Carlos Aparecido Serrato Júnior

A Privacy Preserving System to Consult Public Institutions Records

1 2 9 0
UNIVERSIDADE D COIMBRA

Carlos Aparecido Serrato Júnior

A Privacy Preserving System to Consult Public Institutions Records

**Internship Report in the context of the
Master's in Informatics Engineering, Specialization in Software Engineering,
advised by Professor Nuno Antunes and by Professor Marilia Curado and presented to
Faculty of Sciences and Technology / Department of Informatics Engineering.**

October 2020

This page is intentionally left blank.

Acknowledgements

To my father Carlos Serrato, for embracing my idea of studying across Brazilian borders and allowing me to be who and where I am today.

To my supervisors Professor Nuno Antunes and Professor Marilia Curado, for the knowledge, patience, belief and being a reference model of individuals and professors.

To the PoSeID-on UC team, Professor Edmundo Monteiro, Professor Fernando Boavida, Professor Paulo Simões and Dr. Paulo Silva, for the amazing environment, lessons and opportunities provided.

To my family, for supporting and motivating me through my entire life.

To my girlfriend's family, for accommodating and helping me, whenever I needed in Portugal.

To my dearest friends, for the advice, laugh, and conversations, even from afar.

To my girlfriend, for the love, everyday learning and never giving up on us.

This page is intentionally left blank.

Abstract

In the past decades, public and private organizations have been improving the way of dealing with information. Nowadays, both the quantitative and qualitative aspects of information processing are important drivers of business strategies and market decisions.

The regular processing of personal information creates a risk as serious as the sensitivity of the data being processed. In that sense, personally identifiable information demands special treatment to safeguard the data owner from possible threats such as identity theft and banking fraud.

With the enforcement of the EU GDPR, organizations increased dramatically the attention towards personal data protection and individual's privacy. Ensuring "Data protection by design and by default" as specified in Art. 25 of GDPR, means taking those concerns into account since the very early development stage of new systems.

The modernization and digitalization of public institutions' processes have made the privacy of personally identifiable information an even more complex challenge, while complying with the GDPR.

To facilitate the management of personally identifiable information, the PoSeID-on H2020 project emerged, which foresees the development of a platform on which individuals can manage the sharing of personal data with organizations and services. Besides, the platform ensures compliance with GDPR for both individuals and organizations.

This work presents the design and development of a reputation-based privacy-preserving system for the PoSeID-on platform under its Risk Management Module. The developed reputation system ensures the establishment and management of trust between the parties while safeguarding an individual's rights.

To this end, it is presented the state of the art study regarding reputation systems, blockchain, smart contracts, and GDPR reflecting upon their usability into a public institution's scenario. The overall contributions for the project, along with the specification and validation of its reputation system, are addressed in this thesis.

Keywords

Reputation Systems, Trust, Risk, Personal Identifiable Information, Privacy

This page is intentionally left blank.

Resumo

Nas últimas décadas, as organizações públicas e privadas vêm melhorando a maneira de lidar com informações. Atualmente, tanto os aspectos quantitativos quanto os qualitativos do processamento de informações são importantes impulsionadores das estratégias de negócios e das decisões de mercado.

O processamento regular de informações pessoais cria um risco tão sério quanto a sensibilidade dos dados que estão sendo processados. Nesse sentido, informações pessoalmente identificáveis exigem um tratamento especial para proteger o proprietário dos dados de possíveis ameaças como roubo de identidade e fraude bancária.

Com a implementação do Regulamento Geral de Proteção de Dados da UE, as organizações aumentaram drasticamente a atenção à proteção de dados pessoais e à privacidade dos indivíduos. Garantir "proteção de dados por projeto e por padrão", conforme especificado no art. 25 do RGPD significa levar essas preocupações em consideração desde o estágio inicial de desenvolvimento de novos sistemas.

A modernização e a digitalização dos processos das instituições públicas tornaram um desafio ainda mais complexo a garantia da privacidade das informações de identificação pessoal, ao mesmo tempo em que obedecem ao RGPD.

Para facilitar o gerenciamento de informações de identificação pessoal, surgiu o projeto PoSeID-on H2020, que prevê o desenvolvimento de uma plataforma na qual os indivíduos possam gerenciar o compartilhamento de dados pessoais com organizações e serviços. Além disso, a plataforma garante a conformidade com o RGPD para indivíduos e organizações.

Este trabalho apresenta o design e o desenvolvimento de um sistema de preservação da privacidade baseado em reputação para a plataforma PoSeID-on no seu Módulo de Gerenciamento de Riscos. O sistema de reputação desenvolvido garante o estabelecimento e o gerenciamento da confiança entre as partes, salvaguardando os direitos de um indivíduo.

Para isso, é apresentado o estudo no estado da arte sobre sistemas de reputação, blockchain, contratos inteligentes e RGPD, refletindo sobre sua usabilidade no cenário de uma instituição pública. As contribuições gerais para o projeto, juntamente com a especificação e validação de seu sistema de reputação são abordadas nesta tese.

Palavras-Chave

Sistemas de reputação, Confiança, Risco, Informação Pessoalmente Identificável, Privacidade

This page is intentionally left blank.

Contents

This page is intentionally left blank.

Acronyms

DLT Distributed Ledger Technology. 10

DPO Data Protection Officer. 9

EC European Comission. 3

eID Electronic IDentification. 7

eIDAS Electronic IDentification, Authentication and trust Services. 7

EU European Union. 1, 4, 15

G2B Government to Business. 14

G2C Government to Citizen. 14

G2E Government to Employee. 14

G2G Government to Government. 14

GA General Assembly. 3

GDPR General Data Protection Regulation. 1

JAR Java Archive. 47

JVM Java Virtual Machine. 47

ML Machine Learning. 29

P2P Peer to peer. 11, 21

PDA Personal Data Analyzer. 1, 6, 38

PII Personally Identifiable Information. 1, 4, 5, 16

PoS Proof of Stake. 12

PoW Proof of Work. 12

RMM Risk Management Module. 1, 2, 6, 38

UUID Universally unique identifier. 43

VAT Value Added Tax. 15

This page is intentionally left blank.

List of Figures

This page is intentionally left blank.

List of Tables

This page is intentionally left blank.

Chapter 1

Introduction

With the enforcement of the new European Union (EU) General Data Protection Regulation (GDPR), organizations are held responsible for ensuring individual's privacy and data protection, now more than ever. Taking those concerns into account since the beginning of the development of its new systems is of utmost importance [1].

Considering situations where organizations collaborate with other public agencies in an openly and joint-approach for the sake of service delivery, into what is known as Ecosystem Platform, regulation compliance and trust becomes a heavy obstacle which is sometimes too difficult to overcome [2].

The rise of the e-government concept aims to provide the digitalisation and modernisation of public administrations as a mean to improve and focus on innovative, user-centric approaches towards public service deliveries [3].

Aligned with digital provision, PoSeID-on, which stands for Protection and Control of Secured Information by Means of a Privacy Enhanced Dashboard, is an H2020 project that "aims to develop and deliver an innovative intrinsically scalable platform as an integrated and comprehensive solution aimed to safeguard the rights of data subjects (i.e. all those natural persons that represent the primary target of the new GDPR), as well as support organizations in data management and processing while ensuring GDPR compliance" [2].

The premise of the project is to leverage the use of cutting-edge technologies such as Blockchain and Smart Contracts, along with specific modules such as: 1) The Risk Management Module (RMM) for detecting and assessing security risks on PoSeID-on; 2) The Personal Data Analyzer (PDA) for detecting and preventing anomalies and misbehaved transactions; 3) The Web Dashboard to provide the necessary interface for data subjects to access the PoSeID-on platform.

The project envisions three different system actors that can interact with the platform. Data subjects represent the majority of PoSeID-on's users, which will use the system to manage their own Personally Identifiable Information (PII). Data processors are the personification of the designation given by GDPR for both data processors and data controllers, that are in charge of storing, processing and exchanging data subject's PII. Lastly, the system also expects an administrator as an actor, responsible for managing the operations within the PoSeID-on platform.

As one of PoSeID-on's requirements, each data processor must be associated with a reputation score in order to improve risk assessment within the platform. For that, there is a *need for the development of a reputation system that will reside in the RMM.*

Research about reputation systems has been evolving side-by-side with the ecosystem platform previously described. This has to do with the need of establishing and managing trust between the parties involved in the transactions, regardless of the service (e.g. e-commerce, e-health) delivered [4]. Reputation systems play a crucial role in such environments by collecting, distributing and aggregating reputation values for the selected target(s), becoming an important component in order to ensure trust and security.

With the already existence of several reputation systems both for academic and commercial purposes, selecting an appropriate one to be incorporated requires an evaluation of its applicability, structure, and fulfillment of trust challenges raised by PoSeID-on.

1.1 Objectives

This work aims to contribute towards the development of the PoSeID-on project by incorporating a reputation system mechanism which should be robust, secure and consistent on providing a reputation value for the intended target. Moreover, a contribution to the state-of-the-art related to the reputation system adopted and its applicability is one of the main goals expected to be fulfilled.

The detailed objectives of this work are the following:

1. Study the state-of-the-art in reputation systems.
2. Select and justify a candidate reputation system to be incorporated into the RMM.
3. Design and develop an approach to implement the reputation system.
4. Test and validate the results gathered from the reputation system developed.
5. Develop and validate the batch processing layer of the RMM.

1.2 Contributions

Initially, a set of possible contributions to the PoSeID-on project was explored, leading to the study of the state-of-the-art of technologies such as Blockchain and Smart Contracts. After analysis and considering the project objectives, it was decided that the focus of the author's work should be related to reputation systems.

The contributions and outputs achieved from this work were divided into three different categories and are presented following:

1. **Reputation systems**
    - Design decision regarding the reputation system to be embedded into the RMM.
    - Implementation and validation of the developed reputation system.
    - Extending the reputation system to be incorporated into a distributed environment.
2. **RMM lead development**
    - Development of the layer responsible for batch processing and its subcomponents.

- Updating the RMM's architecture to cope with continuous improvements made over the course of the project.

3. **Project management and documentation**

   - An overview about the PoSeID-on project, GDPR, blockchain, smart contracts and reputation systems.
   - Documentation of the Risk Management Module Final implementation part of Deliverable 4.4 of the PoSeID-on project.
   - Participation and presentation of the RMM status in PoSeID-on's General Assembly (GA) and to the European Comission (EC) Review.

Moreover, the author expects to submit a paper as the outcome of this work (and the months to follow), until the end of the project.

1.3 Thesis Structure

The structure of the rest of this document is the following:

Chapter 2 introduces the background and related work. It starts with an overview about the PoSeID-on project, followed by a brief introduction of GDPR. Then, it is presented an overview about blockchain, smart contracts, it's use cases and tension with GDPR. Finally, it is given a description about key aspects of reputation systems, including their models, aggregation methods and existing solutions.

Chapter 3 presents a detailed description about the author's contribution towards the PoSeID-on project, namely to the Risk Management Module (RMM). The contributions were divided with respect to the specific layers in which the development and improvements were made, the first one being the batch layer and the second one, the service layer.

Chapter 4 exposes the details of the developed reputation system, by covering its objectives, proposed solution, architectural scenarios and aggregation of reputation.

Chapter 5 describe the results and discussion provided by the evaluation of the reputation system, through the details of the dataset used, experimental setup and finally, the evaluation performed.

Chapter 6 concludes with the final considerations and main conclusions from the author, along with directions for future work.

Chapter 2

Background and Related Work

This chapter presents the theoretical background and related work about core subjects of this thesis, in order to contextualize the reader. At first, in Section 2.1 it is presented an overview about the PoSeID-on project, which encompasses the context of this work. Section 2.2 gives a brief introduction about GDPR, addressing its key terms, principles and human rights. Furthermore, Section 2.3 incorporates the theoretical perspective of blockchain, smart contracts and their applicability. Finally, Section 4 describe the state-of-the-art regarding reputation systems.

2.1 PoSeID-on project

This section starts by providing a brief introduction to the motivation behind the PoSeID-on project, its goals and current architecture along with its components, allowing readers to become more familiar with the project.

2.1.1 Motivation and Goal

Nowadays, both governments and private organizations strive to deal with a multitude of challenges, such as increasing effectiveness and innovation of the delivered services. Further, customer's expectations and budget limitation make those challenges even more complex.

To deal with this scenario, public organizations can create what is called Ecosystem Platform: a collaborative and innovation-focused approach where governments openly collaborate with citizens, companies, other government organizations for the sake of service delivery [2].

There are several barriers related to Personally Identifiable Information (PII) that need to be overcome such as standardization, individuals and organizations trust, and maybe the most important one, regulation compliance.

From the regulation perspective, GDPR sets a obligatory step on the establishment of any ecosystem platform. Every organization inside and outside the ecosystem plaform, selling goods or processing data in the European Union (EU) will be held accountable to whose data they process. Surely, GDPR has a significant impact for organizations, with new requirements and for individuals as well, with new rights.

With that in mind, PoSeID-on's goal is to develop a platform that will allow for a transparent and accountable ecosystem for personal data protection, in compliace with GDPR with respect to security. PoSeID-on leverages the use of innovative technologies like blockchain, cloud and smart contracts, enabling end-users to manage their personal data in an intuitive manner.

The project was designed to target both the public and private entities, helping them to identify new business opportunities and process personal data while remaining GDPR compliant. Moreover, it will impact the society as well, as it will support the assurance of their fundamental digital rights and also increase the trust in the digital market.

The PoSeID-on Consortium is composed by ten participants, besides two third-parties, from seven different European countries (Italy, Spain, Netherlands, Portugal, France, Malta, Belgium) and from which three are public administrations (Ministero dell'Economia e delle Finanze, Malta Information Technology Agency, Santander city), three large industries (Accenture, Softeam, PNO), two SMEs (E-lex and JIBE), one research center (Tecnalia) and one university (University of Coimbra).

PoSeID-on will be evaluated through four different pilot studies, in different countries (Italy, France, Spain and Austria) and contexts (public, private and mixed). For the Italian pilot, the goal is to improve their e-services for public officials; the Spanish pilot aims to enhance e-Government services for Santander citizens; the Austrian aims to help business to better offer and sponsor their services to customers; the French pilot targets at making their e-services simpler to French citizens. Every pilot will run in a controlled environment and at first, they will involve a small set of users to be increased during the evaluation process.

2.1.2 Architectural Overview

PoSeID-on's architecture represents the sum of the original project concept, an extensive discussion process, which culminated into identified user and system requirements, all performed on an interactive basis. Figure 2.1 depicts the overview of PoSeID-on's architecture.

There are three actors considered:

- **Data Subjects**: Represents the main target of GDPR. Data subjects have personal data, commonly referred as PII, to be shared with third-parties and that represents a privacy risk.
- **Data Processors**: Designates the third-parties exchanging PII with data subjects or between themselves.
- **Administrators**: Represents the users, working on behalf of a preassigned third-party, in charge of managing the PoSeID-on platform.

Also, PoSeID-on's architecture takes into consideration several modules/components. A brief description of each one is given as follows:

- **Web Dashboard**: It is a web-based application representing the interface provided to data subjects, in order to access the PoSeID-on platform (it is not the platform itself). Authentication is performed with the data subject's authorized credentials (eID or a similar one, in case of a specific PoSeID-on instance). The dashboard

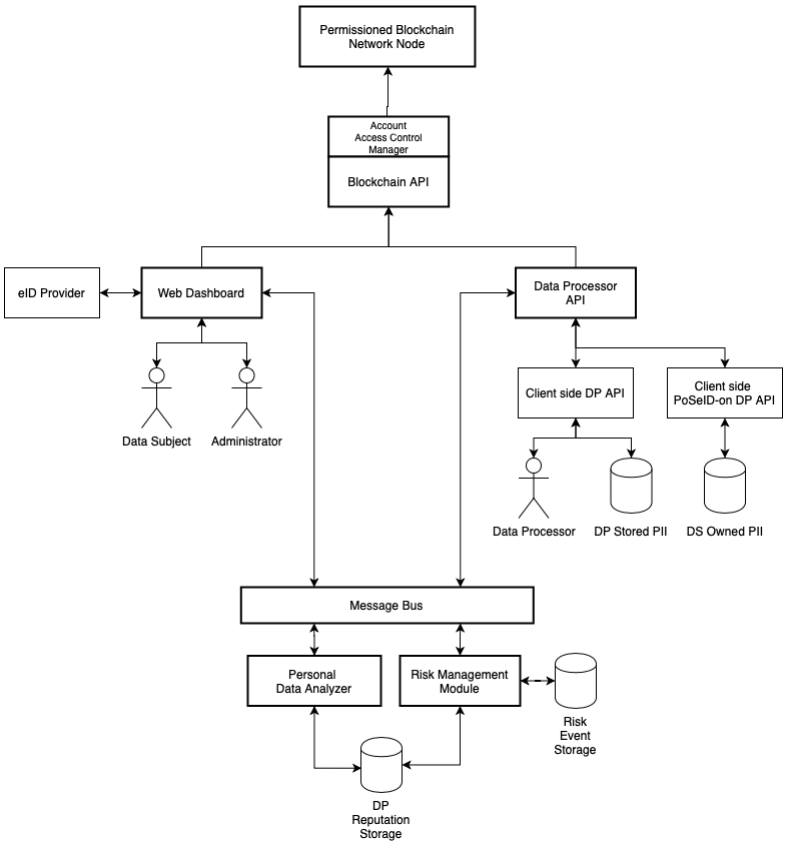

Figure 2.1: PoSeID-on Overall Architecture (from [5]).

serve as an interface for PoSeID-on's administrators as well, but with a different set of functionalities.

