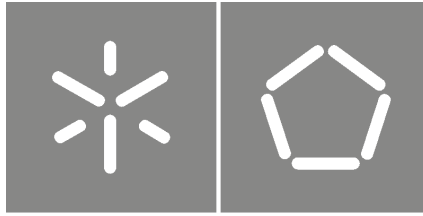


**Universidade do Minho**  
Escola de Engenharia

Mafalda Guimarães Nunes

**Privacy and Security  
in Data Mining**

fevereiro de 2021



**Universidade do Minho**  
Escola de Engenharia

Mafalda Guimarães Nunes

**Privacy and Security  
in Data Mining**

Dissertação de Mestrado  
Mestrado Integrado em Engenharia Informática

Trabalho efetuado sob a orientação do  
**Professor Doutor José Carlos Bacelar Almeida**

fevereiro de 2021

## **DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS**

---

Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada.

Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositóriUM da Universidade do Minho.

### ***Licença concedida aos utilizadores deste trabalho***



### **Atribuição**

### **CC BY**

<https://creativecommons.org/licenses/by/4.0/>

## **ACKNOWLEDGEMENTS**

---

I would like to thank the following people for helping with this project:

My supervisor at the University of Minho, Dr. José Carlos Bacelar Almeida, for his guidance and insights in the data security field.

Altice Labs staff, for their welcoming reception at the company. A special thanks to my company supervisor, Luís Cortesão, and Paula Cravo, for their continued support and guidance throughout this project. Ricardo Machado, for his support regarding the health-related scenarios, as well as the development and results project phases. Telma Mota e Bruno Vitor Cabral for their support regarding the health-related scenarios. Isilda Costa, for her clarifications regarding the data protection legal context, namely the GDPR.

Bernardo Portela, professor at the Faculdade de Ciências da Universidade do Porto (FCUP) and researcher at HASLab/INESC TEC, for his insights in the data security field, especially regarding techniques to perform operations over encrypted data.

**STATEMENT OF INTEGRITY**

---

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

## RESUMO

---

Na atualidade, cada vez mais informação pessoal é recolhida, armazenada, analisada e partilhada. Tal é possível e encorajado devido a avanços na área tecnológica, nos métodos de armazenamento de dados e nas capacidades analíticas, incluindo *machine learning*.

Apesar deste novo paradigma de recolha de dados oferecer grandes oportunidades de negócio, a quantidade crescente de informações, que podem ser bastante sensíveis, levanta graves questões de privacidade e segurança. Existe uma necessidade cada vez maior de adaptar os mecanismos existentes de exploração de dados, com garantias de segurança, a cenários reais. Este trabalho explorará alguns desses mecanismos, tomando em consideração requisitos funcionais e de desempenho.

Para compreender melhor o contexto legal e os mecanismos de exploração de dados com garantias de segurança disponíveis, também conhecidos como *Privacy Enhancing Technologies (PETs)*, realizou-se uma pesquisa preliminar. De seguida, vários cenários de exploração de dados adequados para a empresa Altice/MEO foram definidos e possíveis soluções apresentadas. Entre esses cenários, um foi escolhido para implementação, devido aos desafios tecnológicos associados e à sua relevância a nível do negócio.

O cenário escolhido enquadra-se na área de *e-Health*, dizendo respeito à cifragem dos dados geridos por uma plataforma de vida assistida. Estes dados só podem ser decifrados por utilizadores autorizados, o que não inclui o servidor da plataforma. Assim sendo, este servidor deve ser capaz de realizar as operações necessárias sobre os dados cifrados. Para solucionar este problema, um esquema foi proposto, um protótipo desenvolvido e uma avaliação comparativa de usabilidade efetuada.

A avaliação de usabilidade revelou que o custo da aplicação do esquema proposto, em termos de desempenho, ocupação do espaço e gestão de chaves, é aceitável. Devido à natureza muito sensível das informações envolvidas, a melhoria das garantias de privacidade e segurança, com manutenção da funcionalidade, supera o referido custo.

