interact with it. In the permissioned Blockchain, users have to be identified before joining, and restrictions and access policies can be configured to individual users, or to organizations. To the best of our knowledge, and based on the conducted analysis we found that GDPR compliance would only be possible in permissioned Blockchains.

Several implementations of permissioned Blockchains exist, such as Hyperledger Fabric, Ethereum Geth, Quorum, MultiChain, and R3 Corda. They differ in the support of support for Smart Contract, as well as for security and privacy support [5, 6]. BlockPrivGDPR relies on Hyperledger Fabric due its superior support for privacy, security and for providing documentation that facilitates

the implementation of privacy preserving mechanisms [5, 6]. Further details are provided in section 6.

2.2 GDPR

This section provides an overview of key definitions in GDPR, transcribed from *gdpr.eu website* [2], mapped to a use case. We consider a hypothetical use case where John Doe requests a vigilance from a drone towards its home return, after a dinner with friends. The drones' service is provided by the DGAVigi company.

Personal data — "Personal data is any information that relates to an individual who can be directly or indirectly identified. Names and email addresses are obviously personal data. Location information, ethnicity, gender, biometric data, religious beliefs, web cookies, and political opinions can also be personal data." The Personal data is the data that is used by DGAVigi regarding John Doe, for instance, its email address, home location, and biometric data.

Data processing — "Any action performed on data, whether automated or manual." DGAvigi performs some processing of data, either to authenticate John Doe or when the drone identifies John Doe in a street.

Data subject — "The person whose data is processed." The data subject corresponds to the user John Doe.

Data controller — "The person who decides why and how personal data will be processed." This person corresponds to the Data Protection Officer of DGAVigi.

Data processor — "A third party that processes personal data on behalf of a data controller." The DGAVigi company uses AI services, from a third-party enterprise to enable users identification by drones using photos or live images.

The **Data protection principles** in GDPR are the following [2]: (1) Lawfulness, fairness and transparency, (2) Purpose limitation, (3) Data minimization, (4) Accuracy, (5) Storage limitation, (6) Integrity and confidentiality, and (7) Accountability.

Every user has the following **rights**, regarding their data [2]:

1. *The right to be informed*, John Doe needs to be informed regarding which processing is done regarding its data.
2. *The right of access*, John Doe has the right to access its data
3. *The right to rectification*, John Doe has the right to modify and correct data that is stored in the platform(s) of DGAVigi.
4. The right to erasure, John Doe can erase its profile and associated data in the DGAVigi platform(s).
5. *The right to restrict processing*, John Doe can exercise this right if he believes the data used is not accurate.
6. *The right of data portability*, John Doe may request the data stored in DGAVigi and provide it to another enterprise.
7. *The right to object*, John Doe has the freedom to not use any new service that might be provided by DGAVigi.
8. *Rights in relation to automated decision making and profiling.* John Doe needs to provide explicit consent regarding the automated processing of his data.

3 BlockPrivGDPR: GDPR vs BLOCKCHAIN

The first step in the specification of the BlockPrivGDPR includes a detailed analysis on the GDPR compliance in Blockchain technologies, so the conflicts can be identified. For such, we analyzed the study of Dr Michéle Finck at the request of the Directorate-General for Parliamentary Research Services (European Parliament) [3] and relevant standards, namely the DIN SPEC 4997 [7]. The analysis is reported by grouping GDPR principles and/or rights according to the mechanisms available in the Blockchain.

3.1 Lawfulness, fairness and transparency, and the rights to be informed/of access, and of data portability

By specifying how the data is used by each entity, and collecting free and informed consent from the data subjects, we are essentially able to cover lawfulness and fairness topics. Entity identification, however, might not be possible in public permissionless Blockchains, as it is in permissioned ones [3].

In terms of transparency, it can be difficult to comply if there is not a specific and simple channel of communication between the data subjects and the data controller(s), so this must be ensured [3]. Another problem can be the assurance of which entities actually processed the data, and on which basis (purpose) the processing had been done [7].

It is fundamental that the data subjects can access all the required information to be fully informed on how their data is used and processed, by whom and for what purposes. In a decentralized network, it can be hard to guarantee the availability of the controller(s) at any given time. This also applies to the data request in a structured and readable format.

3.2 Purpose limitation, and the rights to restrict processing and to object

The data provided by a data subject has a defined purpose, that has to be respected. Given the nature of the Blockchain, when the data is stored on the Blockchain itself, it will be used to calculate the block hash value, which is then used on the next block, and therefore on the next block, and so on; meaning, that there is a "perpetual processing" of the data [3]. In addition, a clear and transparent purpose statement for each intervening entity should be provided, along with access control mechanisms that enforce it [3].

Due to the design of the decentralized network, native data immutability, and the number of processors and controllers, there might be some governance obstacles to providing the rights to object and to restrict processing [3].

3.3 Data minimization and storage limitation

Blockchain by definition only allows append operations [1], meaning that it will continue to grow over time, keeping all the introduced data. This leads to the

question: *When does the data become obsolete?* [3]. Even though anonymization techniques may help to comply with these principles, they do not provide all the necessary guarantees [3].

The fact that multiple copies of the ledger exist in the network also raises concerns regarding these principles [3].

3.4 Accuracy, Integrity, and the rights of erasure and rectification

Immutability and integrity of the data are assured by design, so, accuracy is guaranteed, however, the rights of erasure and rectification cannot be achieved without additional mechanisms [3].

3.5 Confidentiality

Confidentiality is one of the biggest challenges, as personal data must only be accessible to authorized entities.

Encryption might help achieving some level of confidentiality, however, it has to be assured that the algorithms used will remain secure throughout the whole lifetime of the platform [3]. In permissioned Blockchains, as the entities are identified when joining, they are legally bounded to compliance, however, in public permissionless Blockchains, it's not possible to control who joins the network, and therefore, who has a copy of the ledger, so, ill-intentioned entities might try to keep an illegal copy of the data for future exploitation.

Even when access control is enforced, there can be publicly accessible "unsafe" components of the data by default (e.g., plain hash values), which must be taken into consideration when configuring and designing Blockchain-based systems.

3.6 Accountability

The data controller is the identity responsible to specify and implement the technical and organizational measures to ensure that the processing of the data is compliant with GDPR, and it should be demonstrable [8].

In the eventuality of existing more than one entity defining the terms and purposes of the processing, there is not a data controller, but rather joint controllers [8]. By defining and signing a Joint Controllership Agreement (JCA), the responsibilities, purposes, measures and means for processing are agreed upon, and must be reflected in the design and mechanisms of the system [3].

3.7 Rights in relation to automated decision making and profiling

"Smart contracts are simple programs stored on a Blockchain that run when predetermined conditions are met. They typically are used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss", as described by IBM [9].

Smart contracts can be fully automated, and not require any human intervention, so it's important to identify the automated processes and analyze how they process data, so safeguards can be introduced, and explicit consent asked if necessary.

4 Feasible Solutions and related work

After understanding the conflicts between Blockchain and GDPR, it is relevant to identify proposed solutions in the literature. To avoid repetitions, we grouped the principles/rights, and we present the solutions based on the use case previously introduced.

4.1 Lawfulness, fairness, transparency and the rights to be informed and of access

When John Doe first joins the network, he must be informed of the joining conditions and data usage in an explicit, detailed and accurate way, so he can provide informed consent.

Smart Contracts can be ideal for this management, mainly for two reasons: (1) The use of machine (binary) language makes it easier to understand since it removes ambiguity and subjectivity; (2) All Smart Contract interactions are logged in the Blockchain, creating records that assure integrity, immutability and non-repudiation [10].

Being smart contracts created by humans, the possibility of errors or software bugs exists [10]. This should be taken into consideration, and there should be ways to update the software and re-ask subjects for consent if needed.