- **Risk Management Module (RMM)**: It is the module responsible to detect and assess possible security risks by monitoring PoSeID-on's actors operations. For example, successive attempts to login or a data processing that is requesting an anomalous amount of personal data, are possible events that will trigger the module. Risk detection is performed combining machine learning algorithms with user-level and system-level behaviours. When triggered, the module may send alerts to administrators and data subjects, according to the RMM settings.

- **Personal Data Analyzer (PDA)**: The PDA is responsible to detect and prevent anomalous transactions in the blockchain platform, by monitoring personal data and blockchain related warnings. A warning can be generated in transactions, where a personal data for which there is no data subject authorisation occur. Since the module needs to analyze personal data, an explicit consent from the related data subject is mandatory so that the PDA can work. If the consent is not given, PDA will not operate on personal data for that particular data subject.

- **Permissioned Blockchain and Smart Contracts**: The so-called permissioned blockchain is a special type of blockchain, where a participant needs to be given permission by a central authority in order to be part of the network, and also, it is designed to work only within the PoSeID-on platform. Smart contracts will handle the system functionalities, describing the management of permissions related to PII access.

- **Blockchain API**: It is an abstraction of all blockchain functionalities to ease the

integration with other applications. The access to the API will be provided only once the user has been authenticated by the Electronic IDentification (eID) Provider.

- **Data Processor API**: Considered as the access point for Data Processors, its goal is to publish an authenticated API, where Data Processors can send request about/for PII to PoSeID-on. Basically, it interconnects the business logic of Data Processors with the PoSeID-on platform.

- **Client-side Data Processor API**: It serves as an interface through which Data Processors can access their stored PII. Also, it handles PII transportation and permission revocation.

- **Message Bus**: It is responsible for providing the messaging infrastructure through which components will be able to communicate with each other. The message bus enables scalability (easy addition and removal of components), asynchronous communication and fault tolerance. Whenever a components is not reachable, the message is kept until either a timeout occurs or the component comes back.

- **Data Subject's PII Repository**: This component is responsible for storing all data subject's PII for which no Data Processor exists. In order to store its PII, data subject will always use a data processor, even when this data processor has no purpose, acting just as a storage point. All data stored within this component is encrypted with the Data Subject's public key.

- **eID Provider**: Represents an identity provider, in charge of authenticating end-users and organizations, following the European Electronic IDentification, Authentication and trust Services (eIDAS) regulations and ecosystem.

2.2 General Data Protection Regulation

GDPR is the result of a historical evolution of technology and human rights. The European Convention on Human Rights, back in 1950, already stated in its article 8, the fundamental human right to privacy.

Since then, the Internet was invented and modern measures of ensuring this right. With that, the EU passed in 1995 the European Data Protection Directive, which included standards to privacy and security, being each member state responsible to implement it.

The decision of working to update the 1995 directive through a more comprehensive approach of securing personal data took place in 2011, due to incidents such as Google suing another company for scanning its email.

Thus was born GDPR and in 2016, after passing the European Parlament, all organizations were required to be compliant by May 25, 2018.

Some definitions were created along with GDPR in order to ease the understand of certain contexts. These definitions are explained in Table 2.1 as follows:

It is also equally important to understand where the GDPR must be applied. The regulation must be complied to any company that processes personal data of EU citizens/residents or offer services (paid or free), regardless of where the company is located.

The penalties for violating the GDPR are substantially high. Article 83 of GDPR states that there are two tiers of fines, being the more severe one translated into a fine of up to

Terms	Definition
Personal data	Information that relates, directly or indirectly, to a person.
Data processing	Any action that is performed on personal data such as collecting, storing and using.
Data subject	The person whose data is processed.
Data processor	A third-party that processes data on behalf of the data controller.
Data controller	The person who decides how and why the personal data should be processed.

Table 2.1: Key definitions of legal terms used in GDPR.

€20 million, or 4% of the firm's worldwide annual revenue from the preceding financial year, whichever amount is higher [6].

2.2.1 Principles

To make it easy for companies to know what they have to comply with when they process personal data, GDPR stands its grounds into seven core principles, each of which are explained following:

1. **Lawfulness, fairness and transparency**: Personal data must be processed in a lawful, fair, and transparent manner to the data subject.
2. **Purpose limitation**: Personal data must be processed only for the purposes that were explicitly specified to the data subject.
3. **Data minimization**: The amount of data to be collected and processed must always be minimized for the necessary purpose.
4. **Accuracy**: Personal data kept must always be up to date.
5. **Storage limitation**: Personal data must be stored just as long as necessary for the specified purpose.
6. **Integrity and confidentiality**: Data processing must be performed in order to ensure security by means of integrity and confidentiality.
7. **Accountability**: Data controller must be held responsible for ensuring GDPR compliance to all seven principles.

2.2.2 Individual rights

From the perspective of the data subject, GDPR now introduces a collection of new privacy rights, giving individuals more control over the personal data they provide to organizations. There are eight privacy rights that need to be ensured by organizations in order to be GDPR compliant, which are:

1. **The right to be informed**: Individuals have the right to be informed about the collection and use of their personal data.
2. **The right of access**: Individuals have the right to access a copy of their personal data.
3. **The right to rectification**: Individuals have the right of rectifying inaccurate personal data, or complete it if it is incomplete.

4. **The right to erasure**: Also known as the “right to be forgotten”, ensures that individuals have the right to erasure of any information related to them.

5. **The right to restrict processing**: Individuals have the right to restrict processing of their personal data.

6. **The right to data portability**: Individuals have the right to receive their personal data in a portable, commonly used format.

7. **The right to object**: Individuals have the right to object to processing their data at any time.

8. **Rights in relation to automated decision making and profiling**: “The data subject shall have 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." [7]

2.2.3 Accountability

Under certain conditions, GDPR obligates organizations that process personal data to appoint an employee in charge of supervising the organization’s GDPR compliance. The Data Protection Officer (DPO) is the employee in question and will have to possess the know-how about data protection law and practices.

The conditions under which the appointment of a DPO is required, are the following:

1. **Public authority**: Personal data processing is performed by a public authority, with exception to courts or other independent judicial authority.

2. **Large scale, regular monitoring**: The processing of personal data is performed on a large scale and it constitutes the main activity of an organization who regularly monitors data subjects systematically.

3. **Large-scale special data categories**: A organization’s main activity consists of large-scale processing of special categories under GDPR’s Article 9 and 10. For example if the organization is a medical office.

Even when not required, an organization can designate a DPO. Its set of tasks can include advising people about their responsibilities, conducting protection training and auditing and of course, monitoring the organization’s GDPR compliance as well.

2.3 Blockchain

One of the main keywords to understand the motivation behind blockchain is trust, that can be seen as “a certain belief in behavior, an expectation of outcome, an expectation that is all the more convincing the more knowledge one holds of the agent to be trusted” [8].

From there, two types of trust relationships appear: direct and indirect. The first one, has the benefit of holding more knowledge from the agents than the second one. Unfortunately, the larger a society becomes more difficult it is to know all the agents.

This is the where the indirect trust relationship turns out to be useful. It eliminates that need and ensures that anything, sent from agent A in Brazil, is delivered successfully to agent B in Japan, in a process also known as transaction.

There are some downsides by using the indirect trust relationship. The first one is the speed. Transactions are much slower, when compared to direct trust relationships for obvious reasons. The second and perhaps, the biggest disadvantage, is dependency and therefore, centralization.

If one intermediary node fails, the entire transaction goes down. The success of the transaction is now centralized, in the hands of the intermediary nodes. Also, most of these transactions are charged, in order to be delivered.

Blockchain came as a solution for this problem, providing both a decentralized system and a way to ensure trust. In the next subsections, a more detailed view of blockchain will be presented.

2.3.1 Definition and Operation

Blockchain can be defined as a type of Distributed Ledger Technology (DLT), that is, “technologies which store, distribute and facilitate the exchange of value between users, either privately or publicly" [9].

In other words, rather than keeping information in one central point, several copies of the same data are stored in different places and devices on the network. This means that even if one piece of information is changed without the agreement of the rightful owners, there are countless other examples in existence, where the information is true, making the false record obsolete [10].

Also, distributed ledgers have three key attributes:

1. **Recorded**: Stored information is time-stamped.
2. **Transparent**: Anyone can see the ledger of transactions.
3. **Decentralized**: The ledger exists on multiple computers, often referred to as nodes.

The name blockchain comes from the way of how data is stored in the technology, which is by packaging it into blocks linked to each other, creating a chain of blocks.

Recording a transaction requires a confirmation from several entities/devices, such as computer/nodes of the network. When the agreement, also known as consensus, is reached between the nodes to store something on a blockchain, it cannot be altered or removed.

While efforts to specialize the taxonomy of blockchain had been carried out [11], for the scope of this work we can segment the blockchain into three different categories [12]:

1. **Public**: They are fully decentralized and anyone can join the network, as such being permissionless.
2. **Consortium**: Only a set of selected nodes can participate in the consensus process, as such being permissioned.
3. **Private**: Only the nodes from a specific organization can join the consensus process, as such being permissioned.

Blockchain consists of a mix of several other technologies, each one being a little piece of a larger picture. The main technologies/concepts that compounds blockchain are:

- **Peer to peer (P2P) network**: A network where each peer/node utilizes and provides the basis of the network at the same time, and there is no central point of storage, instead, information is being constantly recorded and interchanged between all of the participants on the network.

  In the context of blockchain, the nodes can assume different roles. The P2P makes up the network foundation of blockchain and its contribution is rather obvious, since all premises of blockchain rely on this type of network.

- **Cryptography**: Provides the possibility of encrypting/decrypting messages, as well as creating digital signatures.

  These two functionalities are expressed in blockchain in terms of ensuring the past records cannot be tampered and securing the identity of the sender of transactions, through the use of assymetrical cryptography, which provides both a public and a private key.

- **Nodes**: As said before, the nodes can assume different roles and responsibilities. In the case of Bitcoin blockchain there are, predominantly, three types of nodes.

  Full nodes which store a copy blockchain and thus guarantee the security and correctness of the data on the blockchain by validating data. The second type is a lightweight node – each user participating, who needs to connect to a full node in order to synchronize to the current state of the network and be able to participate [13].

  The third type is known as a miner. This node can create/register new blocks of transactions into blockchain and this process is associated with the issuance of new currency, alluding to the precious metals mining process.

- **Hashing**: Is the process of taking an input and producing a fixed size (tipically, given in bits) output. The main difference between hashing and cryptography and the reason why they are both in the list of concepts used by blockchain, is that hashing is a non-reversible process, meaning that one cannot figure out the input, given the output. In cryptography, on the other hand, one can use the public-key to decrypt the message received.

  The security is heavily improved with hashing, because it can provide integrity and reliability. By comparing, the hash of the received information with the real one, the integrity of the information can be checked.

  Blockchain leverages hashing by using, as input, the blocks of transactions so far. Therefore, hashes represent the current state of the blockchain world. “As such, the input represents everything that has happened on a blockchain, so every single transaction up to that point, combined with the new data that is being added" [14].

  If anyone attempts to change any record, this would represent a different hash of the entire blockchain, making them all false and obsolete. Along with that, hashed is used in core of the whole structure of the blockchain.

  Essentially, each block contains: a hash of the previous block (which is the hash of the entire blockchain so far), a timestamp, a nonce and a set of transactions. These transactions are organized via a data structure of hashes called Merkle Tree. The benefit of using this data structure is to allow anyone to confirm the validity of an individual transaction without having to download a whole blockchain.

- **Consensus protocols**: This is the main innovative concept of blockchain. Basically, it help us answer two big questions: 1) How to synchronize the blockchain between its participants?; 2) How do we all make sure that we agree on what the truth is?

  Since anyone can submit information/transaction in the blockchain, it must be revised and confirmed, in a self-auditing environment, in order to ensure that problems like double-spending does not happen. An example of this problem can be, an attacker attempting to spend some cryptocurrency and then reversing the transaction by broadcasting its own version of that blockchain, not including the transaction.

  The aim of the consensus protocols is to ensure, that, from the point-of-view of each participant, the trust no longer must placed in anyone, but in the system itself. The two most known consensus protocols are 1) Proof of Work (PoW); 2) Proof of Stake (PoS), and they will be explained later.

- **Proof of Work**: This proposed consensus mechanism, it was implemented along with Bitcoin in 2008 and it is widely used by other cryptocurrencies. It is highly related to the mining process, where the miners need to solve a mathematical problem, in order to successfully create the new block and claim their reward.

  This mathematical problem is hard to solve, but easily verifiable. Miners nowadays spend about 10 minutes to answer the problem and therefore, spend computational resources and energy. So, even though it is rewardable, one has to spend a very high amount of money to invest in this.

  With the mathematical problem solved, the full nodes then, verify if it is correct, and start to propagate the blockchain with the new block added. Even though it is widely used, PoW has the downside of consuming too much energy (a single bitcoin transaction consumes the same amount of a average Dutch house in two weeks) and therefore, others consensus mechanism were developed.

- **Proof of Stake**: First implemented in 2012 in Peercoin, it has the advantage of being more energy efficient than PoW consensus. In PoS, “the creator of the next block is determined by a randomized system that is, in part, dictated by how much of that cryptocurrency a user is holding or, in some cases, how long they have been holding that particular currency" [15].

  The randomized part is essential in order to avoid centralization and therefore, the richest ones becoming more rich. The PoS also prevents that any individual controls the network by some sort of 51% attack, because he would need the majority of coins to be successful.

The sum of these concepts gave birth to Blockchain. It can be seen that it is a complex scenario, composed of several well-known concepts plus innovation.

2.3.2 Architecture

Blockchain, as said earlier, is a sequence of blocks, which holds a complete list of transaction records like conventional public ledger [16]. Figure 2.2 illustrates the blockchain architecture. The block header always points to the previous block hash (parent block). The first block of a blockchain is called genesis block, and is the only block that has no parent.

The block structure is composed by the block header and block body, as depicted in Figure 2.3.

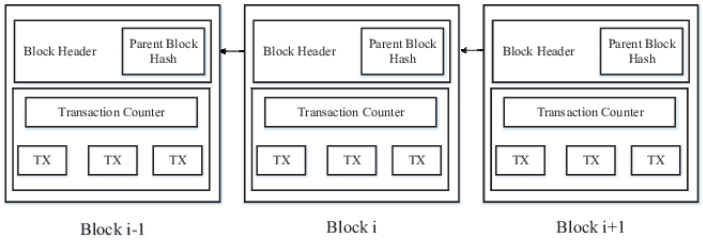

Figure 2.2: Blockchain as a sequence/chain of blocks (from [12]).

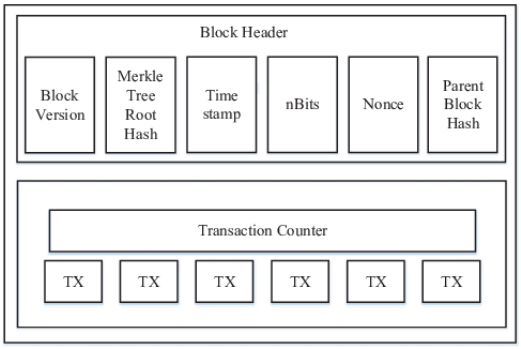

Figure 2.3: Block Structure and Fields (from [12]).

The block body consists of a transaction counter and transactions itself. The number of transactions that a block can hold depends on the size of each transaction and the block size. As for the block header, it consists of the following fields:

- **Block version**: Indicates the correct set of block validation rules.
- **Merkle tree root hash**: The hash value of all the transactions stored in a data structure called Merkle tree.
- **Timestamp**: Current universal time since January 1, 1970.
- **nBits**: Target of a valid block hash.
- **Nonce**: An 4-byte field, used to create a valid block.
- **Parent block hash**: a 256-bit hash value pointing to the previous block.

2.3.3 Smart Contracts

The term “Smart Contract" was invented in 1994 by Nick Szabo [17], an american cryptographer and computer scientist. Basically, a smart contract is a computer code that improves and emulates the performance of a real-world contract. The purpose is to execute the contract by computers in the network, only when certain defined conditions are met [18].

Smart contracts key function is to automate different kinds of processes and operations. They are self-executed contracts, represented in the form of a transaction, which is then appended into the chain of blocks. So, it leverages the whole structure of blockchain (it is witnessed by the network of computers that run the blockchain), while enforcing that the exchange of pretty much anything is successfully performed, in an autonomous way.

The flexibility of setting the conditions and also executing the terms of the contracts, made smart contracts a potential technology in an enormous set of use cases such as supply chain, health systems and insurance companies.

Also, it is worth to highlight that one of the most important features of smart contracts and also blockchain, is the independence and removal of trust from a third-party, which shall translate into lower taxes and faster transaction speed. By setting the rules and penalties associated with an agreement, smart contracts are in charge of automatically enforce those obligations [19].

2.3.4 Blockchain use cases for e-goverment services

Government agencies serve a different number of stakeholders, being a Government to Business (G2B), Government to Citizen (G2C), Government to Employee (G2E) or Government to Government (G2G). In order to improve their service delivery, government agencies turned their attention to technologies/tools to support their work. In fact, the use of digital tools to optimize efficiency and reduce unnecessary costs in the government context is defined as e-government.

A recent technology that caught the eyes of government administrators is Blockchain, due to its intrinsic properties, as described in the previous section. For example, it enables trust in information and processes for a large set of stakeholders and it delivers with an ease, transparency, allowing for, almost effortless, auditing trails and tracking of information. [20]

Blockchain has already been used to provide e-government services. Some real use cases are presented as follows:

1. **Securing and sharing data**: Set of use cases related to the verification and sharing of data.