Apesar de ainda existir a necessidade de desenvolver novas PETs e melhorar a eficácia das existentes, os resultados obtidos permitem concluir que atualmente existem implementações que podem ser aplicadas a cenários reais.

**Palavras-chave:** Criptografia; Privacidade; *Privacy Enhancing Technologies*; Segurança.

## **ABSTRACT**

---

Personal information about individuals is currently being collected, stored, analyzed, and shared in increasingly greater quantities. This is possible and encouraged due to advances in technology, data storage methods, and analytical capabilities, including machine learning.

Although this new data collection paradigm offers great business opportunities, the growing amount of information, which can be quite sensitive, raises serious privacy and security issues. There is an increasing need to adapt existing data exploration mechanisms with security guarantees to real scenarios. This work will explore some of those mechanisms, taking into account their functionalities and performance.

To better understand the legal context and the available data exploration mechanisms with security guarantees, also known as Privacy Enhancing Technologies (PETs), preliminary research was carried out. Then, several data exploration scenarios suitable for the company Altice/MEO were defined, and possible solutions were presented. Between these scenarios, one was chosen for implementation due to the related technological challenges and its relevance at the business level.

The chosen scenario fits in the area of e-Health, concerning the encryption of data managed by an assisted living platform. This data can only be decrypted by authorized users, which does not include the platform's server. Therefore, this server should be able to perform the needed operations over the encrypted data. To solve this problem, a scheme was proposed, a prototype implemented, and a comparative usability evaluation carried out.

The usability evaluation revealed that the cost of applying the proposed scheme, in terms of performance, space occupation, and key management, is acceptable. Given the very sensitive nature of the handled information, the increase in privacy and security, while maintaining the functionality, far outweighs the mentioned cost.

Even if there is still the need to develop new PETs and improve the existing ones' effectiveness, the obtained results allow concluding that currently there are implementations that can be applied to real scenarios.

**Keywords:** Cryptography; Privacy; Privacy Enhancing Technologies; Security.

---

## CONTENTS

---

1	INTRODUCTION	1
2	STATE OF THE ART	3
2.1	General Data Protection Regulation	3
2.2	Privacy Enhancing Technologies	10
2.2.1	Anonymization and Pseudonymization	11
2.2.2	Privacy-Preserving Record Linkage	16
2.2.3	Privacy-Preserving Storage	17
2.2.4	Privacy-Preserving Storage and Computation	22
2.2.5	Privacy-Preserving Computation	29
2.3	Summary	31
3	SCENARIOS	34
3.1	Cross-service Profiling	34
3.2	Health – Assisted Living Platform	36
3.2.1	Database Encryption	38
3.2.2	Data Enrichment with Third Parties	42
3.3	Final Considerations	46
3.4	Summary	46
4	CHALLENGES	48
4.1	Client Devices with Limited Resources	48
4.2	Cross-devices Usage	50
4.3	Access Policy Update	51
4.4	Server operations	52
4.5	Summary	54
5	DEVELOPMENT	55
5.1	Decisions	55
5.1.1	REST Services	55
5.1.2	Authentication	56



5.1.3	Cryptography	58
5.2	Implementation	70
5.2.1	System Architecture	70
5.2.2	Data Storage	72
5.2.3	Communication Flow	74
5.3	Summary	80
6	RESULTS AND DISCUSSION	83
6.1	Performance Evaluation	84
6.1.1	Setup	84
6.1.2	Evaluation	86
6.2	Databases Size	89
6.2.1	Setup	89
6.2.2	Evaluation	89
6.3	Summary	91
7	CONCLUSION	92
7.1	Conclusions	92
7.2	Prospect for future work	94
A	USERS' PERSPECTIVE ON THE EASE OF EXERCISING THEIR RIGHTS	110
B	LEGAL PERSPECTIVE ON CUSTOMER RIGHTS – GOOGLE VS MEO	113
C	ADDITIONAL INFORMATION ABOUT PETS	133
c.1	Anonymization and Pseudonymization	133
c.2	k-Anonymity and its extensions	138
c.3	Differential privacy and related models	141
c.4	Privacy-Preserving Record Linkage	145
c.5	Traditional Encryption	155
c.6	Attribute-Based Encryption	161
c.7	Searchable encryption	167
c.8	Homomorphic encryption	173
c.9	Secure Multi-Party Computation	177