If John only wants to consent to some of the data usages, the use of Resource Description Framework in Attributes (RDFa) [11] to manage and store the different types of consent can be employed [12]. By doing so, controllers and processors can easily access the consent to lawfully handle data.

For fairness to be assured, John has to be able to reach (one of) the controller(s) at any given time. For that to happen, there has to be a direct channel of communication, which is only possible on permissioned Blockchains, where the identities are properly identified, unlike public permissionless ones [3, 10].

For the network to be transparent, John must be able to know which entities processed the data and on which basis [7]. A technical solution would be the creation of a logging Smart Contract [13], which would allow not only John to consult it, but also the controller(s), who could use it to audit and prove GDPR compliance.

With the stated mechanisms, we can also assure that the basis to comply with the right to be informed, and the right of access are put in place, as long as John is able to request and receive his information in a machine-readable format (which can be automated via Smart Contract).

4.2 Purpose limitation, and the rights to restrict processing and to object

When John gives consent, he must be able to withdraw or change it at any time. The use of RDFa mechanisms could be a way to restrict (or narrow) the consent, and if expanded, could also be used for the users to exercise the right to restrict processing, and to object.

4.3 Storage limitation, accuracy and the rights to rectification and erasure

After John provides his personal data, he might change his ideas and request data deletion, or simply require its update. Given the append-only nature of Blockchain, this is not possible. To mitigate this issue, additional mechanisms to edit the information are required [3].

A technical solution would be to keep the personal data off-chain and to store only an integrity-proof on-chain. By doing so, the immutability of the records is bypassed, since off-chain records can be edited or deleted, and a new block would be created on the chain with the corresponding ("new") integrity proof. Despite this, we still have to guarantee the untraceability and anonymity of the Blockchain recorded values, as explained in section 4.4.

It's important to note some limitations of the off-chain mechanisms [10]: it can increase costs and complexity of the system; the multiple copies of the data can become desynchronized; and it can reduce the security levels.

Another solution could be the use of chameleon hash functions [14]. Cryptographic ("regular") hash functions are by definition collision-resistant, which means that it's computationally infeasible to compute two different inputs that produce the same output. So, when the input changes, the output will also change. Chameleon hashes, however, allow changes in the input value, while the output remains the same: when the hash value is first calculated, a secret key called *trapdoor key* is also generated. This key can be used later to produce the same output value with different inputs [14].

In the model presented by H. Precht [14], the blocks contain two security measures regarding the chameleon hashes:

- The equivalent to the block header contains the hash value and a checksum (the Blockchain implementation used by the authors is simplified):
  - This checksum will change if the information inside the block changes, even though the hash value will remain the same, so it's easily noticeable and controllable when a block changes.
- Only the block owner has the trapdoor key:
  - This guarantees that only the owner of the block is able to change it.

As the decision to alter the block is not of the data subject but rather of an external entity (e.g., the owner of the data), the mentioned mechanism is not viable. The trapdoor key should be only in the possession of a single trusted entity (e.g., the data controller), and all changes would be controlled by them [10].

There are also some discussions regarding deletion consensus mechanisms, where there would be voting performed by all nodes, and if the majority agreed, the block would be deleted; however, due to the chained nature of the Blockchain, this could disrupt it [15].

4.4 Integrity and confidentiality

Integrity is guaranteed by Blockchain mechanisms, yet confidentiality is not. When using the mentioned off-chain techniques to allow compliance with other principles and rights, confidentiality must be kept in mind, as stated by Article 29 Data Protection Working Party [16]:

- The sole use of hash functions is not enough to ensure the confidentiality of a record.
- The use of salted-hash functions can hinder input derivation, but it may still be feasible, when considering short salt values, and/or not secret.
- The use of a keyed-hash function with a stored secret key may introduce enough entropy and difficulty for an attack to be impractical.

In the case of a Blockchain it may be infeasible to keep a secret key between all the involved parties, however, it might be possible to assure enough entropy (and therefore impracticality of attacks) by using sufficiently big single-use salt values that would have to be stored along with the personal data, for integrity validation, but otherwise kept secret [17].

Access Control and/or encryption mechanisms are often chosen to ensure confidentiality [10]. In terms of access control, there are two main techniques: X.509 certificates with the required information on the extensions to identify a subject [18], and Attribute-Based Encryption Keys (ABE Keys), which encrypt only specific items of data.

Regarding encryption to achieve anonymization, there are two proposed encryption techniques: Zero Knowledge Proof (ZKP), and Homomorphic Encryption (HE). ZKP are algorithms that provide boolean outputs that are the result of a verification of a given input, so a statement can be proved without the disclosure of any sensitive data. The usage of these algorithms might be enough to anonymize data [7]. HE allows some computations, like addition and multiplication, to be performed on encrypted information that produces the same result as if the plain text information was used. By using this kind of mechanism, the data can be stored and used in an encrypted format, and only the owner of the encryption key can later decrypt it [19]. The usage of HE has been proposed on Blockchain systems where only simple operations are required on the private data, like electronic voting systems [20].

4.5 Data minimization

If the personal data is kept off-chain, it's easier to comply with this principle, as the original data can be deleted, and it is computationally infeasible to guess the

data values based on the Blockchain-stored integrity proof, assuming single-use salts are used [17].

Along with the erasure requests, there can be time limits regarding the storage of the data that can be defined when the data object is created, and the data is automatically deleted when the 'expiry date' is reached, which is possible by default in some Blockchain implementations, like the Hyperledger Fabric [21].

Another complementary mechanism is Blockchain pruning: where old blocks' content would be erased and only kept the block header [22]. In case an off-chain storage method is used, the off-chain data concerning the target of the pruning would also have to be deleted [10].

Regarding the synchronization of the data between the parties involved, it occurs by default every time a new block is committed to the Blockchain [3], so even if a node stays temporarily offline, it should synchronize its data when the next block is committed. If a node goes offline for a longer period of time than accepted, it can be implemented a data self-destruction mechanism like the one proposed by Zeng et Al to cloud computing [23], that would guarantee that non-compliant data is deleted.

The default mechanisms of Blockchain do not guarantee the synchronization of the off-chain databases, but there have been proposed mechanisms to assure that this is done in an acceptable period of time, through the use of a Digital Data Converter (DDC) [24].

4.6 Accountability

The distributed nature of Blockchain leads to difficulties in terms of accountability. Due to the fact that there are many parties involved, and usually more than one data controller, that may not be available at all times,

H. Precht [4] suggests the implementation of a Joint Controllership Agreement Smart Contract (JCA-SC) that can be used to:

- Guarantee GDPR compliance - The data must be processed according to the JCA.
- Guarantee that a valid JCA is in place - A call to the JCA-SC would be made at every method of the processing contract.
- Guarantee that at least one of the controllers is available - For the JCA to be valid, there has to be at least one controller node online.

This integration impacts the performance of the contracts since calls are less efficient. However, the implementation is considered to be simple and straightforward in implementations that support Smart Contract calls inside other Smart Contracts like Hyperledger Fabric, and there are many benefits in terms of guaranteeing compliance [4].

4.7 The right of data portability

If John wants to request his personal information, as long as the data and, at least, a controller is available, there are no further conflicts that arise from the

Blockchain. The JCA-SC [4] could be enough to guarantee the availability of a data controller, that could send the data in a machine-readable format.

4.8 Rights in relation to automated decision making and profiling

M Finck [25] analyzes how Smart Contracts interact with the rights in relation to automated decision-making and profiling (article 22 of GDPR). M Finck concludes that when considering the original meaning of Smart Contracts [9], those are not compatible with article 22. Notwithstanding, there can be ways to create GDPR-compatible Smart Contracts, namely with the introduction of *"arbitration mechanisms"* [25].

These mechanisms would allow humans to pause, alter, or even remove the software that composes the smart contracts, and furthermore, the contracts could have methods to provide information on data usage to the subjects. These conditions would make the Smart Contracts compatible with article 22 [25].