    - ***Identity***: It is expected that every person in the digital realm would have a unique identity, with ensured security and privacy, and also be able to provide enough proof of identity, without having to appeal to a third-party authority and without disclosing unnecessary information for the transaction. This has been proven to be quite difficult to achieve with old-fashioned centralised technologies and so, governments are turning to blockchain as a potential candidate to pursue this goal. As an real example, in 2017, the city of Zug, in Switzerland, issued the first identity credential using a blockchain-based approach to citizens [21], which they have used for e-voting [22] and renting bikes [23].
    - ***Title/asset registration***: In some sense, registering "title" to an asset is the main functionality of blockchains. Originally considered only crypto-currencies as the asset, nowadays this can be extended to any type of digital form. Therefore, one can consider land registries as the asset to be registered, and in fact, this use case has been used in Africa [24] and India [25] to deal with corruption, where paper-based records are being altered to gain unjustified land tenure. Also, there is the fact of the current slow and expensive process of transferring a title to someone else, in which blockchain can act to simplify such process. Initiatives on this behalf are being worked on, such as the case of the transfer of title that has been successfully carried out in Sweden, on a blockchain-based transaction [26].

- ***Healthcare***: Nowadays, health records are commonly kept in medical offices or hospital databases, and being susceptible to be shared manually, in a insecure way. When we consider the delicate essence of the information, this is clearly a problem. This scenario could leverage blockchain inherent transparency and immutable properties, from which a clear auditing trail, even from multiple sources, could be performed, ensuring the integrity and safety of the data itself. Estonia took a step on this regard, establishing a national Electronic Health Record, that uses a blockchain in order to ensure data integrity to healthcare data records [27]. Sweden is crawling towards the same goal, which is to develop a national blockchain for health records [28].
- ***Educational certification***: This is an area that also suffers from personal data records being held in isolated databases, now in the academic context. Faking degrees is a possibility, as the verification process can sometimes be a cumbersome task. Blockchain can be used to give personal data control back to individuals through verified records that could be used and verified without much effort. As a matter of fact, University of Nicosia (in Cyprus) already issues certificates that could be verified through a blockchain [29]. Also, the government of Malta, in cooperation with a startup is putting effort on building a prototype system to do the same as University of Nicosia [30].
- ***E-Voting***: The idea of having a easy and secure way of voting could, at least in theory, provide more participatory democracies. E-voting, however, had the downside of people arguing that it could be easier to manipulate digital system, than the actual paper-based for vote counting. Election systems can leverage the use of blockchain as an alternative to deliver such trust in a transparent way. Projects related with blockchain-based e-voting are being develop in areas such as West Virginia [31] and Moscow [32]. E-voting is also mentioned as a possible use case in the European Parliament's Blockchain Resolution of 3 October [33].

2. **Monitoring and market regulation**: Governments, more specifically regulators, are dependent of data to understand how to regulate and monitor markets, in order to ensure that laws are being complied and to protect individuals. This data though, can be difficult to come by. Governments usually rely on companies to supply the required information and this can lead to problems such as fraud or inaccurate information. Blockchain can address such issues, because as it is a shared ledger, this can simplify data gathering through easy real-time report request from public institution. Also, having a shared ledger implies a shared data format, from which data aggregation can be easier and, the immutability properties assures the accuracy of the information at any given time [34].

3. **Improving transactions, processes and transparency in public and private-sector markets**: This is the case where blockchain shared ledgers and smart contracts can improve and simplify the interaction of governments with suppliers by making procurement decisions transparently. Project oversight can be improved by leveraging the use of smart contracts, holding funds in escrow on the chain and only paying out when contractors meet certain targets. Currently, the EU is facing a challenge regarding Value Added Tax (VAT) fraud [35] simply due to the lack of transparency and efficiency, from which one of the solutions is to tackle this issue with blockchain [36].

4. **Efficiency**: Blockchain can be used to increase efficiency and reduce costs in government scenarios. This has to do with its very own decentralised and distributed

nature, where the platform's users share the same infrastructure which implies sharing the cost of implementing and maintaining the system [37]. Also, the distributed system such as blockchain are robust, since data is shared across the network, the data is secured as long as there is one functional node. This removes the costs of expensive backup and recovery systems.

2.3.5 Blockchain and GDPR Tension

Blockchain is becoming more and more popular nowadays, due to the amount of applications and use cases that it can support. One can see this effect, by looking at the growth of interest on cryptocurrency, such as Bitcoin for example.

However, blockchain technologies being considered as a potential for usage in the European Union's digital market are facing issues regarding the EU GDPR. There are currently, some points of tension between blockchain and GDPR.

For example, in GDPR, there is the existence of an entity called data controller. This entity is a legal person and corresponds somehow to a central unit, to whom data subjects can address to enforce their rights. However, one of the prerogatives of blockchain is decentralisation and therefore, the delegation of a responsibility such as the data controller to a node can become burdensome.

Another tension comes from the Art. 16 and 17 of GDPR, to which it is expected the obligation of user's "Right to Rectification" and "Right to erasure/Right to be forgotten". In other words, to be GDPR compliant, one must ensure that data can be modified or erased. Being an append-only distributed ledger, it is not the nature of blockchain to perform those operations on inserted data, as they can not be altered.

Another example is that data stored in the blockchain is also of great concern when GDPR takes place. There is a debate whether public keys or transactional data qualify as personal data, specifically, if encrypted/hashed data is qualified as personal. It is clear that this doubt, represents the difficulty of saying whether that a data that was once personal is anonymized enough to meet the GDPR threshold of anonymization.

Finally, the GDPR principles of data minimisation and purpose limitation, stated in Art. 5, are difficult to be achieved in blockchain. This has to do with the nature of blockchain (similar to the Art. 16 and 17 tension), where the ledger is append-only and so, data can only continuously grow when new information is inserted. Besides, the data is now replicated to other nodes, which turns these principles, as said before, rather difficult to be achieved.

From [38], there is an interest argument that it is easier to deploy a blockchain GDPR-compliant, if it is a private and permissioned one. This is because in permissioned networks, the participants are known and then, it can be defined a contractual relationship in order to allocate appropriate responsibilities to the nodes and held them accountable. As we can see, this is a way of mitigating the problem with GDPR having the data controller.

And lastly, the most important thing in order to deploy a blockchain GDPR-compliant, is to ensure that no PII is inserted into the blockchain. In most cases, only a reference/metadata to the personal data is appended to the ledger. With this, there is no need to worry about the tension from GDPR Art. 16 and 17.

2.4 Reputation Systems

As the world becomes increasingly interconnected, the need for reputation systems can only grow in importance. The main goal is to facilitate trust between entities who might have never interacted with each other before [39]. Today's usage of social networks and e-commerce is a good example of such need.

The term reputation has several definitions. According to the Concise Oxford dictionary: "*Reputation is what is generally said or believed about a person's or thing's character or standing*". Mui et al [40], on the other hand, define reputation as: "*the perception that an agent creates through past actions about its intentions and norms*".

Reputation systems work by collecting, aggregating and distributing data about an entity, that can be later used to predict that entity's future behavior. The idea is that by referring to the reputation data, users can take better decisions about whom and to what extent they will trust.

In fact, reputation systems also encourage good behavior, by the constantly evaluation through positive and negative rating scores. Positive rating could enhance the good performance, while negative rating may cause the entity to reflect on getting better at their job.

Reputation is also used to describe a group or an individual. A group's reputation can be computed, for example, taking into account the average of all members individual reputations, or by how they are perceived as a whole by the exterior [41].

It was also shown [42] that an individual inherits an a priori reputation based on the group reputation that it belongs. If the group is perceived as reputable, all its individual members will a priori be perceived as reputable as well and vice versa.

2.4.1 Trust and Risk

Reputation and trust can be confusing [43], as they are often used as synonyms, although their meanings are completely different. The definition of trust can be interpreted as one of the following concepts: reliability trust and decision trust [41].

Reliability trust includes the concept of dependence, as its definition [44] shows: "*Trust is the subjective probability by which an individual A, expects that another individual B, performs a given action on which its welfare depends*".

Relying only in the previous interpretation to define trust is proven to be more complex, as [45] having a high reliability trust is not enough to enter in such dependence situation. For example, a transaction where the risk is too high can cause such scenarios.

Decision trust *is the extent to which a given party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible* [41]. This definition has the intention of covering a more general approach to trust. Elements such as dependence, reliability and even risk are implicitly and explicitly mentioned.

The main lesson to be learned is that key elements of trust, such as dependence and reliability, can be measured through a entity's reputation. This extends the mentioned notion that reputation is a trust facilitator, by exploring trust parameters.

Risk can be seen as a situation where the outcome is important to someone, but the odds

of failure are non-zero. If we take into account the previous concepts of reputation and trust, they can be related as: the more risks a person is willing to tolerate, is proportional to the amount of trust embedded into the other person.

2.4.2 Reputation Models

In the literature [39], it is presented a taxonomy for academic and commercial reputation systems. To achieve the goal of indexing reputation systems in a consistent way, two reference models were developed. Figure 2.4 depicts a reference model for reputation context. This model has the goal of presenting the different contexts in which a reputation could be retrieved.

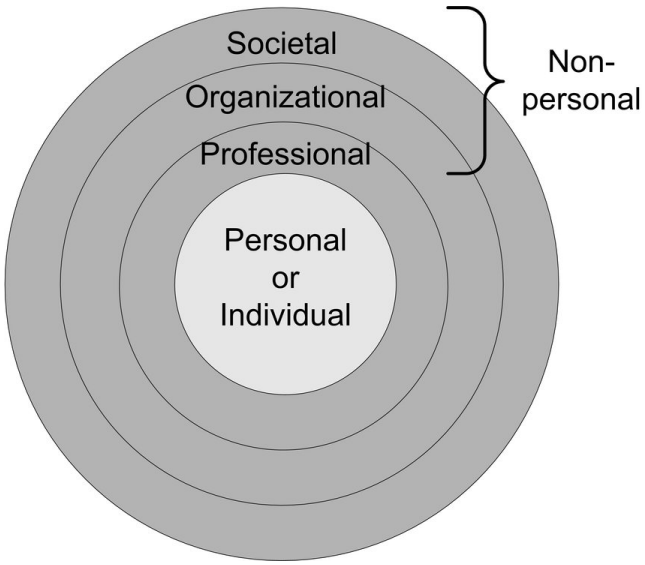

Figure 2.4: Reference model for reputation context (from [39]).

As we use additional information from different contexts (defined as *Contextual Attributes* in [39]), more accurate is the reputation score. The model, starting from the innermost ring, represents the various contexts that goes from personal (who), professional (what), organizational (which) and societal (where).

The second reputation model, presented in Figure 2.5, aims to generalize the model for reputation systems approaches. The idea is to present and define the parties involved and their interactions.

The entities presented in the model are the trustor, the trustee and the recommender. The trustor is the entity in charge of deciding whether to trust the party called the trustee. To make the decision, the trustor relies on the trustee reputation [46].

It retrieves the trustee reputation by first looking into his own internal reputation information and seeing if it has previously interacted with the target entity. His own base of information should be the prior source of reputation, since there is no better person to describe the trustee reputation than itself.

However, there are often cases where no interaction has occurred. In such scenarios, the trustor will then query one-to-many (1...n) recommenders, that may have interacted with the trustee before.

The recommender can provide information by: 1) Its own history of transactions; 2)

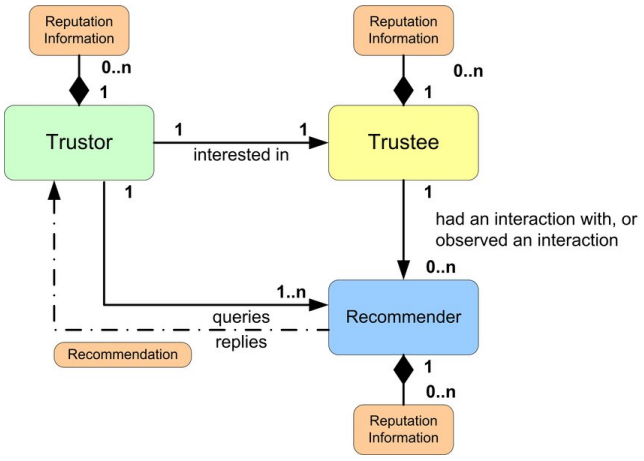

Figure 2.5: Reference model for reputation systems (from [39])

Observing the interactions between two parties; 3) Collecting information from other sources. Then, if any recommender has an adequate information about the trustee in question, it replies to the trustor with the recommendation.

Using the economic theory jargon, the first hand information (the one from the own trustor reputation base) is called private information, whereas information obtained from third parties are often called public information.

Hence, alongside with the recommender, the trustor is able to gather relevant data about the reputation of the trustee and then be able to decide. It is important to note that the roles in this reference model are interchangeable, since after the transaction takes place, both parties will update their own reputation information and may share with other parties.

2.4.3 Aggregation methods

The way in which the reputation score is computed is called aggregation [39]. Other sources in the literature, such as [41], name this as reputation computation engines.

Reputation systems, in majority, are designed to handle public information as input, in order to reflect the community's point of view from a certain party. There are system though, that takes both private and public information as input, where the private often is considered more reliable than the public ones. The aggregation methods are categorized further.

Counting

Considered the simplest form of aggregation, the reputation is computed by simply summing positive and negative ratings separately and returning the reputation score as the positive minus negative ratings. This is the principle applied in eBay reputation forum [47], for instance.

A slightly advanced approach of using this aggregation method is to compute the reputation score as the average of all ratings. This sort of principle is used in reputation systems of Amazon and Epinions.

Something even fancier is to consider weighted factors in order to compute the reputation score, those factors being, for example, rater own reputation or the “freshness" of the rating.

Discrete

In this approach the reputation is considered as a set of multiple discrete values, opposed to continuous. For example, the model presented in [48] could judge an entity as either “Very Trustworthy, Trustworthy, Untrustworthy and Very Untrustworthy".

Humans perceive reputation better when using this method, but practically speaking, it adds an extra overhead of computation, since values must be converted back and forth, which is not optimal.

Probabilistic

Probabilistic aggregation infers the use of a probability model to compute the likelihood of a certain hypothesis being correct. Often it uses prior knowledge (a priori) about the entity to predict its future actions (a posteriori).

One of the advantages of using such aggregation method is that it provides theoretically sound models in order to compute reputation. Also, different types of probability models can be applied, such as the bayesian reputation model approach.

Fuzzy

Aggregation methods using Fuzzy logic fall into this category. In this case, fuzzy logic is used to provide rules for reasoning and membership functions describe the degree of trustworthiness of an entity.

There are reputation schemes, such as [49] and [50] that uses this type of aggregation. The latest, refers to individual, social and context dependent reputation to describe, respectively, private, public and contextual information.

Flow

Flow aggregation method is related to the use of transitive trust interaction by network loops or arbitrary long chains, in order to compute the reputation. In simple words, this means that an entity reputation is calculated, taking into account the overall network opinion and their respective reputation themselves.

In some flow models, a constant reputation factor is assigned and distributed to the whole network. Hence, the increase of reputation comes at the cost of others. Google’s PageRank [51] fall into this category, by which the function of incoming flow (increase of reputation) is translated into hyperlinks pointing to the webpage, and the function of outgoing flow (decrease of reputation), is translated into hyperlinks pointed from the webpage.

2.4.4 Existing solutions

The eBay is an online e-commerce website that allows buyers and sellers to trade goods. At the end of each transaction, the parties have to leave each other a feedback, given in the form of a single rating (1, 0, -1). eBay's reputation system computes the overall reputation of each party by simply calculating the amount of positive feedbacks minus the amount of negative feedbacks.

REGRET [50] defines three different dimensions of reputation: individual, social and ontological. The individual dimension reputation is gathered through the direct interaction between the parties. The social dimension allows for the retrieval of reputation information from the target's belonging groups and communities. The ontological dimension enables a more complex evaluation of reputation, by considering different aspects of a interaction. For example, in order to calculate the reputation of a good seller, one can use the ontological dimension and consider delivery date, product price and product quality as the aspects to be evaluated. Each aspect uses the social and individual dimensions and will be combined assigning different weights for each aspect. Therefore, REGRET applies weighted average to compute the reputation.

FIRE [52] is an extended version of REGRET as it uses the same model for assessing an entity's reputation. FIRE incorporates a referral reputation for each entity, built on top of a witness reputation system.

CORE [53] uses a reputation mechanism in order to enforce node cooperation in MANETs and prevent the appearance of selfish nodes. The reputation mechanism uses the weighted mean to calculate the so-called "Subjective Reputation" and gives more relevance to past observations.

Sporas [54] can be seen as an evolved version of the online marketplace reputation models. Sporas provides a reputation service with the following characteristics: (1) Only the most recent submitted rating between two users is kept by the system, if those users happen to interact more than once; (2) Users with very high rating suffers much lesser rating changes and (3) the algorithm adapts to changes in user's behaviours, by attributing more weight for most recent rating submissions. The overall reputation of a user is calculated through Equation 2.1, which falls into a counting aggregation method category.

$$R_i = R_{i-1} + \frac{1}{\theta}\Phi(R_{i-1})R_i^{other}(W_i - E_i) \tag{2.1}$$

Histos [54] proposes a solution to the lack of personalization of Sporas. Histos considers a pairwise rating system represented as a directed graph, where nodes represent the users and the weighted edges represent the most recent reputation rating submitted from a user to another. The reputation of a user at a certain level X of the graph (with X > 0) is calculated recursively as a weighted mean of the ratings given by the users in the level X - 1 of the graph.