## **ACRONYMS**

---

3DES	Triple Data Encryption Standard
ABE	Attribute-Based Encryption
ABSE	Attribute-Based Searchable Encryption
AES	Advanced Encryption Standard
API	Application Programming Interface
ARM	Advanced RISC Machine
ASIC	Application-Specific Integrated Circuit
B2B	Business-to-Business
B2C	Business-to-Consumer
BMI	Body Mass Index
CA	Certification Authority
CIA	Confidentiality, Integrity, and Availability
CL-PKC	Certificateless Public Key Cryptography
CNPD	National Data Protection Commission
CORS	Cross-Origin Resource Sharing
CP-ABE	Ciphertext-Policy Attribute-Based Encryption
CPU	Central Process Unit
CSO	Chief Security Officers
CSS	Cascading Style Sheets
D2D	Device to Device
DDoS	Distributed Denial of Service

DES	Data Encryption Standard
DH	Diffie-Hellman
DLP	Discrete Logarithm Problem
DPA	Data Protection Authority
DPIA	Data Protection Impact Assessment
DPO	Data Protection Officer
DSA	Digital Signature Algorithm
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic-Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
ECP-ABE	Enhanced Ciphertext-Policy Attribute-Based Encryption
EdDSA	Edwards-curve Digital Signature Algorithm
EEA	European Economic Area
ERD	Entity Relationship Diagram
ETSI	European Telecommunication Standards Institute
EU	European Union
FALCON	FAst-Fourier Lattice-based COmpact Signatures Over NTRU
FHE	Fully Homomorphic Encryption
FPE	Format-Preserving Encryption
FPGA	Field-Programmable Gate Array
GCE	Garbled Circuit Evaluation
GCM	Galois/Counter Mode
GDPR	General Data Protection Regulation
GLS	Google Location Services
GPS	Global Positioning System
GPU	Graphics Processing Unit

HBC	Honest-but-curious
HE	Homomorphic Encryption
HIPAA	Health Insurance Portability and Accountability Act
HLSH	Hamming Locality-Sensitive Hashing
HMAC	Hash-based Message Authentication Code
HSM	Hardware Security Module
HTML	Hypertext Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IBE	Identity-Based Encryption
IBSE	Identity-Based Searchable Encryption
ICT	Information and Communications Technology
IdP	Identity Provider
IFP	Integer Factorization Problem
IIoT	Industrial Internet of Things
IND-CCA	Indistinguishability under Chosen-Ciphertext Attack
IND-CKA	Indistinguishably under Chosen Keyword Attack
IND-CPA	Indistinguishability under Chosen-Plaintext Attack
IoT	Internet of Things
IP	Internet Protocol
IT	Information Technology
JS	JavaScript
JSON	JavaScript Object Notation
KDF	Key Derivation Function
KEM	Key Encapsulation Mechanism
KGA	Keyword Guessing Attack
KGC	Key Generation Center
KP-ABE	Key-Policy Attribute-Based Encryption