4.9 Exemption of GDPR for Blockchain

The study of A Mirchandani [26] analyzes the interaction between Blockchains and GDPR. The author suggests that the easier way to ensure compliance would be to exempt permissioned Blockchain from the GDPR. For this to happen, one of the solutions would be for the users to provide explicit consent not to exercise some of their rights, namely the right to be forgotten, the right to data portability, and the right to rectification.

Another solution for this exemption would be [26]: to alter the language of GDPR to include an exception for permissioned Blockchains; to clarify the definition of data erasure under the GDPR; and to classify hashed personal data as anonymized.

5 BlockPrivGDPR: High-Level Specification

This section provides a high-level overview of the BlockPrivGDPR considering the solutions found, that together are able to cover all principles and rights of GDPR, enabling the possibility of full compliance.

Figure 1 states the selected mechanisms (middle column) and demonstrates how all rights and principles are accounted for. Figure 2 depicts how the components can be deployed and interact with each other. The blue rectangle represents the Blockchain, which contains the Smart Contracts, the nodes, and the ledger. Outside the Blockchain, we have the off-chain databases, the users, the controllers, and the mechanisms to control the off-chain databases.

6 BlockPrivGDPR: Practical Aspects

This section provides details regarding the BlockPrivGDPR Proof of Concept towards a practical implementation of a Blockchain fully compliant with GDPR.

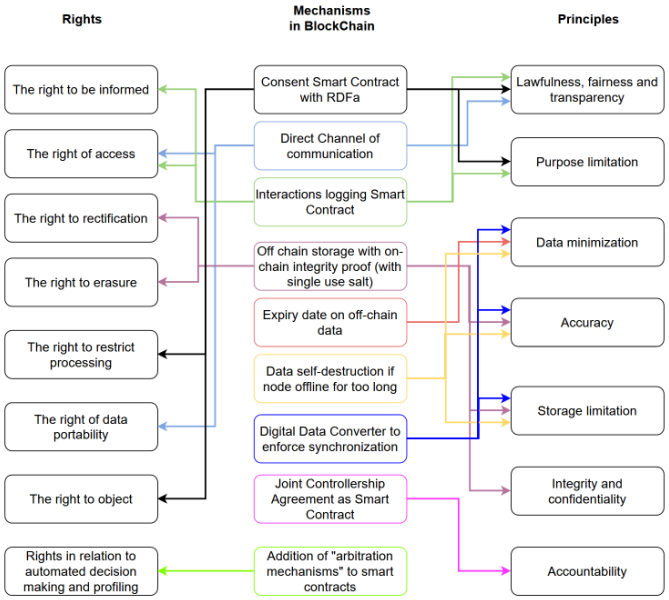

Fig. 1: Chosen mechanisms connection to rights and principles

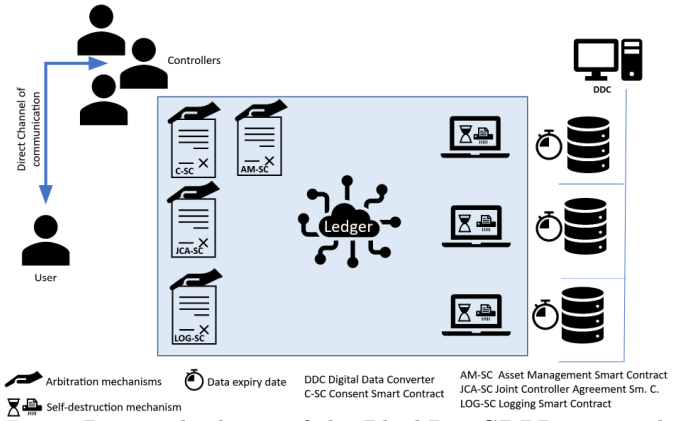

Fig. 2: Practical scheme of the BlockPrivGDPR proposal

6.1 Introduction

To avoid the overhead and complexity of using an external database, BlockPrivGDPR uses the private database mechanisms supported by Hyperledger Fabric. Besides the main/public state database that can be accessed by any peer/node, there is the possibility to have private state databases, that enforce access control through the use of X.509 certificates [27]. These databases can be configured to be completely private, or with restricted access.

The Proof of Concept relies on HyperLedger Fabric version 2.5, due to the higher GDPR compliance levels. The versions prior to 2.0 do not contain some of the privacy mechanisms used, and version 2.5 includes a critical improvement

on the private data deletion mechanism, as it includes the erasure of the private data history from both state and peer's database [28].

The Proof of Concept considers the vigilance scenario introduced in section 2, where DGAvigi service and users like John Doe are associated with reputation values. In this regard, the implemented mechanisms include:

- Simplified version of Asset Management Smart Contract, that includes two different Asset Types. Before generating UserData Objects, the Consent flag of the Consent Object will be checked: if it's set to false, or does not exist, the creation of the asset will be denied. The asset types include:
  - Consent {UserID:<string>, Consent:<bool>} – to simulate user consent. UserData objects can only be created if the Consent flag is set to True.
  - UserData {UserID:<string>, Reputation:<numeric>, Owner:<string>, Name:<string>, Email:<string>, Salt:<string>} – actual user data.
- Logspout [29] is used to simulate the Logging Smart Contract, by monitoring all actions regarding the Blockchain.
- Expiry dates are defined on the UserData objects, using the BlockToLive property [21].
- Default LevelDB is used, due to security concerns.

Figure 3 demonstrates how consent can be given, and how user objects can be created and used by Ledger(s).

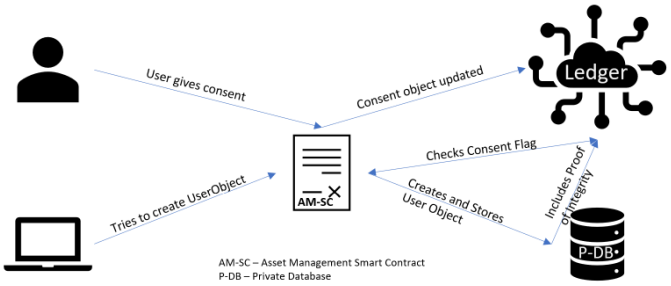

Fig. 3: Creation flow of user consent and object

Figure 4 shows how the system behaves if the user revokes their consent.
In terms of objects' access policies, there are three different possibilities:

- Public - everyone can query the asset/value stored on the public state database. The parameters are sent as input and therefore are stored on the chain.
- Restricted - the organizations with permissions can query the value, as it is stored on the shared private database. The parameters are sent as transient input and therefore are not stored on the chain [27].
- Private - only the organization that created the asset can query the value, as it is stored on the private database of the organization. The parameters are sent as transient input and therefore are not stored on the chain.

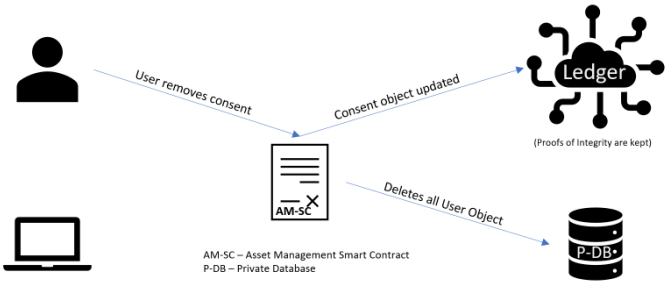

Fig. 4: Consent revoking flow

Two types of objects were defined with the following policies:

- Consent Object - with the policies: *User ID* - public, *Consent* - public
- User Object - with the policies: *User ID* - restricted, *Reputation* - restricted, *Owner* - restricted, *Name* - private, *Email* - private, *Salt* - private.

6.2 Blockchain deployment and requirements

For Blockchain deployment, we used the provided scripts by Hyperledger Fabric official tutorials, launched along with Certification Authorities. Two different organizations (*org1*, *org2*) are created, and the peers (with corresponding X.509 certificates) are launched as Docker containers. The default LevelDB database is used, for security reasons.