P-Grid [55] is a P2P distributed access network for information management. P-Grid is complete decentralized, self-organized that implements an underlying virtual binary search tree for replica distribution among peers. Its entities are considered trustworthy and only bad behaviours are considered. The nodes of P-Grid can forward complaints about a transaction, in the form of distributed messages, to other nodes. The trust assessment implemented in P-Grid is binary, being a node trustworthy or not. Equation 2.2 is used whenever a node i wants to evaluate the trustworthiness of a target j, where Cr and Cf denotes complaints received and filed, respectively. Cr_x^{norm} and Cf_x^{norm} represents the

normalized number of complaints received and filed by the available witnesses x. If the equation evaluates true, then the target j can be trusted, otherwise it is not.

$$Cr_x^{norm} Cf_x^{norm} <= (\frac{1}{2} + \frac{4}{\sqrt{Cr_i^{avg} Cf_i^{avg}}})^2 Cr_i^{avg} Cf_i^{avg} \tag{2.2}$$

XRep [56] is a reputation system extension for P2P platforms. Whereas the vast majority of reputation systems for P2P associate a reputation to nodes, XRep allows the management of reputation for both servents (nodes) and resources. Each node store its personal history for nodes and resources, accounting the number of successful and unsuccessful downloads for the nodes and a binary rating for the resources (trustworthy or not). Prior to downloading a resource, each node first contact its peers for advice. After selecting the appropriate resource, the node evaluates potential nodes offering that resource and selects based on its reputation.

RATEWeb [57] is a reputation framework intended for service-oriented environments. RATEWeb uses a cooperative model where web services share their experiences about service providers, providing a feedback rating that are further aggregated in order to arise with a service provider's reputation. The reputation of a service s_j is computed according to Equation 2.3, where L refers to the number of consumers that have interacted with the service s_j. The personal evaluation, $PerEval_j^x$, represents the consumer x's perception of the provider s_j's reputation. Finally, $C_r(x)$ represents the credibility of a service rater x as seen by the service consumer, ranging from [0, 1] with 0 being a dishonest rater and 1 a honest rater.

$$Reputation(s_j) = \frac{\sum_{x=1}^{L} (PerEval_j^x * C_r(x))}{\sum_{x=1}^{L} C_r(x)} \tag{2.3}$$

EigenTrust [58] is a P2P reputation system for entity trust assessment. Each node maintains a local trust value for other nodes, which is just the sum of positive and negative interactions. Those values are then normalized between 0 and 1, where negative local trust values are replaced by 0, to obtain a Markov chain. The global trust value is computed using Equation 2.4. T_{ik} represents the trust that the node i places in node k after the aggregation of all local trust values. c_{ij} is used not only as the local trust value seen by node i of node j, but also as the weight assigned to assess the global trust. Converting the equation to a matrix notation and conducting a series of observations, will lead that the global trust values converge to the left principal eigenvector of the matrix C, for every node i.

$$T_{ik} = \sum_j c_{ij} c_{jk} \tag{2.4}$$

Absolute EigenTrust [59] augments EigenTrust, providing a solution to its major drawbacks. As aforementioned, during the normalization process, negative trust values are replaced by 0, which does not distinguish neutral users from bad users. Also, most of flow models represents the reputations in the form of ranking, while Absolute EigenTrust provides an absolute value.

Peertrust [60] is a trust supporting framework for P2P networks. It is very similar to EigenTrust, although it incorporates five important parameters to compute a general trust metric for assessing a peer's trustworthiness. PeerTrust provides interaction context

information while computing reputation, taking into account factors such as the number of transactions, number of shared files and value of the transaction.

Beta [61] reputation system uses a beta probability density function (PDF) to represent the posteriori probabilities of binary events. The beta distribution is defined by two parameters α and β. Suppose an event with two possible outcomes $\{x,y\}$, and let r and s be the number of observations of outcome x and y, respectively. Then, the values of α and β are:

$$\alpha = r + 1 \quad \text{and} \quad \beta = s + 1, \quad \text{where} \quad r, s >= 0 \tag{2.5}$$

The beta distribution function $f(p|\alpha\beta)$ can be expressed as:

$$f(p|\alpha\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1}(1-p)^{\beta-1}, \quad \text{where} \quad 0 <= p <= 1, \quad \alpha > 0, \beta > 0 \tag{2.6}$$

Then, the probability expectation value is given by:

$$E(p) = \frac{\alpha}{\alpha+\beta} \tag{2.7}$$

Along with this probabilistic way of computing the reputation, the Beta reputation system allows for the aging of ratings (forgetting factor) which gives less weight to old feedbacks than recent ones.

Travos [62] is another reputation system based on the beta PDF. Trust computation is performed based on previous interactions between agents directly and indirectly. For the latter, Travos provides a mechanism for handling possible misleading recommendations. Also, Travos emphasizes the distinction between functional and referral reputations.

Dirichlet [63] is a reputation system designed to extend the beta reputation system capabilities through the adoption of the Dirichlet distribution. The whole idea is to overcome the weakness of assessing only binary event outcomes (x and y, as previously stated) to allow the possibility of more fine-grained ratings (x, y, z, for instance).

Subjective Logic [64] extends the notion of probabilistic logic, by adding a uncertainty factor to it. In subjective logic, ratings are described as a tuple (b, d, u), where b stands for belief, d for disbelief and u for uncertainty, such that $b + d + u = 1$. Belief corresponds to the probability of a statement being true, disbelief the probability of being false and uncertainty for being unknown. Subjective logic incorporates aging of ratings as the Beta reputation system does.

CertainTrust [65] is another probabilistic reputation system based on subjective logic. It also incorporates the notion of uncertainty, however, subjective logic uses belief and disbelief to compute uncertainty, while CertainTrust evaluate it based on the amount of information available.

Hedaquin [66] is a reputation system for assessing the reliability of healthcare information, built on top of Beta and CertainTrust reputation systems. In Hedaquin, each there are two types of ratings and reputations, namely functional and referral. Ratings are extended with a certainty factor that takes into consideration the quality and amount of information. Reputations are evaluated the same way Beta does, but certainty is used to weight each

rating. Hedaquin allows the aging of rating by incorporating the timestamps in each rating.

Fuzzy Trust [67] uses fuzzy concepts to calculate the reputation of users. Furthermore, membership functions are employed to represent the degree of a fuzzy variable, under a set of possible reputations (e.g. bad, neutral or good). In Fuzzy Trust, each rating consists of a value and a timestamp that is used to assess reputation, through trust transitivity and to give recommendations, therefore, enabling referral trust.

2.5 Summary

This chapter presented an overview about the PoSeID-on project, detailing its context and architectural components. Following, GDPR was introduced, giving its historical background, key terms and definitions, principles, privacy rights and their applicability. Blockchain was also presented by means of its definition, architecture, smart contracts, use cases and exploration of its tension with GDPR.

Furthermore, the state-of-the art on reputation systems, describing core concepts such as trust and risk, as well as generic models for reputation assessment. Also, the main aggregation methods were explained, along with a description of existing reputation systems found on the literature.

It is expected, with this chapter, to provide the reader with a theoretical sound basis for understanding the context of the following chapters.

Chapter 3

Contributions to PoSeID-on

This chapter presents the overall contributions of this thesis to the PoSeID-on project. Section 3.1 will present an overview and current status of the RMM. The architectural contributions of the author will also be discussed. Furthermore, in Sections 3.2 and 3.3, it will be presented a synthesis of the development efforts accomplished by the author, in the batch and service layer's of the RMM, respectively.

3.1 RMM Overview

The Risk Management Module is in charge of monitoring the PoSeID-on platform, in order to detect and evaluate privacy and security risks from a system-wide and individual data subject perspective.

One example of risk can be a data processor that suddenly requests much more information than usual from the data subjects, indicating that the data processor has been compromised. In such cases, the RMM must act by detecting and notifying the data subject(s) involved as well as the system administrator.

The RMM must also assign a reputation score to data processors, as to evaluate their behavior within the platform. This enables the dashboard to advise about which data processor can be trusted or not.

To accomplish those tasks, the module resorts on machine learning techniques and real-time data analysis frameworks. The source of information comes in the form of system logs, provided by each module of the platform. PII transaction and management operations metadata are incorporated in the system logs, whenever explicit consent is given by the data subject.

The result of the analysis is the identification of a set of data processors and data subjects involved in an anomalous pattern of logs. The identified set of data processors will have their reputations updated accordingly and the data subjects will be notified through a warning message, advising which services should be disabled.

3.1.1 Architecture

The primary goal of the architecture design was to allow for an efficient analysis of system logs and dispatch of results. Therefore, the architecture of the RMM is based on the so-

called *Lambda Architecture* [68], which is a proposal made by Nathan Marz for a generic, scalable, and fault-tolerant data processing architecture.

The adoption of the lambda architecture enables for efficient processing of a large amount of information, like the ones expected to be generated by the PoSeID-on platform. Furthermore, it allows for a near-real-time data analysis, providing the data subject with fast feedback concerning security risks. The response time can be understood as within a user's session on the platform.

In parallel, the lambda architecture provides the extraction of valuable insights from large volumes of information, using more complex and resource-intensive methods. It does so, by dividing the architecture into three different layers: a speed layer, a batch layer, and a service layer. The architecture of the Risk Management Module is depicted in Figure 3.1.

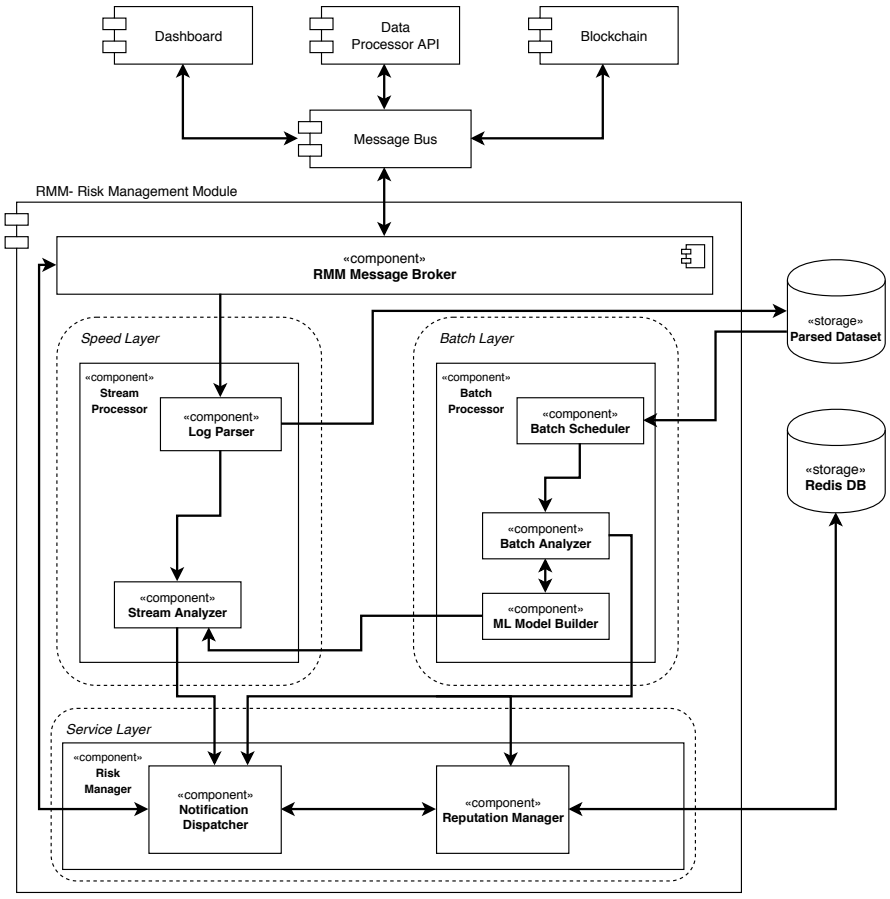

Figure 3.1: Architecture of the Risk Management Module.

As the system logs arrive at the Risk Management Module, they are forwarded directly to the speed layer into the Log Parser component. After that, each parsed log is stored in the module's external database and sent down-stream for the anomaly detection pipeline.

The batch layer fetches the external database for the parsed logs and uses them for historic risk analysis, being the period depends on how long the RMM is allowed to retain data, with the consent given by the data subject. This historic risk is performed in a more in-depth analysis and provides warnings in case of risk detection in an "offline" manner.

Having a batch layer also provides the advantage of being able to use machine learning models that do not have a stream-based counterpart. This layer is also in charge of training and updating machine learning models for use in a near real-time analysis by the

speed layer, for algorithms that require an offline training step, as is the case with most supervised learning algorithms.

The speed layer trains the models on the go, leveraging machine learning libraries provided by the framework. Also, the speed layer deploys the models created by the batch layer whenever they are trained. It analyses the stream of data in real-time, dividing it into small batches of data provided by the message queue. Whenever an anomaly is detected, both the stream and batch layers dispatch a warning to the service layer.

The service layer is in charge of notifying the proper entities about security issues through the web-dashboard. It receives the results from both the speed and batch layers and, if anomalous behavior is caught, the warning message is issued.

It is also in charge of receiving feedback from administrators regarding such risk notifications, by providing the identification of the log window where the anomaly was identified and allowing them to confirm or deny that real risk is present within it.

Lastly, the service layer has the responsibility of calculating and storing the reputation of each data processor within the platform. The reputations are updated according to the administrator's feedback and the history of anomalous logs involving each data processor.

Most of the architectural design remained the same, prior to the author's work. However, the inclusion of Redis as another storage and the exclusion of the "Monitoring Module" subcomponent can be cited as the contributions of the author, to the architecture.

3.1.2 Dependencies, Frameworks and Storages

The newest version of the RMM was build on top of a collection of libraries and frameworks in order to speed up the development time. Most of them are well-known, such as Apache Spark for big data processing.

It is important to remark that the module was developed in Java 8. This choice was driven by the used data processing framework, since it is the only supported version for Java. Other languages could be used as well, such as Scala and Python, however because it is a compiled language and also, because of personal preference, Java was the selected one.

In the following subsections, the dependencies and frameworks will be presented in the same structure of the POM configuration file: plugins first, repositories second and libraries third. Lastly, we will include the description of the external storages used by RMM.

Plugins

There are two plugins adopted by the RMM: Maven Compiler and Maven Assembly. The maven compiler is used to compile the sources of the project, while the maven assembly creates the uber-jar for the RMM application and its dependencies.

Repository

The only repository appended to the module is the Lazysodium Java Repository. This repository is needed to include the library for Lazysodium, that provides cryptographic operations.

Libraries

The collection of libraries used for the RMM's implementation is presented in Table 3.1. It includes a multitude of tools, needed for unit testing, logging, messaging, data processing and reputation management.

Library Name	Version
junit-jupiter-engine	5.4.2
junit-jupiter-api	5.4.2
log4j-core	2.11.2
log4j-api	2.11.2
logstash-gelf	1.13.0
protobuf-java	3.8.0
lazysodium-java	3.6.0
amqp-client	3.3.4
spark-sql_2.11	2.4.0

Library Name	Version
spark-streaming_2.11	2.4.0
spark-rabbitmq	0.6.0-SNAPSHOT
akka-actor_2.11	2.5.3
spark-mllib_2.11	2.4.0
lombok	1.18.6
jackson-dataformat-yaml	2.9.8
commons-math3	3.6.1
jedis	3.2.0

Table 3.1: RMM adopted libraries

There are two libraries that could not be imported directly into the POM of the RMM project: Libsodium and RabbitMQ Spark Streaming Receiver. The first one was added as a resource to the project, in order to be used by the Lazysodium Java. The second one is a connector between the Spark Streaming framework and RabbitMQ message bus. Although is has not been updated in two years, there are currently no other implementation of such connector. The compromise is to have its jar installed into the local maven repository, in order to be imported and compiled with the RMM.

Storages

The external storages denoted in Figure 3.1 by "Parsed Dataset" and "Redis DB" are implemented using Apache Cassandra [69] and Redis [70], respectively, and they are both deployed within the platform scope. The former is used only for the RMM and is responsible for the keeping the records of parsed logs, log templates, feature vectors and analysis results. As for the latter, it can be used by any module inside the platform and includes the records for the reputations of every data processor.

Apache Cassandra is an open-source distributed NoSQL database, that provides continuous availability, high performance and scalability. The design decision of using Cassandra comes from its alignment with the need for faster write speeds and the distributed deployment strategy of PoSeID-on.

Redis is also an open-source data structure storage, that can be used as a database. The outstanding performance achieved from Redis is due to the fact that it works as an in-memory data structure. Prior to the implementation of the RMM, the Redis was already deployed for communication between other components of the platform. Therefore, the decision of using it for reputation management was naturally made, leveraging its already deployment.

3.2 Batch Layer

As aforementioned, the batch layer works in order to create a better representation of machine learning models, hence improving the accuracy of the predictions made. The batch layer takes into account a large amount of data and could leverage machine learning techniques that does not have a stream-based counterpart. The subcomponents of the batch layer are: Batch Scheduler, Batch Analyser and Machine Learning (ML) Model Builder.

It is important to state that the contribution of the author, to the batch layer, was merely from a implementation level. The design of the architecture and its subcomponents was already made prior to the author's work.

Also, in its current version, the batch layer was not able to be validated, due to issues with Apache Spark garbage collection that reflects into memory leaks, causing the batch layer to not complete the anomaly detection pipeline. This will be further fixed, as the author will be enrolled into the project even after the thesis completion.