After the successful deployment of the Blockchain, we launch Logspout, to monitor all docker containers, and therefore log all interactions that occur in the network, independently of its source and destination.

6.3 Smart Contract definition and considerations

The first step to properly define the assets/objects to be created is to define/configure the collections, which defines the security/access policies of the assets to be created. We defined three different configurations:

- **AssetCollection**: Accessible by both organizations and locked to their members. *BlockToLive*: 100000.
- **Org1MSPPrivateCollection**: Only accessible by Org1, *BlockToLive*:3.
- **Org2MSPPrivateCollection**: Only accessible by Org2, *BlockToLive*:3.

As for the Smart Contract itself, we defined three different objects/assets:

- **AssetConsent** – the object saving the user consent (ID, ConsentFlag).
- **Asset** – the object with the user restricted properties (ID, rep, owner).
- **AssetPrivateDetails** – the object containing the private properties of the user (ID, Name, Email).

In terms of functions, we present the most relevant ones, and a brief summary of their intended purpose:

- **CreateAssetConsent** – allows the creation of the consent object. It receives as input parameters the ID and the value for the consent flag.
- **ReadAssetConsent** – queries a consent object using the ID as input.
- **UpdateAssetConsent** – allows the update of the consent object. It receives the ID and the value for the consent flag as input parameters. If the user changes their consent to false, it will call the function DeleteAsset to delete the user object, if existing.
- **CreateAsset** – allows the creation of a user object. The parameters are sent using the transient flag (so the input values are not recorded on the chain). It calls the ReadAssetConsent function to check if consent has been provided by the user. The field "Owner" of the asset, is populated with the information regarding the X.509 certificate of the peer that invokes it.
- **DeleteAsset** – allows the deletion of a user object. Like the previous function, it has no input parameters, as the ID is sent using the transient flag.
- **ReadAsset** – queries the restricted data of the user object. Receives the ID as an input parameter.
- **ReadAssetPrivateDetails** – queries the private data of the user object. Receives the ID and the name of the private collection the object is stored in as input parameters.

6.4 Demonstration

In this section, we provide screenshots of BlockPrivGDPR implementation demonstrating how the system behaves in aspects related to privacy. First, we created

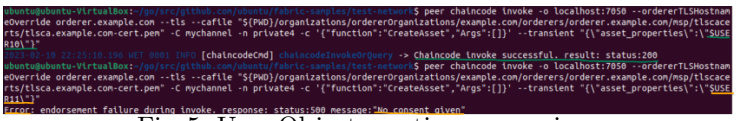

Fig. 5: User Object creation comparison

two Consent Objects, one for user ID 10, with consent set to true, and another with user ID 11, with consent set to false. The following steps included the creation of a User Object for each of the users. As expected and shown in Figure 5, we were able to create the User Object for user ID 10, but not for user ID 11 (status result = 500, indicating an error).

Figure 6 shows the difference in the creation of Consent Objects and User Objects, in terms of transactions, and what's recorded on the chain blocks. On the left side of the figure, we can see that all data regarding the inputs and the created asset are recorded in plain text. On the right side, we can see that there are no input arguments stored (only the function name), and only the hashes of the objects are stored.

Figures 7 and 8 depict the result of querying the restricted and private properties of the User Object, respectively. In figure 7, the transaction creator belongs to *org1*, which is the same organization that created the asset and can fetch the

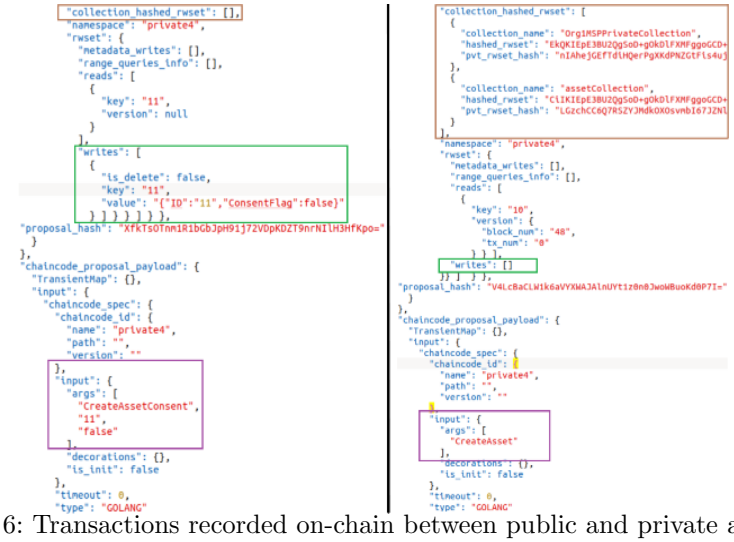

Fig. 6: Transactions recorded on-chain between public and private assets

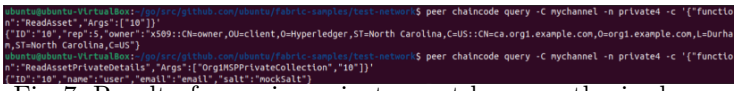

Fig. 7: Result of querying private asset by an authorized user

respective private values. While in figure 8 the querying user, is from *org2* and doesn't have permission to fetch the private values.

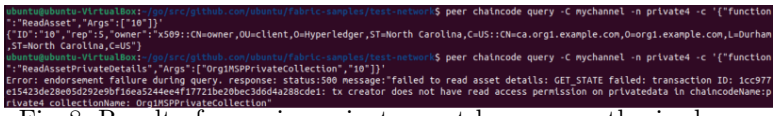

Fig. 8: Result of querying private asset by an unauthorized user

When a user revokes the consent, the User Object is automatically erased. On a request to read an asset by its ID, not information is returned back.

Figure 9 demonstrates the BlockToLive feature, which automatically erases the User Object when the defined number (3) of new blocks are created and there's no interaction with the object, it is also automatically erased.

Finally, figure 10 portrays the logging information regarding the last query done, as shown in figure 9 to User Object, and it's given as an example of what is recorded in the logging functionality.

With this Proof of Concept, we were able to show some of the privacy-preserving mechanisms included in Hyperledger Fabric, that can help with system compliance, namely the transient mechanism, the private/restricted asset management, and the BlockToLive functionality. We have also shown the operation of Smart Contracts, and how it can be used to enforce and automate consent checking, and deletion of no longer lawful assets.

Fig. 9: Demonstration of BlockToLive feature

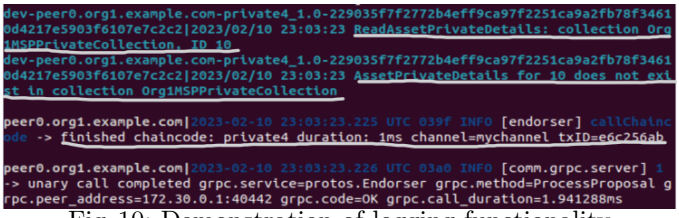

Fig. 10: Demonstration of logging functionality

7 Conclusion

BlockPrivGDPR is a mechanism aiming to address the privacy issues of Blockchains and their compliance with GDPR. To this aim, BlockPrivGDPR provides an in-depth analysis of Blockchain and GDPR, identifying the issues and the solutions towards GDPR compliance. Within the identification of feasible approaches and mechanisms that can work together to cover all rights and principles of the GDPR. BlockPrivGDPR contributes to the deployment of a Blockchain-based system towards full GDPR compliance.

The validation of the BlockPrivGDPR mechanisms relies on a detailed design leveraging the state of the art and on a Proof of Concept implementation that demonstrates a set of functionalities compliant with GDPR, through the design of diverse types of Smart Contracts in scenarios involving diverse organizations and data controllers.

ACKNOWLEDGMENTS

This work was supported by the project ARCADIAN-IoT project, which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101020259.