Therefore, in the following subsections, only the approach taken for the implementation of the batch layer subcomponents will be explained.

3.2.1 Batch Scheduler

The goal of the batch scheduler is to periodically fetch data that was once parsed by the stream layer. The parsed data is located in the Cassandra database, specifically in the parsed logs table, and the entire set of logs, from that batch run, is loaded into the batch processor environment and passed to the analyzer component.

The scheduling is performed using the scheduled executor service java interface. Its parameters are a runnable interface, an initial delay and the periodic time in which the function is called. In our case, the runnable interface is the batch processor instance, the delay is zero and the batch run period is set to a default of one day.

3.2.2 Batch Analyser

The batch analyser implements the feature extraction phase of the anomaly detection pipeline. The whole framework of the anomaly detection pipeline is illustrated in Figure 3.2. After the log parsing, a group of event templates (constant part) is extracted with some parameters (variable part). The batch analyser acts as a bridge between the raw logs information and the machine learning models used to detect suspicious events. The output of the batch analyser is a set of event count vectors or an event count matrix.

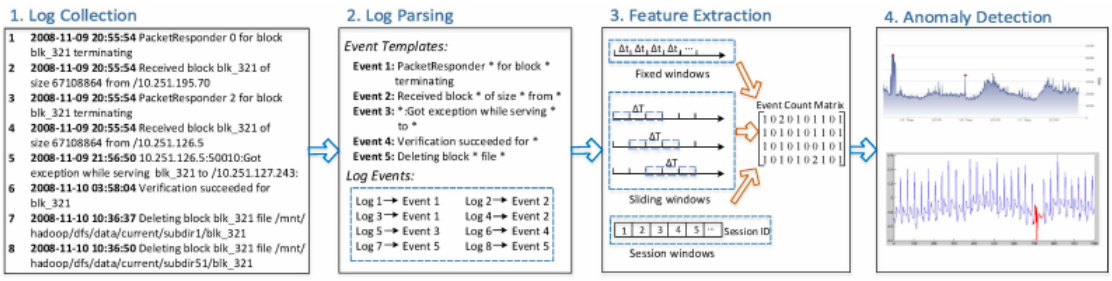

Figure 3.2: Framework for the anomaly detection pipeline (from [71]).

To achieve this goal, raw logs are split into a set of log events using some sort of grouping technique, such as fixed windows or sliding windows. Then, each log sequence is converted into a event count vector, which represents the number of occurrences of each event template in the log sequence. In the end, all the feature vectors can be combined into a event count matrix.

3.2.3 ML Model Builder

The machine learning model builder is a straightforward subcomponent that uses a predefined algorithm to predict whether a log is anomalous or not. For the PoSeID-on case, we will rely on unsupervised learning algorithms since we have unlabeled training data.

Apache spark provides a set of machine learning algorithms through its machine learning library, known as Apache MLlib. From the unsupervised algorithms available, the selected one was the K-means since it is a well-known and test clustering algorithm and also because it is used in the speed layer.

There are two parameters to be configured before using the K-Means: the number of clusters and the maximum number of iterations to run. From the tests conducted in the speed layer [72], the best results were achieved using a cluster of size 4 and 24 iterations at maximum. Hence, those are the values selected for the algorithm in the batch layer.

3.3 Service Layer

The service layer is responsible for the communication from the RMM to the message bus, with the additional responsibility of managing properly the reputation of the data processors. Following, the implementation of its two submodules will be described.

3.3.1 Notification Dispatcher

The notification dispatcher is in charge of handling the communication from the RMM to the web dashboard. Since all messages passes through the message queue, the notification dispatcher needs to ultimately interact with the RabbitMQ connector.

All the messages are signed, encrypted and encapsulated before being sent to the message queue. The structure of the message protocol was defined by JIBE. In order to send a warning message for a certain data subject, the RMM need to attribute the message to the queue "dashboard" with the data subject certificate as the recipient.

In the following subsections, the unit testing and validation of the notification dispatcher, performed by the author, will be presented.

Unit Testing

Tests were developed in order to verify the message protocol implementation as it can be seen in Figure 3.3. These tests comprise:

- Verification of signed payloads using correct and forged signatures
- Successful decryption of messages with the correct keys and failure otherwise

- Verification of signed and unsigned certificates

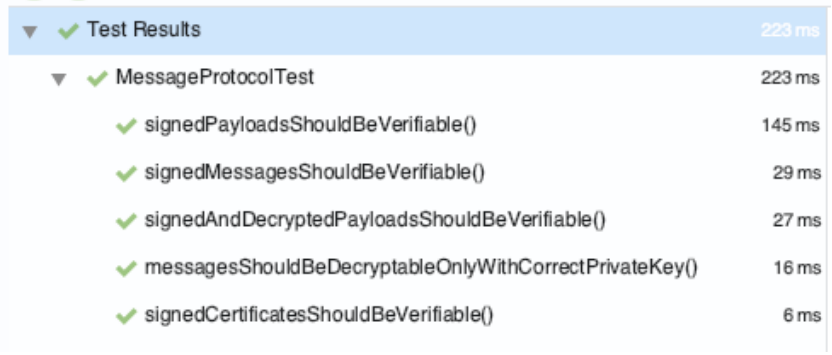

Figure 3.3: Unit test of the notification dispatcher.

We can state that the unit test was successfully performed, as not a single test yield a negative result.

Validation

The approach taken to validate the notification dispatcher was to send a message to a data subject, on the integrated environment of the PoSeID-on platform and verify if the notifications were received. As can be seen in Figure 3.4, the data subject received successfully a request permission from RMM, asking to grant access to be used in the detection of anomalies involving this particular data subject.

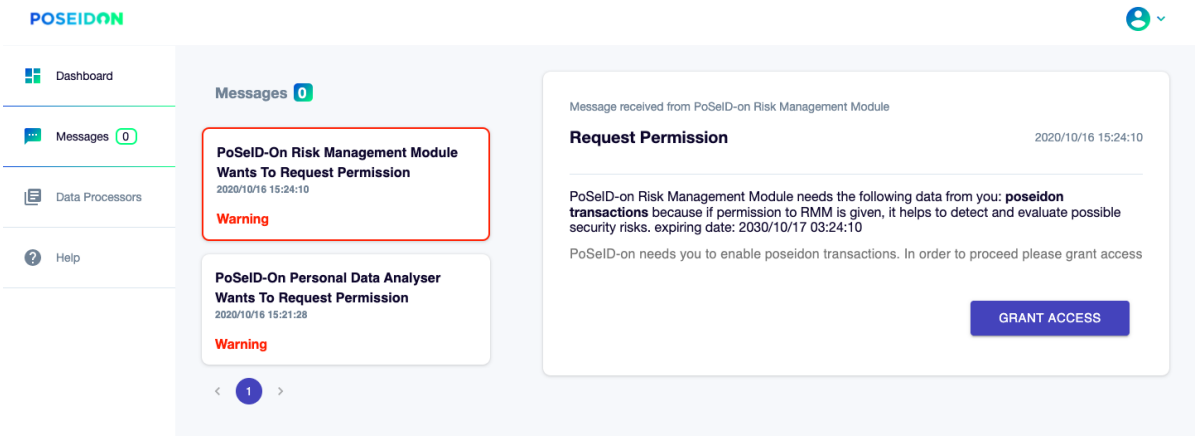

Figure 3.4: Validation of the notification dispatcher.

3.3.2 Reputation Manager

The reputation manager is responsible for ensuring the management and storage of data processor's reputation score. The management of reputations includes the CRUD operations. As for the storage, the two obvious operations of read and write were implemented.

The computation of the reputation relies on the beta distribution function for reasons that shall be explained later in Chapter 4. Therefore, it was included the library Apache Commons Math containing mathematics and statistics components that are not provided in the standard Java programming language libraries.

Moreover, the reputation management holds an FORGETTING_FACTOR (λ) variable in order to account more weight towards recent interactions behaviours. This variable can range from [0, 1], where 0 accounts only for the most recent interaction and 1 accounts all the interactions seen so far.

Regarding the storage, since Redis will be used to record the reputations, we resort on the usage of the Jedis library which provides an abstraction of Redis client for Java.

In the following subsections, the unit testing and validation of the reputation manager, performed by the author, will be presented.

Unit Testing

Tests were developed in order to experiment functional aspects of reputation manager class as it can be seen in Figure 3.5. These tests comprise:

- Updating reputation with anomalies and non-anomalies as first inputs
- Retrieving reputation score for new data processors
- Initialization of reputation manager class with list

Figure 3.5: Unit test of the reputation manager.

We can observe that all of the tests created were successfully performed and achieved positive results.

Validation - Normalization weight

In Beta reputation system feedbacks are given as a pair (r, s) and a normalization weight, given by w, can be imposed in such way that $r + s = w$. In that sense, the value of the transaction can be translated according to its weight (e.g. higher the transaction value, higher the weight).

Also, in order to ease the feedback given by the users, instead of using a pair (r, s), we can define a new variable v, such that v $\in$ [-1, +1]. Hence, the pair (r, s) can be derived according to Equation 3.1.

$$r = wv \quad s = w(1 - v) \tag{3.1}$$

The validation of the normalization was carried out by assessing the evolution of the reputation rating after a consecutive number of positive feedbacks (n), while varying the value of w. So, the variable v will remain 1 and aging is not considered.

We can see, in Figure 3.6, that the reputation rating grows quickly whenever the normalization weight approaches to 1, whereas the reputation value do not change if the w equals to 0.

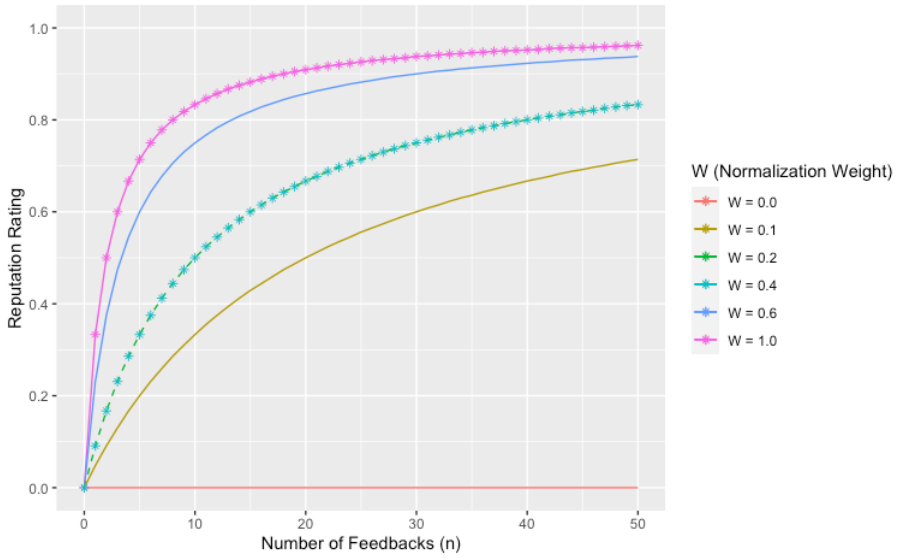

Figure 3.6: Validation of the reputation manager with respect to the normalization weight.

Also, when we compare the results achieved to the original Beta reputation system paper [61], we state that they yield to the same conclusion, therefore, validating the implementation with regard to the normalization weight.

Validation - Forgetting factor

For the validation of the forgetting factor, we again assess it after a number of consecutive positive feedbacks while varying the value of forgetting factor variable. We've fixed the values of w and v to 1. The result is depicted in Figure 3.7

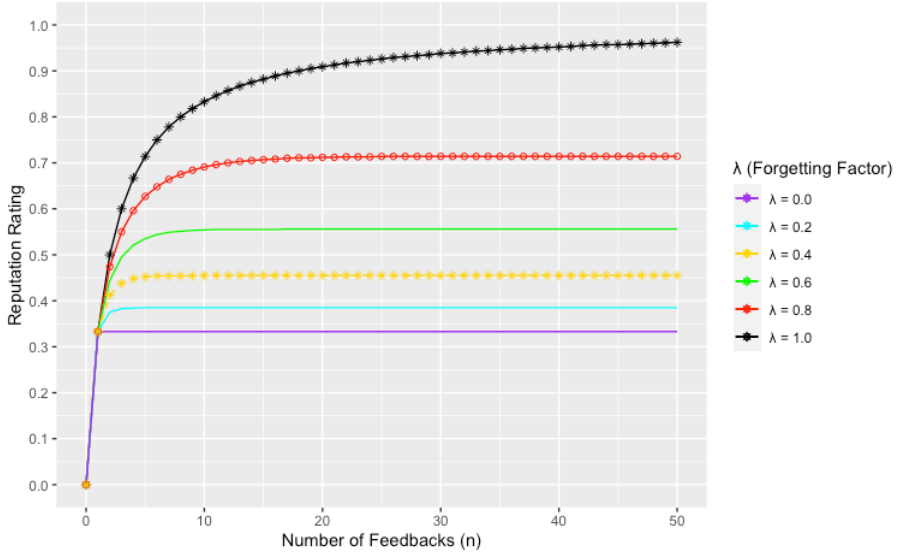

Figure 3.7: Validation of the reputation manager with respect to the forgetting factor.

As we can see, with λ equals 1, nothing is forgotten and when λ equals 0 only the last

feedback is remembered. Also, we have stated that the results match the ones presented in the original beta reputation system paper.

3.4 Summary

This chapter presented an overview about the RMM, highlighting its architecture, layers, components and interactions. Further, the technical development stack choices were introduced through the used plugins, repositories and libraries. The design decision of the storages which are external to the RMM were also explained along with their usage.

The batch and service layers and their respective subcomponents were detailed, providing the implementation approach followed by the author. Furthermore, the unit testing and validation performed by the author, of every subcomponent of the service layer was presented.

Chapter 4

Reputation System

This chapter will present the design decisions behind the reputation system developed for the Risk Management Module. It appeared as a requirement for the PoSeID-on project and therefore, needed to be carefully designed in order to accommodate the project's objectives.

First, a requirement analysis for the reputation system will be conducted in Section 4.1. Then, the proposed solution based on the requirement analysis will be presented in Section 4.2. Further, a set of possible scenarios will be depicted in Section 4.3. Finally, the results and discussion will be addressed in Section 4.4.

4.1 Objectives

Being of utmost importance, the reputation system needs to be properly defined in order to address the scenario in which the RMM is currently embedded. Defining the reputation system, as for this case, means defining the reference model of the system and the aggregation method to be used, as it was previously shown in subsection 2.4

Figure 4.1 shows the result of applying the reference model for reputation systems in the context of PoSeID-on. As it can be seen, the elements of such reference model are mapped as: data subject being the trustor, data processors being the trustees and the reputation manager being the recommender.

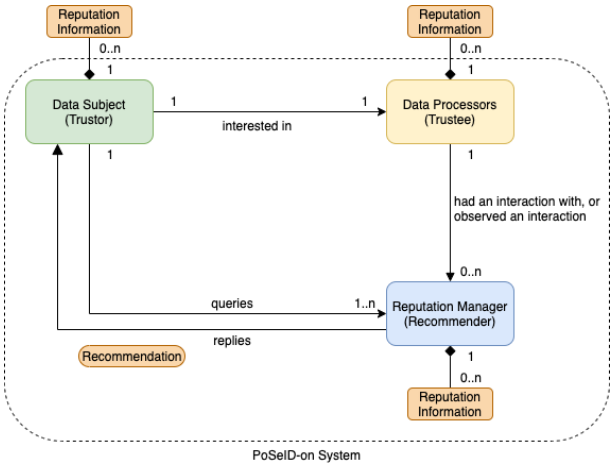

Figure 4.1: Reference model for reputation systems in PoSeID-on's context.

Also, it is important to notice that the PoSeID-on system only covers reputation information for the reputation manager. This means that both data subjects and data processors own base of reputation information is out of the scope of PoSeID-on. Adapting the reference model for the PoSeID-on's case gives a better picture of the overall trust interaction between users.

As for the aggregation method to be used, a literature study was conducted to investigate which method better suits the project. A specific study [73] on eliciting the requirements and features of reputation systems was conducted and a validation of those requirements by well-known reputation systems was made.

The approach to select the best aggregation method was to reason about the main features required by the project and compare with the current solutions that include those features in order to give a hint on which aggregation method to use and how it's been used. The features elicited are shown in Table 4.1.

ID	Feature	Description
F1	Trust/Distrust	Represents the ability to express the entire range of user behaviour.
F2	Absolute Rep. Values	Represents the possibility of expressing reputation value in an absolute way, instead of a ranking-fashion.
F3	Origin/Target	Represents the possibility of identifying the origin/target of ratings.
F4	(un)Certainty	Represent the ability of reputation metrics to specify the level of confidence in trust information.
F5	Interaction Scope	Indicate whether the reputation system is able to discriminate ratings and reputations according to the type of interaction.
F6	Scope Similarity	Denotes whether the degree of similarity between scopes is used to assess reputation.
F7	Trust Transitivity	Denotes the possibility to infer the indirect trust between two users due to their trust relationships with other users.
F8	Functional vs Referral	Represents the ability to distinguish between the trust on a user's ability to provide a service and the trust in a user's ability to provide recommendations.
F9	Interaction Context	Represents the ability of reputation metrics to discriminate interactions on the basis of their cost.
F10	Timestamp	Represents the ability to consider the timestamp of the interaction in the assessment of reputation.

Table 4.1: Reputation systems features elicited (from [73]).

In the previous table ten features were described, from which their applicability on PoSeID-on platform will be explained as follows. Envisioning those features into the project, one can think of feature F1 as being able to be expressed in terms of non-anomalies/anomalies. In other words, the behaviour of the data processor will be assessed according to its interactions being part of an anomaly detection or not.

Feature F2 can be accomplished simply by representing each data processor's reputation within the range of 0 (poor reputation) to 1 (perfect reputation). As for feature F3, the origin of the ratings will ever be the RMM itself, whereas the target of ratings needs to be looked up at the log event information.

Feature F4 could be partially handled by considering the fact that (un)certainty can be derived by the number of interactions so far (as more interactions happens, less uncertain it is the level of confidence on the information). Feature F5 could be investigated on how distinct interactions (e.g. data processor burst of requesting permission versus data

processor sending compromised PII) might weight differently towards a data processor reputation. However, due to project's deadline constraints, this feature was not explored.

Feature F6 falls into the same explanation given for feature F5. As for feature F7, it is inapplicable in PoSeID-on's context, since the RMM is the only entity in charge of rating the data processors and in that case, inferring the RMM trust does not apply. Feature F8 also does not concern the PoSeID-on platform, because there is not the possibility of rating recommendations given by a data processor.

Feature F9 can be given the same reasoning as features F5 and F6. Lastly, feature F10 can be partially incorporated by assigning more weight towards more recent interaction than older ones. In that case, it is not the timestamp itself, but the time of an interaction will matter to the assessment of reputation.

A summary of the features needed for the PoSeID-on project is presented in Table 4.2. The support column express the level of implementation expected to be accomplished by each feature. As we'll see in the next subsection, it will help in the choice of the proper reputation system to be developed.

ID	Feature	Support
F1	Trust/Distrust	Full Support
F2	Absolute Rep. Values	Full Support
F3	Origin/Target	Full Support
F4	(un)Certainty	Partial Support
F5	Interaction Scope	No Support
F6	Scope Similarity	No Support
F7	Trust Transitivity	No Support
F8	Functional vs Referral	No Support
F9	Interaction Context	No Support
F10	Timestamp	Full Support

Table 4.2: Reputation system features needed for PoSeID-on project.

4.2 Proposed Solution

With all features evaluated for PoSeID-on, an in-depth comparative study of current reputation systems solutions can be made with the help of Figure 4.2.

When comparing the features needed for the project with the provided ones from each reputation system, only a few with probabilistic metric assessment fulfill our needs, namely the Beta, Subjective Logic and Hedaquin reputation systems. No other aggregation method can provide those features, hence the selected one for the project will be the Probabilistic one.

As the majority of the probabilistic reputation systems inherits and extend features from the Beta reputation system, the approach to be followed into the reputation manager is to use the Beta reputation system as the basis for reputation scoring and extend it, whenever necessary.

However, as can also be seen in Figure 4.2, Beta reputation system is characterized for

	eBay [16]	REGRET [42]	FIRE [43]	CORE [44]	EigenTrust [20]	Absolute EigenTrust [22]	Peertrust [45]	Beta [26]	Travos [46]	Dirichlet [47]	Subjective logic [28, 29]	CertainTrust [48]	Hedaquin [27, 49]	Fuzzy Trust [51]	FuzzyTrust/eTrust [18]
Metric	Counting				Flow			Probabilistic						Fuzzy	
Structure	C	D	D	D	D	C	D	C	D	C	C	D	C	D	D
Measure	D	C	C	C	C	C	C	C	C	C	C	C	C	C	C
F1	✔	✔	✔	❊	❊	✔	❊	✔	✔	✔	✔	✔	✔	✔	✔
F2	✔	✔	✔	✔	✘	✔	✘	✔	✔	✔	✔	✔	✔	✔	✔
F3	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
F4	✘	❊	❊	✘	✘	✘	✘	❊	❊	✘	✔	✔	✔	✘	✘
F5	❊	✔	✔	✔	†	†	†	†	†	†	✔	✔	✔	✘	✘
F6	✘	❊	❊	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘
F7	✘	✔	✔	✔	✔	✔	✔	✔	✔	✘	✔	✔	✔	✔	✔
F8	✘	✘	✔	✘	✘	✘	❊	✔	✔	✘	✔	✔	✔	✔	✘
F9	✘	✔	✔	❊	✘	✘	✔	✘	✘	✘	✘	✘	✘	✔	✔
F10	✔	✔	✔	✔	✘	✘	✔	✔	✘	✔	✔	✘	✔	✔	✔

Legenda

Structure:
C: Centralized
D: Decentralized

Measure:
C: Continuous
D: Discrete

Features:
✔: Full support
†: Limited support
❊: Partial support
✘: No support

Figure 4.2: Reputation systems survey matching corresponding features (from [73]).

having a centralized structure and producing a continuous reputation measure output. The latter characteristic is in accordance with the PoSeID-on platform, as for example, the PDA expects reputation to be stored as a real value in the range of [0, 1].

The former characteristic, though, has to be further exploited, as it goes in the opposite direction of the project's evolution and its core technology (blockchain). The natural path of PoSeID-on is to allow for a distributed environment created under the scope of Blockchain and to be followed by its modules (RMM and PDA). Therefore, a centralized reputation scheme needs to be carefully thought to work in a distributed environment.

Another point to consider is the rather novelty of applying the Beta reputation system under a distributed approach. To the best of the author's knowledge there are only two papers [74, 75] that achieve the aforementioned. They differ from our solution, however, from a contextual perspective and the way in which the reputation is computed. Both of them employ the exchange of information between peers, in order to compute the overall reputation, which shall not be our case.

Ultimately, the benefits of having a distributed environment can be translated in terms of scalability (multiple RMM instances) and resilience. As the PoSeID-on platform is deployed under a container orchestrator (Kubernetes) and, the RMM is fully containerized, scaling up the number of instances is trivial and can improve availability, while reducing the work load.

4.3 Architectural Scenarios

There are several ways to implement a distributed environment that matches with the envisaged reputation system. However, not all of them are efficient nor they are effective. To illustrate it better, four different scenarios were designed and will be presented the

next subsections.

4.3.1 First Scenario - Baseline

The first scenario comprises the baseline case of the reputation system. In this case, we consider only one instance of the RMM and the overall pictureo f the architecture was already presented in Figure 3.1.

The RMM handles the reputation of all data processors in a centralized manner. This scenario will serve as a starting point that will be used as a reference for comparison with any other scenario. In Figure 4.3, is possible to observe the sequence diagram of the expected events concerning the usage of RMM.

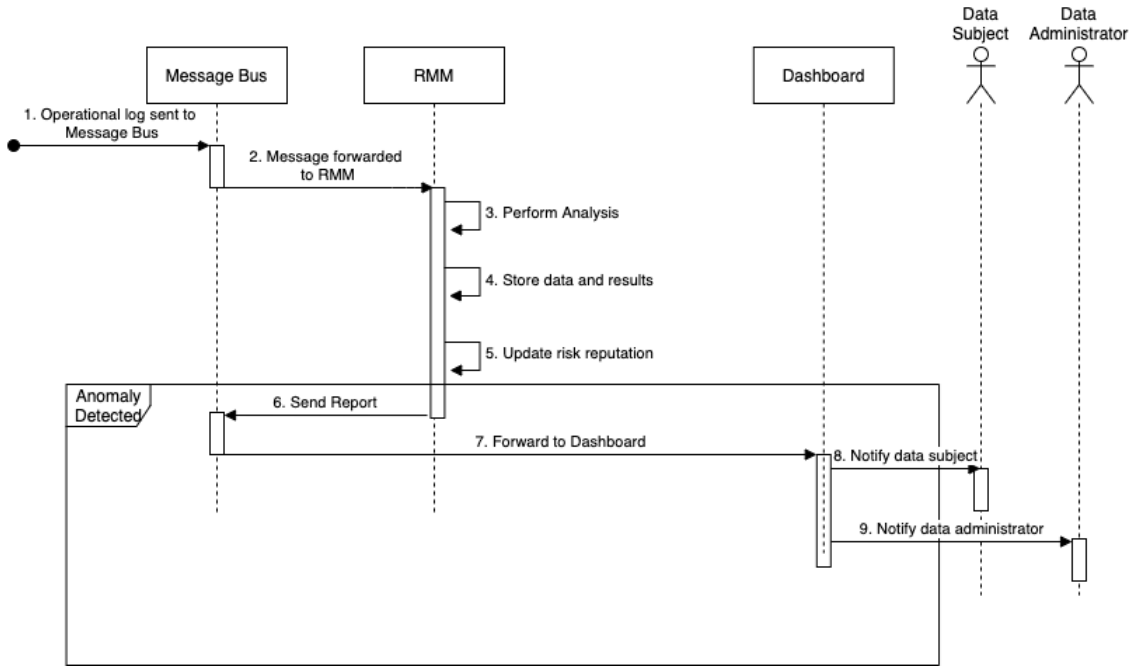

Figure 4.3: Sequence diagram envisioned for the baseline.

4.3.2 Second Scenario

The second scenario encompasses the usage of a new component, depicted in Figure 4.4, which corresponds to an orchestrator. The goal of the orchestrator is to handle all the requests from the message queue and forward them to the proper RMM instance.

The message handler is in charge of managing the communication with the message queue, while the data processor identifier extracts the data processor involved in the message received. Then, it passes the information to the load balancer which maintains a routing table, mapping each RMM instance to a specific data processor. The instance monitor keeps checking if the instances are still active and informs the load balancer, otherwise. The instances health are logged in a storage.

In this case, each RMM instance handles the reputation of its respective data processor and the overall scenario can be seen in Figure 4.5. Differently than the baseline, this scenario anticipates the usage of the orchestrator and allows for multiple RMM instances. Whenever a message arrives, the orchestrator forwards to the according RMM instance

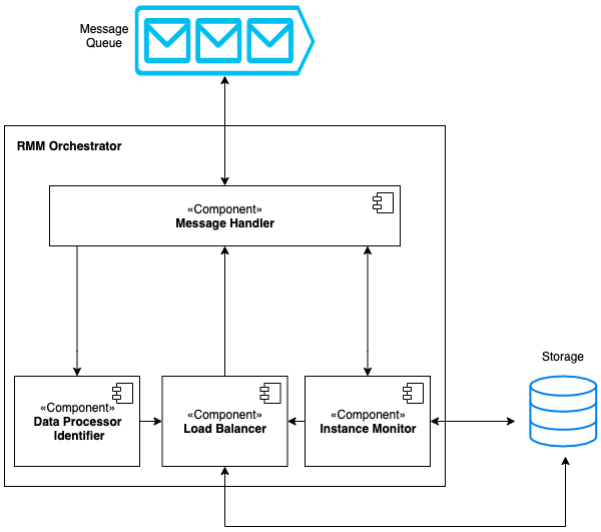

Figure 4.4: Orchestrator envisioned for the second scenario.

by sending the message back to the message queue, with the RMM instance as recipient.

The downside of this scenario, apart from having to implement the orchestrator itself, is that one message can be handled only by one RMM instance which loses the whole point of the distributed environment. Also, if there are several messages associated with a particular data processor, the same RMM instance will have to handle it, not leveraging the distributed property of reducing work load.

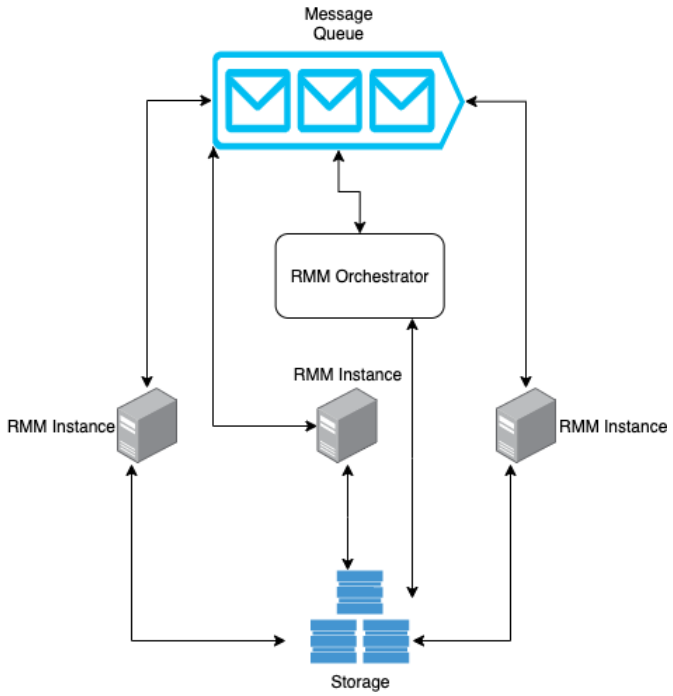

Figure 4.5: RMM architecture for the second scenario.

Lastly, for this scenario, we present the sequence diagram in Figure 4.6. From the baseline, we observe that the first set of events involves the orchestrator, which was now added as an actor.

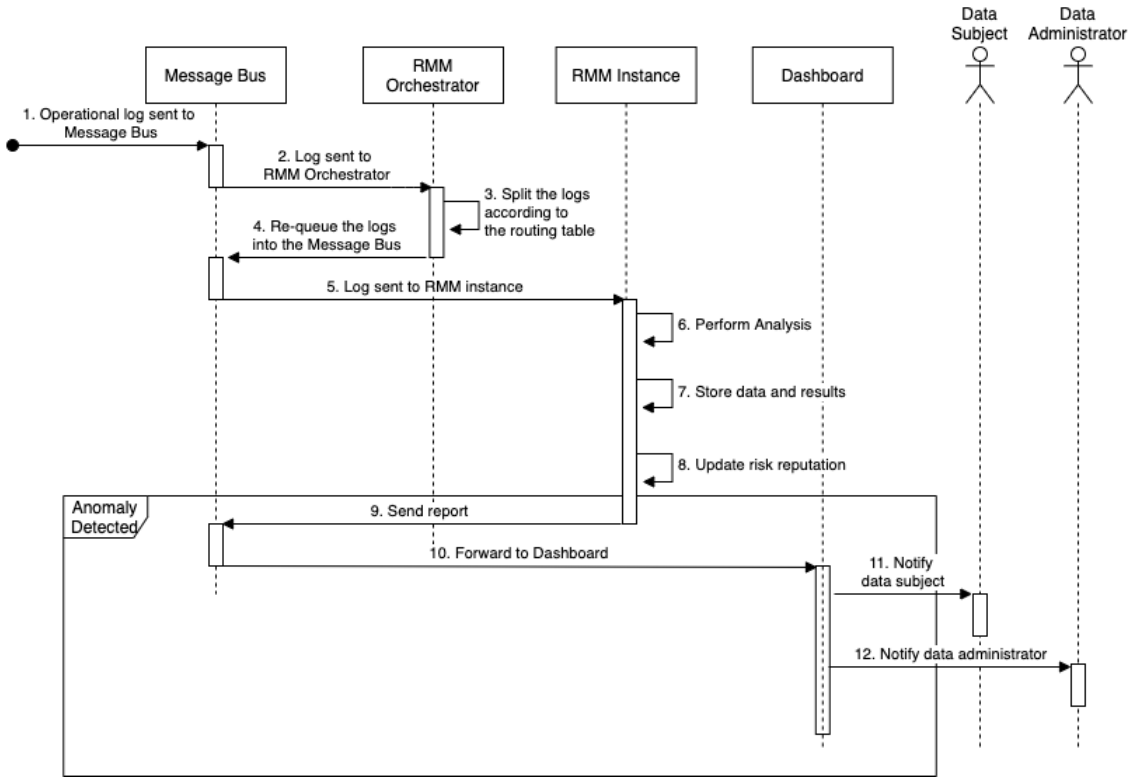

Figure 4.6: Sequence diagram envisioned for the second scenario.

4.3.3 Third Scenario

The third scenario also incorporates the usage of an orchestrator, as can be seen in Figure 4.7.

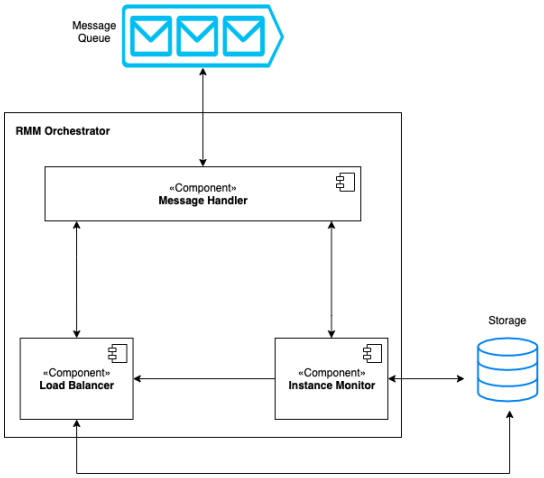

Figure 4.7: Orchestrator envisioned for the third scenario.

In order to solve the issues related with the second scenario, the data processor identifier subcomponent is no longer used, but rather, the orchestrator forwards the messages according to the work load. The load balancer does not need to maintain a routing table anymore and just has to keep the load of the instances evenly as possible. The overall architecture of the third scenario remains the same as the previous scenario.

In that sense, each RMM instance can handle the reputation of all data processors, but

yet, it depends on the orchestrator to handle all incoming and outcoming messages. In Figure 4.8, the sequence diagram for scenario three is presented, which only differs from the second scenario with respect to the load balancing.