References

1. Yaga, Dylan and Mell, Peter and Roby, Nik and Scarfone, Karen, "Blockchain technology overview," *arXiv preprint arXiv:1906.11078*, 2019.

2. P. AG, "What is GDPR, the EU's new Data Protection Law?" May 2022. [Online]. Available: https://gdpr.eu/what-is-gdpr/
3. European Parliament and Directorate-General for Parliamentary Research Services and Finck, M, *Blockchain and the general data protection regulation : can distributed ledgers be squared with European data protection law?* Publications Office of the European Union, 2019.
4. H. Precht and J. M. Gómez, "Towards GDPR Enforcing Blockchain Systems," in *International Conference on Wirtschaftsinformatik*. Springer, 2021, pp. 440–446.
5. J. Polge, J. Robert, and Y. Le Traon, "Permissioned blockchain frameworks in the industry: A comparison," *Ict Express*, vol. 7, no. 2, pp. 229–233, 2021.
6. M.-J. Lagarde, "Security assessment of authentication and authorization mechanisms in ethereum, quorum, hyperledger fabric and corda," *Ecole Polytechnique Federale De lausannne*, pp. 1–99, 2019.
7. Schwalm, Steffen, *DIN SPEC 4997 Privacy by Blockchain Design: A standardised model for processing personal data using blockchain technology.* DIN SPEC, 04 2020.
8. The European Parliament and the Council of European Union, "Regulation (eu) 2016/679 of the european parliament and of the council," 2016. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679
9. IBM, "What are smart contracts on blockchain?" 2022. [Online]. Available: https://www.ibm.com/topics/smart-contracts
10. S. Han and S. Park, "A Gap Between Blockchain and General Data Protection Regulation: A Systematic Review," *IEEE Access*, vol. 10, pp. 103 888–103 905, 2022.
11. Shane McCarron, Applied Testing and Technology and Ben Adida, Creative Commons and Mark Birbeck, Sidewinder Labs and Gregg Kellogg, Kellogg Associates and Ivan Herman, W3C and Steven Pemberton, CWI, "Html+rdfa 1.1 - second edition," 2015. [Online]. Available: https://www.w3.org/TR/html-rdfa/
12. Davari, Maryam and Bertino, Elisa, "Access control model extensions to support data privacy protection based on GDPR," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 4017–4024.
13. D. E. MAJDOUBI, H. EL BAKKALI, and S. SADKI, "Towards Smart Blockchain-Based System for Privacy and Security in a Smart City environment," in *2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech)*, 2020, pp. 1–7.
14. H. Precht and J. Marx Gómez, "Redactable Blockchain-Leveraging Chameleon Hash Functions for a GDPR Compliant Blockchain," in *Konferenzband zum Scientific Track der Blockchain Autumn School 2020*. Hochschule Mittweida, 2020, pp. 66–70.
15. E. Kadena and P. Holicza, "Security Issues in the Blockchain(ed) World," in *2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI)*, 2018, pp. 000 211–000 216.
16. Article 29 Data Protection Working Party, "Opinion 05/2014 on anonymisation techniques," 2014. [Online]. Available: https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf
17. A. E. de Protección de Datos, "Introduction to the hash function as personal data pseudonymisation tecnhnique," October 2019. [Online]. Available: https://edps.europa.eu/sites/default/files/publication/19-10-30_aepd-edps_paper_hash_final_en.pdf

18. Hyperledger, "Certificates management guide," 2022. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/latest/certs_management.html
19. A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Computing Surveys (Csur)*, vol. 51, no. 4, pp. 1–35, 2018.
20. B. Wang, J. Sun, Y. He, D. Pang, and N. Lu, "Large-scale Election Based On Blockchain," *Procedia Computer Science*, vol. 129, pp. 234–237, 2018, 2017 International Conference on identification, information and knowledge in the Internet of Things.
21. Hyperledger, "Using private data in fabric," 2022. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5/private_data_tutorial.html#pd-purge
22. S. Farshid, A. Reitz, and P. Roßbach, "Design of a forgetting blockchain: A possible way to accomplish GDPR compatibility," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.
23. L. Zeng, Z. Shi, S. Xu, and D. Feng, "Safevanish: An improved data self-destruction for protecting data privacy," in *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, 2010, pp. 521–528.
24. S. Biswas, K. Sharif, F. Li, Z. Latif, S. S. Kanhere, and S. P. Mohanty, "Interoperability and Synchronization Management of Blockchain-Based Decentralized e-Health Systems," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1363–1376, 2020.
25. M. Finck, "Smart Contracts as a Form of Solely Automated Processing Under the GDPR," *Max Planck Institute for Innovation and Competition Research Paper*, vol. 19-01, 2019.
26. A. Mirchandani, "The GDPR-blockchain paradox: exempting permissioned blockchains from the GDPR," *Fordham Intell. Prop. Media & Ent. LJ*, vol. 29, p. 1201, 2018.
27. Hyperledger, "Private Data," 2022. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5/private-data/private-data.html
28. hyperledger, "What's New," 2022. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5/whatsnew.html
29. Mezmo, "What is Logspout?: Mezmo," 2022. [Online]. Available: https://www.mezmo.com/blog/what-is-logspout

Appendix B

User Consent - Draft

Information for consent for collection and processing of user data

When you accept to join the BlockPrivGDPR project, you are accepting your personal data collection, that will be referred as *user data*. The General Data Protection Regulation (GDPR) requires the project owners to provide any user that considers joining the below presented information.

The user data will be obtained directly from you and from the ARCADIAN-IoT network companies that are part of the Joint Controllership Agreement designed for the BlockPrivGDPR, namely: DGAVIGI, DGAALERTS, DGAOUTER.

The data collected will be of the following types:

- Contact information (provided exclusively by you)
- Information about how reliable you are when requesting a service from the companies listed above: your "reputation" (this is provided as a grade from 1 (worst) to 10 (best))

When an evaluation regarding a service request is provided, it will be automatically handled by BlockPrivGDPR, that will use it to calculate your average reputation. Even though the storage and calculus are automatic, BlockPrivGDPR does not perform any automated decision making.

The user data will be stored at most until the 31st of December of 2026, however, the system is equipped with an automatism that will automatically erase your user data, if not interacted with in a period of 6 months.

The user data will only be shared with the three named companies above.

The GDPR gives you the following rights:

- To withdraw consent to use your data at any time.
- Of access to all of your data, along with its portability.
- To rectify and erase your data.
- To restrict and object the processing of your data.

When you choose to erase or rectify your data, the processed data will be updated/erased in a near-instant manner, however, the system has a maximum retention period of the old data of 24 hours. Since the system is based on a Blockchain system, there is a remote possibility that it takes up to 24 hours for all of the participants to obtain the updated information.

When you provide your personal data, it is possible that your personal data is temporarily stored in plain text in an access-controlled directory of one or more participants of the Blockchain system. The maximum time your data can be stored in these directories is 6 hours. The reason why this happens is because the participants keep a temporary log of all the objects they receive in a location only accessible by the administrator of the machine. The maximum retention time of this logs is assured by an automated script running in each node, that is mandatory and legally assured in the Joint Controllership Agreement.

DGAVIGI, DGAALERTS, and DGAOUTER are the cooperative responsible parties of your study data, as part of the signed Joint Controllership Agreement. You can contact the parties if you have questions or complaints about the user of your user data, or if you want to make a request regarding your rights. To do so, you can use the following email addresses:

- generalBPGDPR@DGAVIGI.com
- generalBPGDPR@DGAALERTS.com
- generalBPGDPR@DGAOUTER.com

Document based on: https://cphs.berkeley.edu/guide/gdpr_consent.docx

Appendix C

Joint Controllership Agreement - Draft

Joint controllership agreement

General description:

The parties are involved in a collaboration project, in which they together determine how and why personal data is being processed. To comply with the General Data Processing Regulation (GDPR), this agreement defines how organize share and process personal data.

The name of the project is:
BlockPrivGDPR

The main goals and main activities in the project are:
Store Data in a GDPR-compliant Blockchain System

The category of data subjects / the group of people to whom the data relates is:
ARCADIAN-IoT users who consented with reputation data storage

The processing will start at 01/07/2023 and will end at 31/12/2026.

Obligations

All parties are aware of the General Data Protection Regulation and will endeavor to meet all requirements of the GDPR.

Each party will make sure that data subjects receive the required information (as described in article 13 and 14 of the GDPR) when personal data is collected by that party. They will make sure that data subjects have the name of the controller, the purposes of data processing, the legal basis for processing and who receives the data. All this information will be available for the user in the consent provider form and will be kept updated.

Each party agrees to takes reasonable, appropriate technical and organizational measures to protect the personal data, so that the risk of data breaches in minimized. It is the responsibility of each party to ensure that each of their participant Blockchain nodes will be timely executing the node backup data erasure script, according to the specifications of maximum retention time, on the user consent document. It is also the responsibility of each organization to assure that there are no illegal access to the backup data temporarily stored.

Each party will inform all other parties immediately in the case of a serious information security incident. This way, each party can determine if the serious information security incident is a data breach that must be reported. Parties will keep each other informed whether they have reported the data breach as the controlling party, and if and how they have informed data subjects.

Each party will make sure that that data subjects can make a request to exercise their GDPR rights, including the right of access to data, rectification, erasure, restriction of processing and data portability if applicable.

Whenever a party receives a GDPR request from a data subject, it will inform all other parties of the request. All parties will then work together so that the request is fully and completely handled. The first party receiving the request will communicate with the data subject.

If one party is audited by their supervisory authority (e.g. Comissão Nacional de Proteção de Dados) for a joint activity, the other parties will support the audited party, for instance by providing information that is requested by the supervisory authority.

Appendix D

Messages specification

The Gateway accepts the following messages:

- "createConsent-id:<*string*>,flag:<*bool*>" - Message that will create consent for a given user id.
- "readConsent-id:<*string*>" - Message that will query the State Database for a given user id details.
- "updateConsent-id:<*string*>,flag:<*bool*>,blocked:<*bool*>" - Message that will update consent for a given user id, based on the flag and blocked parameters.
- "score-id:<*string*>,score:<*integer*>,name:<*string*>,email:<*string*>" - Message that will create a User/Score asset for a given user id, as long as consent as already been provided.
- "read-id:<*string*>" - Message that will query the restricted details of a given user id.
- "readPvt-id:<*string*>" - Message that will query the private details of a given user id.
- "updateScore-id:<*string*>,score:<*integer*>" - Message that will update a User/Score asset for a given user id, based on the score value provided.
- "updateUser-id:<*string*>,name:<*string*>,email:<*string*>" - Message that will update a User personal data for a given id.
- "other-id:<*string*>,unk1:<*string*>,unk2:<*string*>,pvt1:<*string*>,pvt2:<*string*>, pvt3:<*string*>" - Message that will create an Other/Unknown asset. It allows for up to two restricted parameters, and three private ones. PDA will be used to evaluate the sent parameters.
- "readUnk-id:<*string*>" - Message that will query the restricted details of a given Other/Unknown asset id.

- "readUnkPvt-id:<*string*>" - Message that will query the private details of a given Other/Unknown asset id.
- "rightOfAccess-id:<*string*>" - Message that will query all of the details regarding a given user id. More information of what information is provided in section 7.2.1.

As a response, the Gateway will send one of the two messages:

- "Success: <*returned value/object from chaincode execution*>"
- "Failure: <*failure message received from chaincode execution*>"

Appendix E

GDPR.EU GDPR compliance checklist

This checklist is available on GDPR.EU website [3]. The ✓ symbols mean that we believe that BlockPrivGDPR has the necessary mechanisms to handle it, however, the × symbols mean that we believe it has not, and we include some information in *italic* as to why.

This checklist is meant to help organizations prepare for GDPR compliance, and therefore some of the bullets do not apply to our system evaluation, being marked with the × symbol.

- **Lawful basis and transparency**
  - ✓ Conduct an information audit to determine what information you process and who has access to it.
  - ✓ Have a legal justification for your data processing activities.
  - ✓ Provide clear information about your data processing and legal justification in your privacy policy.
- **Data security**
  - ✓ Take data protection into account at all times, from the moment you begin developing a product to each time you process data.
  - ✓ Encrypt, pseudonymize, or anonymize personal data wherever possible.
  - × Create an internal security policy for your team members, and build awareness about data protection. *This would be done on the organization level, and not on the system level.*
  - × Know when to conduct a data protection impact assessment, and have a process in place to carry it out. *This would be done on the organization level, and not on the system level.*
  - × Have a process in place to notify the authorities and your data subjects in the event of a data breach. *This would be done on the organization level, and not on the system level.*

- **Accountability and governance**
  - × Designate someone responsible for ensuring GDPR compliance across your organization. *This would be done on the organization level, and not on the system level.*
  - ✓ Sign a data processing agreement between your organization and any third parties that process personal data on your behalf.
  - × If your organization is outside the EU, appoint a representative within one of the EU member states. *This would be done on the organization level, and not on the system level.*
  - × Appoint a Data Protection Officer (if necessary). *This would be done on the organization level, and not on the system level.*
- **Accountability and governance**
  - ✓ It's easy for your customers to request and receive all the information you have about them.
  - ✓ It's easy for your customers to correct or update inaccurate or incomplete information.
  - ✓ It's easy for your customers to request to have their personal data deleted.
  - ✓ It's easy for your customers to ask you to stop processing their data.
  - ✓ It's easy for your customers to receive a copy of their personal data in a format that can be easily transferred to another company.
  - ✓ It's easy for your customers to object to you processing their data.
  - ✓ If you make decisions about people based on automated processes, you have a procedure to protect their rights.

Appendix F

System Demonstration

In this appendix we will show, with more detail, how the requests are made, and the possible system responses to them.

F.1 JCA Smart Contract interaction

Unlike AMSC, JCASC is not user oriented, but rather Data Controller oriented, which means, that all of the data produced and created by this Smart Contract are meant to be used only by the Data Controller(s). Having that in mind, and for security reasons, all of the JCASC actions are not supported via Gateway, and have to be done by controllers via manual invoke.

For Data Controllers to be able to perform this operation, they must first load their peer configuration, information, certificate, and Message Service Provider (MSP) configuration path, as shown in figure F.1.

```
 export PATH=${PWD}/../bin:$PATH
export FABRIC_CFG_PATH=$PWD/../config/
export CORE_PEER_TLS_ENABLED=true
export CORE_PEER_LOCALMSPID="Org1MSP"
export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org1.e
xample.com/peers/peer0.org1.example.com/tls/ca.crt
export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.examp
le.com/users/Admin@org1.example.com/msp
export CORE_PEER_ADDRESS=localhost:7051
```

Figure F.1: Variable set for direct Blockchain interaction

After having everything ready, it's possible to create a JCAAssetConsent object, by requesting it directly to the Orderer (figure F.2).

```
 peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.ex
ample.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/
orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mycha
nnel -n JCA -c '{"function":"CreateAssetConsent","Args":["Org1MSP","true","true"
]}'
2023-07-24 17:02:33.815 WEST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery ->
Chaincode invoke successful. result: status:200
```

Figure F.2: Direct invoke for JCA consent via terminal

If the Controller tries to create the JCAAssetConsent again, an error will be returned (figure F.3).

```
 peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.ex
ample.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/
orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mycha
nnel -n JCA -c '{"function":"CreateAssetConsent","Args":["Org1MSP","true","true"
]}'
Error: endorsement failure during invoke. response: status:500 message:"this ass
et already exists: Org1MSP{\"ID\":\"Org1MSP\",\"ReadFlag\":true,\"ConsentFlag\":
true,\"owner\":\"x509::CN=org1admin,OU=admin,O=Hyperledger,ST=North Carolina,C=U
S::CN=ca.org1.example.com,O=org1.example.com,L=Durham,ST=North Carolina,C=US\"}"
```