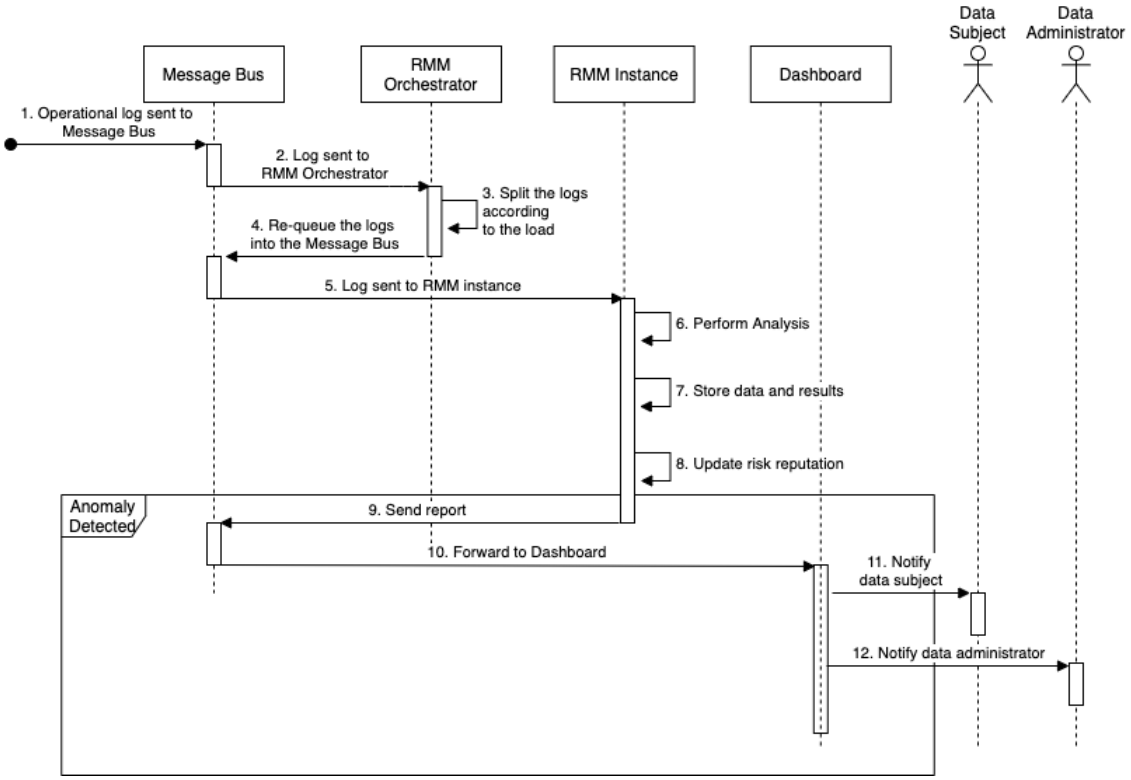

Figure 4.8: Sequence diagram envisioned for the third scenario.

4.3.4 Fourth Scenario

The fourth scenario addresses the complexity overhead of having an orchestrator deployed in the architecture, by simply discarding it and having every RMM instance directly connected to the message queue. This architecture can be seen in Figure 4.9.

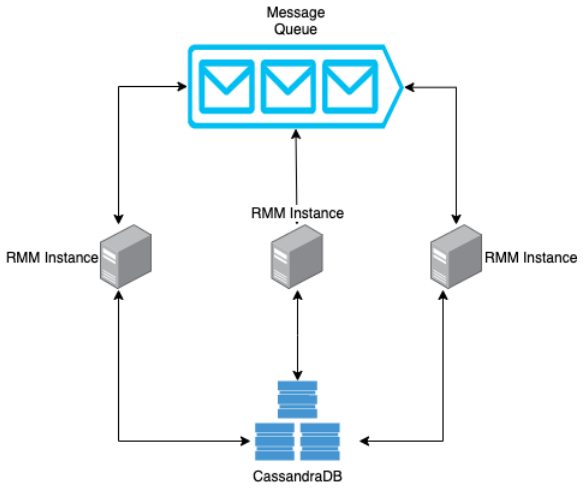

Figure 4.9: RMM architecture for the fourth scenario.

This scenario is the most distributed in nature than the others, as no other component

is present but the instances themselves. In this case, one message can be handled by any RMM instance, which in turn, can handle the reputation of all data processors from the platform.

The sequence diagram for the fourth scenario is depicted in Figure 4.10. It is similar from the sequence diagram of the baseline, but in this case, there is a "RMM instance #" actor, which represents any RMM instance from the architecture. Also, it is expected that each instance retrieves the messsages from the queue in a first come, first served fashion.

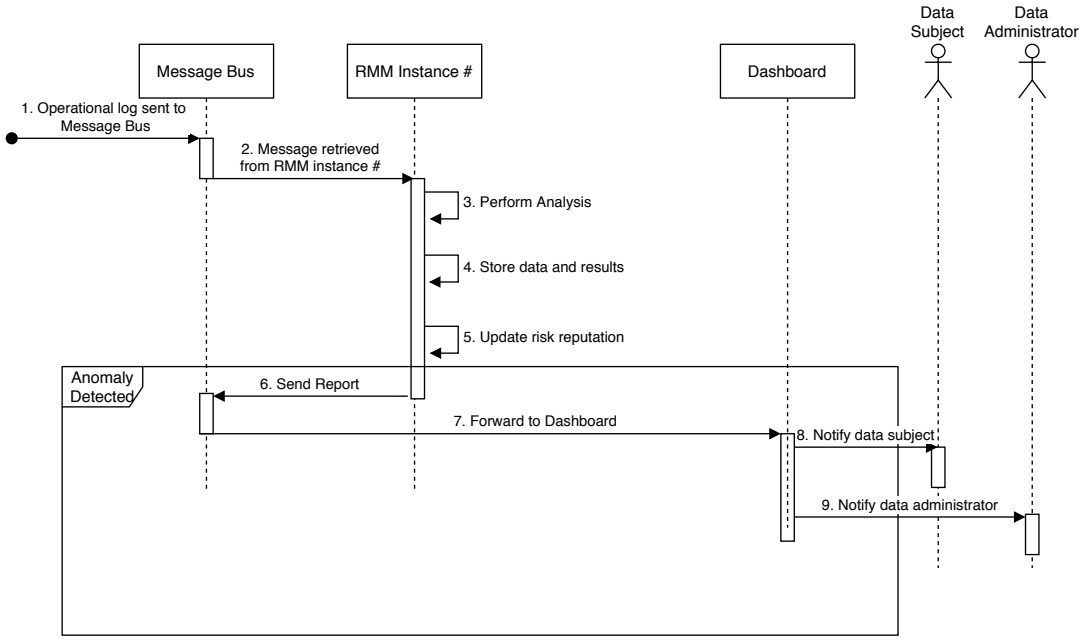

Figure 4.10: Sequence diagram envisioned for the fourth scenario.

4.4 Aggregation of Reputation

The architectural scenarios, on their own, still do not solve a major issue regarding reputation under distributed environments. If it were to put on a question, the problem can be phrased as:

"*Having multiple RMM instances running in a distributed environment, what is the overall reputation of a data processor? How to reach this value?*"

First, we need to understand the way in which the reputation is stored, because the overall reputation of a data processor is sent from a RMM instance, on the reputation storage retrieval with additional processing.

The fields/columns that are related to the reputation storage are:

- **RMM Instance ID**: A Universally unique identifier (UUID) generated for each instance
- **Data processor**: A string value from the set of available data processors
- **Alpha**: The number of events where the data processor is expressing normal behavior

- **Beta**: The number of events where the data processor is expressing anomalous behavior

Table 4.3 presents an example of a possible configuration for the reputation storage.

RMM Instance	Data Processor	Alpha	Beta
....	....	....	....
2e1c79d9-0772-4f3b-b2 d5-4ae6ed21f671	Santander	30	13
cef841a6-34ae-4ffc-83 65-9c1cce6ed8c9	MITA	40	3
847d4f77-52ab-49f7-a5 04-2211155c24ff	MEF	17	19
....	....	....	....

Table 4.3: Example of reputation storage configuration and values.

The simplest solution would be to compute the average of the reputation. Considering Table 4.4, the average value of the pair (alpha, beta) would be (32, 7), from which using equation 2.7, would reach a reputation value of 0.82, for Santander data processor. The key-issue here, as one can imagine, is that every data processor is assigned the same weight.

RMM Instance	Data Processor	Alpha	Beta
....	....	....	....
2e1c79d9-0772-4f3b-b2 d5-4ae6ed21f671	Santander	30	10
cef841a6-34ae-4ffc-83 65-9c1cce6ed8c9	Santander	50	6
847d4f77-52ab-49f7-a5 04-2211155c24ff	Santander	17	5
....	....	....	....

Table 4.4: Reputation storage configuration for the same data processor.

Therefore, an improved solution can be assigning different weights for each RMM instance, taking into consideration the number of messages processed by the RMM instance, for a data processor. Hence, the more messages processed by a RMM instance, for a given data processor, higher its weight. The weight can be further calculated, as a percentage of the total number of messages processed for a given data processor. This is better illustrated in Table 4.5.

RMM Instance	Data Processor	Alpha	Beta	Messages Processed	Weight (%)
....	....	....	....	....	....
2e1c79d9-0772-4f3b-b2 d5-4ae6ed21f671	Santander	30	10	40	34%
cef841a6-34ae-4ffc-83 65-9c1cce6ed8c9	Santander	50	6	56	47%
847d4f77-52ab-49f7-a5 04-2211155c24ff	Santander	17	5	22	19%
....	....	....	....	....	....

Table 4.5: Reputation storage configuration with the addition of a number of messages processed field.

In this case, applying the equation 2.7 and assigning the weights to each RMM instance, the overall reputation value for the Santander data processor would also be 0.82.

4.5 Summary

This chapter presented the work related to the development of the reputation system for the RMM. First, a requirement analysis was performed, taking into consideration the needs of the PoSeID-on project.

Then, several reputation system solutions were compared with respect to their provided features and the one that matches the requirements needed for PoSeID-on was selected. Next, a discussion about the exploitation of the selected reputation system under a distributed environment was made.

Further, a set of architectural scenarios for the distributed environment was presented, along with their properties and downsides. Then, the issue of aggregating reputation of several instances was introduced. We proposed two solutions for addressing this issue, being one of them an improved version of the other.

Chapter 5

Results and Discussion

The objects of assessment in the experiments to follow are the baseline and the fourth architectural scenario, previously described. With respect to the aggregation of reputation to be evaluated for the fourth scenario, we will proceed with the improved solution that assigns different weights to different RMM instances, based on the number of messages processed.

The methodology for the evaluation of the reputation system will consist on first, generating a dataset that reproduces a series of anomaly detection result analysis, involving data processors. This dataset represents what should be the outcome from the batch layer, that are further sent to the service layer.

Then, we will send this dataset into the RMM message queue, from which any instance is able to connect and consume the messages. Each instance will process the messages, updating the values of reputation with respect to the data processors. Lastly, we will send a message requesting the values of reputation for all the data processors and compare the output given by the instance with the theoretical one.

In the next sections, the dataset, experimental setup and the collected results will be described.

5.1 Dataset

The dataset was generated according to a set of parameters developed by the author. The data processors considered for this dataset are: MEF, MITA, Santander and Softeam. The list of parameters and respective values used, are presented in Table 5.1.

Number of Entries	20000
Anomaly Percentage of MEF	30%
Anomaly Percentage of MITA	20%
Anomaly Percentage of Santander	10%
Anomaly Percentage of Softeam	5%

Table 5.1: Parameters used for generating the dataset.

The anomaly percentage was set different for each data processor, in order to observe its reflection into the reputation score. Also, an event had an equally random probability of being created for any data processor (i.e. 25% for each).

With all those parameters set, the dataset was generated and it is described in Table 5.2.

MEF - Number of normal events	3503
MEF - Number of anomaly events	1522
MITA - Number of normal events	3975
MITA - Number of anomaly events	1015
Santander - Number of normal events	4509
Santander - Number of anomaly events	511
Softeam - Number of normal events	4714
Softeam - Number of anomaly events	251

Table 5.2: Description of the generated dataset.

As we can observe, the values of anomaly events for each data processor matches with the percentage parameter defined for the dataset.

5.2 Experimental Setup

The tests were conducted in the same machine where the staging environment is hosted, although not in the real PoSeID-on platform in order to avoid any unexpected behaviour to affect the environment. The machine specifications are presented in Table 5.3.

Model	Dell Precision 5820 Tower X-Series
Operating System (OS)	Ubuntu 18.04.1 LTS
CPU	Intel(R) Core (TM) i9-7920X CPU @ 2.90GHz
Random Access Memory (RAM)	132GB, 2666 MHz, DDR4
GPU	NVIDIA Quadro P1000, 4 GB DDR5
Storage	1TB 7200rpm SATA Hard Drive

Table 5.3: Machine specifications.

Applications needed for the RMM such as Cassandra, Redis and RabbitMQ were also deployed in the same machine, in order to reproduce the components of the PoSeID-on platform environment that the RMM interacts with.

The RMM application, with all its dependencies, was packed into a Java Archive (JAR) and each instance was executed with the Java Virtual Machine (JVM) memory parameters and values, specified in Table 5.4.

Parameter	**Value**
Maximum Memory Allocation (-Xmx)	20 GB
Initial Memory Allocation (-Xms)	20 GB

Table 5.4: JVM parameters set to the RMM application.

The RMM also expects a series of environment variables to be set, in order to be successfully deployed. Table 5.5 presents the variables, their descriptions and values used for the experiment.

Environment Variable	Description	Value
LOCAL_THREADS	The number of threads to be used in a local cluster mode. For the experiment, we used 5 threads.	5
CA_PUB	Path to the Certificate Authority (CA) public key.	"./keys/ca.pub"
RMM_KEY	Path to the public key of the RMM.	"./keys/rmm.key"
RMM_CERT	Path to the certificate of the RMM.	"./keys/rmm.cert"
ENV	Can be either "production" or "debug". They differ from a logging level perspective.	"debug"
CASSANDRA_DB_USERNAME	Username to connect to Cassandra.	"cassandra"
CASSANDRA_DB_PASSWORD	Password to connect to Cassandra.	"cassandra"
CASSANDRA_DB_SERVICE_HOST	IP address to connect to Cassandra.	localhost
CASSANDRA_DB_SERVICE_PORT	Port to connect to Cassandra.	9042
MESSAGE_BROKER_AMQP_SERVICE_HOST	IP address to connect to RabbitMQ.	localhost
MESSAGE_BROKER_AMQP_SERVICE_PORT	Port to connect to RabbitMQ.	9352
EXECUTOR_MEMORY	Amount of memory to use per executor process in Spark.	"10g"
DRIVER_MEMORY	Amount of memory to use for the driver process in Spark.	"10g"
REDIS_SERVICE_HOST	IP address to connect to Redis.	localhost
REDIS_SERVICE_PORT	Port to connect to Redis.	6379
REDIS_DB	The number of the database used for Redis.	2

Table 5.5: Environment variables description and values used for the experiment.

5.3 Reputation System Evaluation

As stated in the previous sections, we will consider first the baseline scenario into the assessment of the reputation system. Hence, after the processing of the messages from the only RMM instance, the results are presented in Table 5.6.

RMM Instance	Data Processor	Alpha	Beta
2ef4ac16-fc2f-4e72-8084-c864191c96f2	MEF	3503	1522
2ef4ac16-fc2f-4e72-8084-c864191c96f2	MITA	3975	1015
2ef4ac16-fc2f-4e72-8084-c864191c96f2	Santander	4509	511
2ef4ac16-fc2f-4e72-8084-c864191c96f2	Softeam	4714	251

Table 5.6: Baseline - Table of results after the processing of the messages from the dataset.

One thing to notice is that the values in the table, matches exactly with what is expected, if we consider the mapping of normal events into the alpha column, and the anomaly events into the beta column, from the generated dataset.

Next, we performed a query for the values of reputation of all the data processors. The results are shown in Table 5.7.

Data Processor	Reputation
MEF	0.70
MITA	0.80
Santander	0.90
Softeam	0.95

Table 5.7: Baseline - Reputation values of all the data processors

From the results, we can state two important things:

1. The reputation value is related with the percentage of anomalies in which the data processor is involved.
2. Once again, we validate the correctness of the developed reputation system, as the values of reputation are exactly the same as if we compute the reputation given the values of alpha and beta.

For the fourth scenario evaluation, we will consider three RMM instances. More instances could be added as well, but for this evaluation, proving that the aggregation of reputation works as intended should be sufficient for three instances.

After sending the dataset through the message bus and instantiating three instances, we've collected the results shown in Table 5.8.

RMM Instance	Data Processor	Alpha	Beta
eae6d4de-4e42-4174-a4ea-4e2fc57abf1d	MEF	51	21
eae6d4de-4e42-4174-a4ea-4e2fc57abf1d	MITA	47	5
eae6d4de-4e42-4174-a4ea-4e2fc57abf1d	Santander	65	4
eae6d4de-4e42-4174-a4ea-4e2fc57abf1d	Softeam	50	1
c8187b84-3d13-4dfe-b267-e877cdc6b426	MEF	2803	1205
c8187b84-3d13-4dfe-b267-e877cdc6b426	MITA	3131	803
c8187b84-3d13-4dfe-b267-e877cdc6b426	Santander	3573	417
c8187b84-3d13-4dfe-b267-e877cdc6b426	Softeam	3769	197
32869142-abdd-4444-9d4b-50e79ff4f2cc	MEF	649	296
32869142-abdd-4444-9d4b-50e79ff4f2cc	MITA	797	207
32869142-abdd-4444-9d4b-50e79ff4f2cc	Santander	871	90
32869142-abdd-4444-9d4b-50e79ff4f2cc	Softeam	895	53

Table 5.8: Fourth Scenario - Table of results after the processing of the messages from the dataset.