Figure F.3: Error returned on duplicated asset creation

In terms of logging assets, the Controllers can obtain them either by proving a range of ID, or by providing an exact ID, as demonstrated in figure F.4.

```
ubuntu@ubuntu-VirtualBox:~/go/src/github.com/ubuntu/fabric-samples/test-network$
 peer chaincode query -C mychannel -n JCA -c '{"function":"GetLogAssetByRange","
Args":["4","6"]}'
[{"ID":"4","Action":"CreateUnknownAsset","Peer":"x509::CN=user1,OU=client,O=Hype
rledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O=org1.example.com,L=Durh
am,ST=North Carolina,C=US","JCAAnswer":true},{"ID":"5","Action":"CreateUnknownAs
set","Peer":"x509::CN=user1,OU=client,O=Hyperledger,ST=North Carolina,C=US::CN=c
a.org1.example.com,O=org1.example.com,L=Durham,ST=North Carolina,C=US","JCAAnswe
r":true}]
ubuntu@ubuntu-VirtualBox:~/go/src/github.com/ubuntu/fabric-samples/test-network$
 peer chaincode query -C mychannel -n JCA -c '{"function":"ReadLog","Args":["4"]
}'
{"ID":"4","Action":"CreateUnknownAsset","Peer":"x509::CN=user1,OU=client,O=Hyper
ledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O=org1.example.com,L=Durha
m,ST=North Carolina,C=US","JCAAnswer":true}
```

Figure F.4: Direct logging assets query and response

F.2 Asset Management Smart Contract interaction

AMSC actions are expected to occur all via Gateway, and to be requested by either user, or by the interacting entities of ARCADIAN-IoT. All of the output shown will be from the Gateway component, except when mentioned otherwise. Some of the errors (e.g. asset already exists, or asset does not exist) occur regarding all types of assets, and will be shown only regarding one of them to avoid repetition.

As shown in the Smart Contract flows (subsection 6.2.5), the first thing verified is if the Organization of the user has provided consent to the JCA in place. If there's no consent given, the requested operation will not take place, and a failure response will be returned (figure F.5).

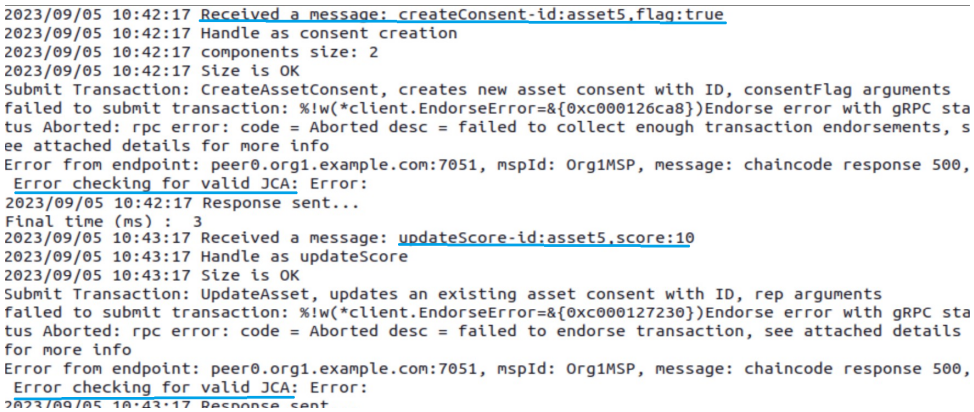

Figure F.5: AMSC execution failure as the organization hasn't provided JCA consent

To create consent, a CreateConsent message has to be handled by the Gateway. Figure F.6 shows two requests for the creation of an asset consent with the same ID: the first one results in consent creating, while the second one fails, and returns an error.

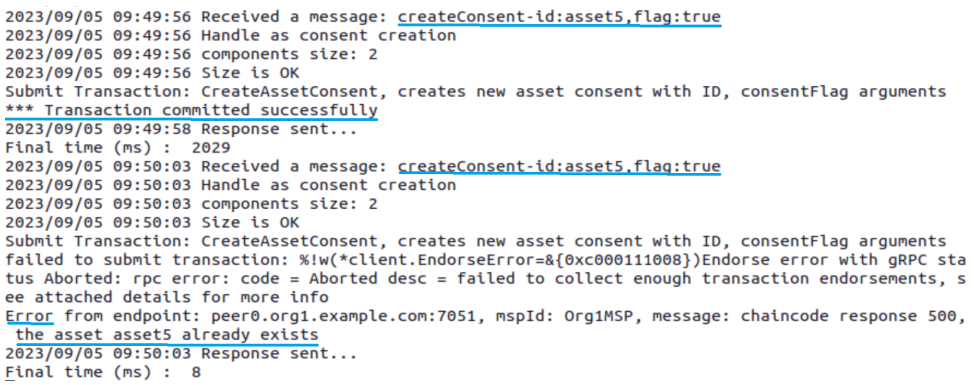

Figure F.6: Consent creation example

If a valid Consent asset exists with the consent flag set to true, and the blocked flag set to false, it is possible to create a user/score asset, read its restricted and private details, and to exercise the Right of Access, figures F.7, F.8, F.9.

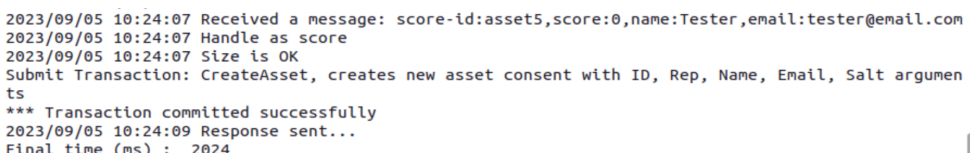

Figure F.7: User/score asset creation example

```
2023/09/05 10:55:42 Received a message: readConsent-id:asset5
2023/09/05 10:55:42 Handle as consent read
2023/09/05 10:55:42 Size is OK
Evaluate Transaction: ReadAssetConsent, function returns asset attributes
*** Result:{
 "ID": "asset5",
 "ConsentFlag": true,
 "BlockedFlag": false
 }
2023/09/05 10:55:42 Response sent...
Final time (ms) :  3
2023/09/05 10:55:45 Received a message: read-id:asset5
2023/09/05 10:55:45 Handle as asset read
2023/09/05 10:55:45 Size is OK
Evaluate Transaction: ReadAsset, function returns asset attributes
*** Result:{
 "ID": "asset5",
 "rep": 0,
 "owner": "x509::CN=user1,OU=client,O=Hyperledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O
=org1.example.com,L=Durham,ST=North Carolina,C=US"
 }
2023/09/05 10:55:45 Response sent...
Final time (ms) :  4
2023/09/05 10:55:46 Received a message: readPvt-id:asset5
2023/09/05 10:55:46 Handle as asset private read
2023/09/05 10:55:46 Size is OK
Evaluate Transaction: ReadAssetPrivateDetails, function returns asset attributes
*** Result:{asset5 Tester tester@email.com hiddenForSecurityReasons %!s(int=0) %!s(int=0)}
2023/09/05 10:55:46 Response sent...
Final time (ms) :  3
```