We can observe that the slightly difference at the start time of the instances could result in an unbalanced message consumption. It means that the instances are able to receive a large number of messages, before even processing them. Table 5.9 presents the weight of each RMM instance, where the instances are numbered according to order of appearance (top to bottom) in the previous table.

Data Processor	RMM Instance #1	RMM Instance #2	RMM Instance #3
MEF	1.5%	80%	18.5%
MITA	1.2%	78.8%	20%
Santander	1.5%	79.2%	19.3%
Softeam	1.1%	80%	18.9%

Table 5.9: Fourth Scenario - Percentage of messages received from the instances.

The second RMM instance, therefore, will be assigned a higher weight, followed by the third instance and lastly, the first instance. The reputation that each RMM instance perceive for the data processors are presented in Table 5.10.

Reputation	RMM Instance #1	RMM Instance #2	RMM Instance #3
MEF	0.70	0.69	0.68
MITA	0.90	0.79	0.79
Santander	0.94	0.89	0.90
Softeam	0.98	0.95	0.94

Table 5.10: Fourth Scenario - Reputation perceived by each RMM instance.

The results corroborate that the improved version of the aggregation solution is a clever option, because if the same weight was attributed to each instance, this would not lead to an adequate reputation value, due to the disparate values achieved from instance #1 related to the others.

Last but not least, we performed a request for the overall value of reputation of all data processors. The output is presented in Table 5.11.

Data Processor	Reputation
MEF	0.70
MITA	0.80
Santander	0.90
Softeam	0.95

Table 5.11: Fourth Scenario - Overall reputation values of all the data processors

We can conclude that the reputation values achieved, for both the baseline and the architectural scenario four, are the same. Therefore, the aggregation solution is proven to be adequate and reasonably sound, opposing to the naive solution, which would've reached different reputation values.

Hence, this lead to the conclusion that the developed reputation system is working properly in a distributed environment, which was the ultimate objective to be fulfilled in this thesis.

5.4 Summary

This chapter presented the results and discussion of the reputation system developed. Initially, the methodology of evaluation was explained including the scenarios that will be evaluated.

Then, the dataset that was generated and used for the assessment was described and presented, along with its set of parameters. Further, the experimental setup was shown, presenting the specification of the machine used for evaluation and the RMM configuration parameters.

Lastly, the evaluation of the reputation system for the baseline and the fourth scenario was performed, presenting the results gathered and validating the aggregation of reputation.

Chapter 6

Conclusion and Future Work

The work addressed in this thesis focused in the reputation system part of the Risk Management Module, which is a core component of PoSeID-on, with respect to risk assessment. The reputation system will evaluate the data processors that are part of the PoSeID-on platform, in order to facilitate trust between data subjects and data processors. The outputs of the reputation system will be used by other components of PoSeID-on, such as the PDA and Dashboard.

The work started with a background study about the PoSeID-on project, its goals and architecture. Then, the General Data Protection Regulation was explored as a way to understand the boundaries of compliance needed for the future work. In parallel, a state-of-the-art analysis on Blockchain and Smart contracts was conducted, because the initial work plan was to develop a blockchain-based solution to be integrated into PoSeID-on.

After analysis and as the objectives of the project became clear, the author became responsible for the lead development of the RMM and, specially, of the reputation system. Hence, a state-of-the-art study on reputation system was performed and documented in this thesis.

Then, an in-depth study about the details of the RMM implementation was made, in order to start the development of both the batch and service layer. Both of them were implemented, but only the latter was validated due to still unknown memory issues.

The next step was to perform a requirement analysis of the PoSeID-on needs, with respect to the reputation system. Comparing the needs with the features provided by the solutions found in the literature, we decided for the implementation of a probabilistic one, namely the Beta Reputation System.

The selected reputation system is characterized as being centralized, but a discussion about its exploitation into a working version under a distributed environment was created. A number of benefits was raised for this challenge, such as scalability, resilience, novelty and alignment with the project future goals.

After accepting the challenge, a series of scenarios of the distributed architecture of RMM was made, specifically four. After brainstorming, only two of them were selected to the implemented for evaluation - the baseline and the fourth scenario.

Further, the aggregation of reputation under a distributed environment was the topic of discussion. Through the analysis of the storage configuration, we've reached into two solutions, and opted to proceed with the improved one.

In the evaluation phase, a dataset needed to be generated to describe either a normal or anomaly event, involving a specific data processor. Then, after its creation, the RMM was deployed on the staging environment machine, along with its dependencies and applications needed. Evaluation steps and results were collected for the baseline and fourth scenario, they've proven that the reputation system works in a distributed environment.

From the objectives stated in Chapter 1, we can conclude that the only thing missing is the validation of the batch layer, which will be performed before the end of the project, as a future step. Also as a future step, the author intends to submit a paper of the work performed in this thesis.

References

[1] P. Voigt and A. v. d. Bussche, *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer Publishing Company, Incorporated, 1st ed., 2017.

[2] POSEIDON, "The project." URL https://www.poseidon-h2020.eu/the-project/. Accessed: 2020-01-13.

[3] D. Allessie, M. Sobolewski, and L. Vaccari, *Blockchain for digital government An assessment of pioneering implementations in public services*. European Commission, 2019.

[4] S. Vavilis, P. Milan, and N. Zannone, "Impact of ict on home healthcare," vol. 386, 09 2012.

[5] P. S. R. Casaleiro and all, "Deliverable 2.2 system requirements and architecture," 2018. PoSeID-on H2020.

[6] GDPR.EU, "What is gdpr, the eu's new data protection law?)." URL https://gdpr.eu/what-is-gdpr/. Accessed: 2020-01-17.

[7] GDPR.EU, "Art. 22 gdpr automated individual decision-making, including profiling." URL https://gdpr.eu/article-22-automated-individual-decision-making/. Accessed: 2020-01-13.

[8] P. Martins, *Introdução à Blockchain - Bitcoin, Criptomoedas, Smart Contracts, Conceitos, Tecnologia, Implicações, FCA , 2018*. FCA, 2018.

[9] M. Thake, "What's the difference between blockchain and DLT? - nakamo.to - Medium." URL https://medium.com/nakamo-to/whats-the-difference-between-blockchain-and-dlt-e4b9312c75dd. Accessed: 2020-01-16.

[10] Lisk.io, "What is blockchain?." URL https://lisk.io/what-is-blockchain. Accessed: 2020-01-16.

[11] P. Tasca and C. Tessone, "A taxonomy of blockchain technologies: Principles of identification and classification," *Ledger*, vol. 4, 2019.

[12] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE 6th International Congress on Big Data*, 2017.

[13] J. Evans, "Blockchain Nodes: How They Work (All Types Explained) - Nodes.com." URL https://nodes.com/. Accessed: 2020-01-16.

[14] Lisk.io, "What is Hashing? Learn about Blockchain | Lisk Academy." URL https://lisk.io/academy/blockchain-basics/how-does-blockchain-work/what-is-hashing. Accessed: 2019-09-23.

[15] Lisk.io, "What is proof of stake?." URL https://lisk.io/academy/blockchain-basics/how-does-blockchain-work/proof-of-stake. Accessed: 2019-10-23.

[16] D. Lee Kuo Chuen, "Handbook of digital currency," tech. rep., Elsevier, 2015.

[17] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9, 1997.

[18] R. A. Küfner, "How smart are smart contracts?." URL https://medium.com/nakamo-to/how-smart-are-smart-contracts-759a1faec8e0. Accessed: 2020-01-16.

[19] R. A. Küfner, "Breaking down the smart contract." URL https://medium.com/nakamo-to/breaking-down-the-smart-contract-45b249b8bc71. Accessed: 2020-01-16.

[20] T. E. U. B. Observatory and Forum, "Blockchain for government and public services." URL https://www.eublockchainforum.eu/sites/default/files/reports/eu_observatory_blockchain_in_government_services_v1_2018-12-07.pdf. Accessed: 2020-01-16.

[21] A. Offerman, "Swiss city of zug issues ethereum blockchain-based eids." URL https://joinup.ec.europa.eu/collection/egovernment/document/swiss-city-zug-issues-ethereum-blockchain-based-eids. Accessed: 2020-01-16.

[22] swissinfo.ch, "Switzerland's first municipal blockchain vote hailed a success." URL https://www.swissinfo.ch/eng/crypto-valley-_-switzerland-s-first-municipal-blockchain-vote-hailed-a-success/44230928. Accessed: 2020-01-16.

[23] A. Nawfal, "Zug residents can now ride e-bikes using their uport-powered zug digital ids." URL https://medium.com/uport/zug-residents-can-now-ride-e-bikes-using-their-uport-powered-zug-digital-ids-7ed31ac9d621. Accessed: 2020-01-16.

[24] R. Aitken, "Bitland's african blockchain initiative putting land on the ledger." URL https://www.forbes.com/sites/rogeraitken/2016/04/05/bitlands-african-blockchain-initiative-putting-land-on-the-ledger/. Accessed: 2020-01-16.

[25] N. De, "Indian state partners with blockchain startup for land registry pilot." URL https://www.coindesk.com/andhra-pradesh-partners-with-chromaway-to-develop-blockchain-land-registry. Accessed: 2020-01-16.

[26] C. Kim, "Sweden's land registry demos live transaction on a blockchain." URL https://www.coindesk.com/sweden-demos-live-land-registry-transaction-on-a-blockchain. Accessed: 2020-01-16.

[27] e estonia, "Healthcare." URL https://e-estonia.com/solutions/healthcare/e-health-record/. Accessed: 2020-01-16.

[28] M. Marenco, "A nordic way to blockchain in healthcare." URL https://www.himss.eu/himss-blog/nordic-way-blockchain-healthcare. Accessed: 2020-01-16.

[29] I. F. the Future (IFF) UNIC, "Blockchain certificates (academic & others)." URL https://www.unic.ac.cy/iff/blockchain-certificates/. Accessed: 2020-01-16.

[30] N. V. Patel, "Malta pilots blockchain-based credentials program." URL https://spectrum.ieee.org/tech-talk/computing/networks/malta-pilots-blockchainbased-credentials-program. Accessed: 2020-01-16.

[31] A. Maak, "West virginia introduces blockchain voting app for midterm election." URL https://slate.com/technology/2018/09/west-virginia-blockchain-voting-app-midterm-elections.html?via=gdpr-consent. Accessed: 2020-01-16.

[32] M. del Castillo, "Russia is leading the push for blockchain democracy." URL https://www.coindesk.com/russias-capital-leading-charge-blockchain-democracy. Accessed: 2020-01-16.

[33] E. Parliament, "European parliament resolution of 3 october 2018 on distributed ledger technologies and blockchains: building trust with disintermediation (2017/2772(rsp))." URL http://www.europarl.europa.eu/doceo/document/TA-8-2018-0373_EN.html. Accessed: 2020-01-16.

[34] S. Takhar and K. Liyanage, "Blockchain application in supply chain chemical substance reporting.," *22nd Cambridge International Manufacturing Symposium*, 2018.

[35] E. Commission, "Vat: Eu member states still losing almost €150 billion in revenues." URL https://ec.europa.eu/commission/news/vat-eu-member-states-still-losing-almost-eu150-billion-revenues-2018-sep-21_en. Accessed: 2020-01-16.

[36] summito, "Application." URL https://summitto.com/application.html. Accessed: 2020-01-16.

[37] S. Higgins, "The eu is building a 'financial transparency gateway' with blockchain." URL https://www.coindesk.com/eu-developing-prototype-blockchain-platform-public-company-data. Accessed: 2020-01-16.

[38] E. Parliament, "Blockchain and the general data protection regulation - can distributed ledgers be squared with european data protection law?." . URL https://www.europarl.europa.eu/RegData/etudes/STUD/2019/634445 /EPRS_STU(2019)634445_EN.pdf, 2019. Accessed: 2020-01-13.

[39] F. Hendrikx, K. Bubendorfer, and R. Chard, "Reputation systems: A survey and taxonomy," *Journal of Parallel and Distributed Computing*, vol. 75, 08 2014.

[40] L. Mui, M. Mohtashemi, and A. Halberstadt, "A computational model of trust and reputation," pp. 2431 – 2439, 02 2002.

[41] A. Jøsang, "Trust and reputation systems," in *Foundations of Security Analysis and Design IV* (A. Aldini and R. Gorrieri, eds.), (Berlin, Heidelberg), pp. 209–245, Springer Berlin Heidelberg, 2007.

[42] S. Tadelis, *Firm reputation with hidden information*, pp. 537–553. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[43] L. Mui, M. Mohtashemi, and A. Halberstadt, "Notions of reputation in multi-agents systems: A review," pp. 280–287, 01 2002.

[44] D. Gambetta *et al.*, "Can we trust trust," *Trust: Making and breaking cooperative relations*, vol. 13, pp. 213–237, 2000.

[45] R. Falcone and C. Castelfranchi, "Social trust: A cognitive approach," pp. 55–90, 05 2001.

[46] E. Koutrouli and A. Tsalgatidou, "Reputation-based trust systems for p2p applications: Design issues and comparison framework," pp. 152–161, 09 2006.

[47] P. Resnick, "Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system," *Advances in Applied Microeconomics*, vol. 11, 10 2002.

[48] A. Abdul-Rahman and S. Hailes, "Supporting trust in virtual communities," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, pp. 9 pp. vol.1–, 2000.

[49] D. W. Manchala, "Trust metrics, models and protocols for electronic commerce transactions," in *Proceedings. 18th International Conference on Distributed Computing Systems (Cat. No.98CB36183)*, pp. 312–321, 1998.

[50] J. Sabater-Mir, "Regret: A reputation model for gregarious societies," 08 2002.

[51] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[52] T. D. Huynh, N. Jennings, and N. Shadbolt, "Fire: An integrated trust and reputation model for open multi-agent systems," *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 13, pp. 18–22, 01 2004.

[53] R. Molva, "Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Communication and Multimedia Security*, vol. 228, pp. 107–121, 01 2002.

[54] G. Zacharia, A. Moukas, and P. Maes, "Collaborative reputation mechanisms for electronic marketplaces," *Decision Support Systems*, vol. 29, pp. 371–388, 12 2000.

[55] K. Aberer and Z. Despotovic, "Managing trust in a peer-2-peer information system," in *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, p. 310, 01 2001.

[56] E. Damiani, D. Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks," *Proceedings of the ACM Conference on Computer and Communications Security*, 12 2002.

[57] Z. Malik and A. Bouguettaya, "Rateweb: Reputation assessment for trust establishment among web services," *VLDB J.*, vol. 18, pp. 885–911, 08 2009.

[58] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in *Proceedings of the 12th international conference on World Wide Web*, pp. 640–651, 2003.

[59] A. Simone, B. Škorić, and N. Zannone, "Flow-based reputation: more than just ranking," *International Journal of Information Technology & Decision Making*, vol. 11, no. 03, pp. 551–578, 2012.

[60] L. Xiong and L. Liu, "Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities," *IEEE transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843–857, 2004.

[61] A. Josang and R. Ismail, "The beta reputation system," in *Proceedings of the 15th bled electronic commerce conference*, vol. 5, pp. 2502–2511, 2002.

[62] W. L. Teacy, J. Patel, N. R. Jennings, and M. Luck, "Travos: Trust and reputation in the context of inaccurate information sources," *Autonomous Agents and Multi-Agent Systems*, vol. 12, no. 2, pp. 183–198, 2006.

[63] A. Josang and J. Haller, "Dirichlet reputation systems," in *The Second International Conference on Availability, Reliability and Security (ARES'07)*, pp. 112–119, IEEE, 2007.

[64] A. Josang, R. F. Hayward, and S. Pope, "Trust network analysis with subjective logic," 2006.

[65] S. Ries, S. M. Habib, M. Mühlhäuser, and V. Varadharajan, "Certainlogic: A logic for modeling trust and uncertainty," in *International conference on trust and trustworthy computing*, pp. 254–261, Springer, 2011.

[66] T. van Deursen, P. Koster, and M. Petković, "Hedaquin: A reputation-based health data quality indicator," *Electronic Notes in Theoretical Computer Science*, vol. 197, no. 2, pp. 159–167, 2008.

[67] S. Schmidt, R. Steele, T. S. Dillon, and E. Chang, "Fuzzy trust evaluation and credibility development in multi-agent systems," *Applied Soft Computing*, vol. 7, no. 2, pp. 492–505, 2007.

[68] N. Marz, "Lambda architecture." URL http://lambda-architecture.net/. Accessed: 2020-07-27.

[69] T. A. S. Foundation, "Apache cassandra." URL https://cassandra.apache.org/. Accessed: 2020-09-28.

[70] R. Labs, "Redis." URL https://redis.io/. Accessed: 2020-09-28.

[71] S. He, J. Zhu, P. He, and M. R. Lyu, "Experience report: System log analysis for anomaly detection," in *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)*, pp. 207–218, IEEE, 2016.

[72] R. J. F. Casaleiro, "Risk management in blockchain based systems," Master's thesis, Universidade de Coimbra, 2020.

[73] S. Vavilis, M. Petković, and N. Zannone, "A reference model for reputation systems," *Decision Support Systems*, vol. 61, 05 2014.

[74] N. Magaia, P. Pereira, and M. Correia, "Repsys: A robust and distributed reputation system for delay-tolerant networks," in *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 289–293, 2017.

[75] S. Buchegger and J.-Y. Le Boudec, "A robust reputation system for p2p and mobile ad-hoc networks," in *Second Workshop on Economics of P2P Systems, Boston*, 2004.