Figure F.8: Consent, restricted and private user details query

```
2023/09/05 10:13:01 Received a message: rightOfAccess-id:asset5
2023/09/05 10:13:01 Handle as right of access
2023/09/05 10:13:01 Size is OK
Evaluate Transactions: ReadAssetPrivateDetails + ReadAsset, function returns asset attributes
Success querying data. Here it is:
Restricted data result:%s
 {
 "ID": "asset5",
 "rep": 5,
 "owner": "x509::CN=user1,OU=client,O=Hyperledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O
=org1.example.com,L=Durham,ST=North Carolina,C=US"
 }
Private data result:%s
 {
 "ID": "asset5",
 "name": "Tester",
 "email": "tester@email.com",
 "salt": "hiddenForSecurityReasons",
 "sCounter": 1,
 "sSum": 5
 }
Purpose of the data: Calculate Reputation Score, based on input from entities who provided service
s
Restricted data shared with: Org1MSP, Org2MSP. Private data shared with Org1MSP
Storage time: 6 months without interaction and/or until 31/12/2026.
Source of the data: AIoT Environment.
Automated decision making envolved?: No. All processes in the Blockchain are automated, however, t
here's no decision making.
2023/09/05 10:13:01 Response sent...
Final time (ms) :  6
```

Figure F.9: Right of access query example

It is also possible to update user data (figure F.10):

```
2023/09/05 17:37:21 Received a message: updateUser-id:asset5,name:NewName,email:new@email.com
2023/09/05 17:37:21 Handle as updateUser
2023/09/05 17:37:21 Size is OK
Submit Transaction: UpdateAssetPersonal, updates personal data with ID, Name, Email arguments
*** Transaction committed successfully
2023/09/05 17:37:23 Response sent...
Final time (ms) :  2020
2023/09/05 17:37:32 Received a message: readPvt-id:asset5
2023/09/05 17:37:32 Handle as asset private read
2023/09/05 17:37:32 Size is OK
Evaluate Transaction: ReadAssetPrivateDetails, function returns asset attributes
*** Result:{asset5 NewName new@email.com hiddenForSecurityReasons %!s(int=0) %!s(int=0)}
2023/09/05 17:37:32 Response sent...
Final time (ms) :  3
```

Figure F.10: Update Personal Data example

However, if there is no consent asset, or there is, but the consent flag is set to false,

or the blocked flag is set to true, the above operations will not be allowed, and will result in errors (figures F.11 and F.12).

```
2023/09/05 09:52:32 Received a message: score-id:asset5,score:5,name:Tester,email:tester@email.com
2023/09/05 09:52:32 Handle as score
2023/09/05 09:52:32 Size is OK
Submit Transaction: CreateAsset, creates new asset consent with ID, Rep, Name, Email, Salt argumen
ts
failed to submit transaction: %!w(*client.EndorseError=&{0xc0002b85a0})Endorse error with gRPC sta
tus Aborted: rpc error: code = Aborted desc = failed to collect enough transaction endorsements, s
ee attached details for more info
Error from endpoint: peer0.org1.example.com:7051, mspId: Org1MSP, message: chaincode response 500,
 No consent given
2023/09/05 09:52:32 Response sent...
Final time (ms) :  9
```

Figure F.11: User/score asset fails to be created, as the user hasn't provided consent

```
2023/09/05 09:53:45 Received a message: score-id:asset5,score:5,name:Tester,email:tester@email.com
2023/09/05 09:53:45 Handle as score
2023/09/05 09:53:45 Size is OK
Submit Transaction: CreateAsset, creates new asset consent with ID, Rep, Name, Email, Salt argumen
ts
failed to submit transaction: %!w(*client.EndorseError=&{0xc000111d40})Endorse error with gRPC sta
tus Aborted: rpc error: code = Aborted desc = failed to collect enough transaction endorsements, s
ee attached details for more info
Error from endpoint: peer0.org1.example.com:7051, mspId: Org1MSP, message: chaincode response 500,
 Processing is blocked (RoO, RoR)
2023/09/05 09:53:45 Response sent...
Final time (ms) :  8
```

Figure F.12: User/score asset fails to be created, as the user has blocked processing

When querying the user data, both organization can query public, and restricted data, however, only the owner of the data can query the private data. When an unauthorized party tries to access private data that belongs to other organization, the result will be as presented in figure F.13.

```
2023/09/05 10:55:22 Received a message: readConsent-id:asset5
2023/09/05 10:55:22 Handle as consent read
2023/09/05 10:55:22 components size: 1
2023/09/05 10:55:22 Size is OK
2023/09/05 10:55:22 idval: asset5
Evaluate Transaction: ReadAssetConsent, function returns asset attributes
*** Result:{
"ID": "asset5",
"ConsentFlag": true,
"BlockedFlag": false
}
2023/09/05 10:55:24 Received a message: read-id:asset5
2023/09/05 10:55:24 Handle as asset read
2023/09/05 10:55:24 components size: 1
2023/09/05 10:55:24 Size is OK
2023/09/05 10:55:24 idval: asset5
Evaluate Transaction: ReadAsset, function returns asset attributes
*** Result:{
"ID": "asset5",
"rep": 0,
"owner": "x509::CN=user1,OU=client,O=Hyperledger,ST=North Carolina,C=US::CN=ca.org1.example.com,O=o
rg1.example.com,L=Durham,ST=North Carolina,C=US"
}
2023/09/05 10:55:26 Received a message: readPvt-id:asset5
2023/09/05 10:55:26 Handle as asset private read
2023/09/05 10:55:26 components size: 1
2023/09/05 10:55:26 Size is OK
2023/09/05 10:55:26 idval: asset5
Evaluate Transaction: ReadAssetPrivateDetails, function returns asset attributes
*** BAD Result:
*** The asset either doesn't exist, or you don't have permission to access it.
```

Figure F.13: Member of organization 2 tries to query organization 1 user data

Another error possibility is regarding multiple simultaneous updates to the same asset. When Hyperledger Fabric detects this, it generates the error shown in figure F.14 to avoid the creation of invalid transactions.

```
2023/09/05 10:24:43 Received a message: updateScore-id:asset5,score:10
2023/09/05 10:24:43 Handle as updateScore
2023/09/05 10:24:43 Size is OK
Submit Transaction: UpdateAsset, updates an existing asset consent with ID, rep arguments
2023/09/05 10:24:44 Received a message: updateScore-id:asset5,score:10
2023/09/05 10:24:44 Handle as updateScore
2023/09/05 10:24:44 Size is OK
Submit Transaction: UpdateAsset, updates an existing asset consent with ID, rep arguments
*** Transaction committed successfully
2023/09/05 10:24:45 Response sent...
Final time (ms) :  2029
failed to submit transaction: %!w(*client.CommitError=&{transaction 82eca55c4373940d8ff8e1b745d516
01e14bf0ee0c8d8b43a538468e251e472b failed to commit with status code 11 (MVCC_READ_CONFLICT) 82eca
55c4373940d8ff8e1b745d51601e14bf0ee0c8d8b43a538468e251e472b 11})Transaction 82eca55c4373940d8ff8e1
b745d51601e14bf0ee0c8d8b43a538468e251e472b failed to commit with status 11: transaction 82eca55c43
73940d8ff8e1b745d51601e14bf0ee0c8d8b43a538468e251e472b failed to commit with status code 11 (MVCC_
READ_CONFLICT)
2023/09/05 10:24:45 Response sent...
Final time (ms) :  1659
```

Figure F.14: Update score mvcc_read_conflict error

When it's requested via the Gateway, the private databases are automatically selected, given the organization of the requesting user, however, if a user tries to bypass the Gateway and tries to query from a database that he/she does not have access, an access error is returned (figure F.15).

```
ubuntu@ubuntu-VirtualBox:~/go/src/github.com/ubuntu/fabric-samples/test-network$
peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.exam
ple.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/ord
erers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel
 -n basic -c '{"function":"ReadAssetPrivateDetails","Args":["Org2MSPPrivateCollec
tion","asset5"]}'
Error: endorsement failure during invoke. response: status:500 message:"failed to
 read asset details: GET_STATE failed: transaction ID: b96e7a0a508c84061b93812867
47a1b2b5b570aaf5c1bfb39fd283d55f69617c: tx creator does not have read access perm
ission on privatedata in chaincodeName:basic collectionName: Org2MSPPrivateCollec
tion"
```

Figure F.15: Org1 user fails trying to access Org2 private data