LSH	Locality-Sensitive Hashing
LWE	Learning with Errors
M2M	Machine to Machine
MA-ABE	Multi-Authority Attribute-Based Encryption
MCC	Model Contract Clause
MIEI	Mestrado Integrado em Engenharia Informática
MKHE	Multi-Key Homomorphic Encryption
MPC	Multi-Party Computation
MQTT	Message Queuing Telemetry Transport
NHS	National Health Service
NIST	National Institute of Standards and Technology
NP	Non-deterministic Polynomial-time
OIDC	OpenID Connect
OPE	Order-Preserving Encryption
ORAM	Oblivious RAM
ORE	Order-Revealing Encryption
PBKDF2	Password-Based Key Derivation Function 2
PCORI	Patient-Centered Outcomes Research Institute
PET	Privacy Enhancing Technology
PGP	Pretty Good Privacy
PHE	Partially Homomorphic Encryption
PIA	Privacy Impact Assessment
PII	Personally Identifiable Information
PIR	Private Information Retrieval
PKCS	Public Key Cryptography Standards
PKI	Public-Key Infrastructure

PPC	Pay Per Click
PPE	Property-Preserving Encryption
PPRL	Privacy-Preserving Record Linkage
PRF	Pseudo-Random Function
PRNG	Pseudo-Random Number Generator
PSE	Public Searchable Encryption
PSI	Private Set Intersection
QI	Quasi-Identifier
RAM	Random Access Memory
RC4	Rivest Cipher 4
REST	Representational State Transfer
RNG	Random Number Generator
RSA	Rivest-Shamir-Adleman
SAML	Security Assertion Markup Language
SE	Searchable Encryption
SEPA	Single Euro Payments Area
SHE	Somewhat Homomorphic Encryption
SIMD	Single Instruction, Multiple Data
SLK	Statistical Linkage Key
SMS	Short Message Service
SOC	System On Chip
SOP	Same-Origin Policy
SP	Service Provider
SQL	Structured Query Language
SSE	Symmetric Searchable Encryption
SSL	Secure Sockets Layer
SSO	Single Sign-On

TLS	Transport Layer Security
UI	User Interface
UM	University of Minho
UN	United Nations
XML	Extensible Markup Language
XSRF	Cross-Site Request Forgery
XSS	Cross-Site Scripting

---

**LIST OF FIGURES**

---

Figure 1	General Data Protection Regulation (GDPR) summary.	32
Figure 2	PETs summary.	33
Figure 3	Diagram representing the proposed solution based on Privacy-Preserving Record Linkage (PPRL).	35
Figure 4	Assisted living platform profiles hierarchy.	37
Figure 5	Assisted living platform logical architecture.	38
Figure 6	Entities involved in the Database Encryption scenario solution.	40
Figure 7	Data writing and reading in the presented solution.	41
Figure 8	Computation stage sequence diagram for data enrichment with the third parties scenario.	45
Figure 9	Cross-service Profiling scenario summary.	46
Figure 10	WebAL related scenarios summary.	47
Figure 11	User's signature key pair generation, storage, and retrieval.	50
Figure 12	Database Encryption scenario challenges summary.	54
Figure 13	Key Generation Center (KGC) certificate structure.	62
Figure 14	Key-Policy Attribute-Based Encryption (KP-ABE) access structure for each profile.	65
Figure 15	System architecture.	71
Figure 16	KGC Core database structure.	72
Figure 17	Security Storage database structure.	73
Figure 18	WebAL database structure.	73
Figure 19	Login sequence diagram.	75
Figure 20	User registration sequence diagram.	76
Figure 21	Relationship registration sequence diagram.	77
Figure 22	Load key sequence diagram.	79
Figure 23	Encryption sequence diagram.	79
Figure 24	Decryption sequence diagram.	80



Figure 25	Technologies and cryptographic algorithms used in each entity.	81
Figure 26	Response time of the assisted living platform operations.	86
Figure 27	Performance evaluation of Attribute-Based Encryption (ABE) encryption and decryption operations.	87
Figure 28	Performance evaluation of FastORE operations.	88
Figure 29	Size of the assisted living platform databases.	90
Figure 30	ABE ciphertexts' size.	90
Figure 31	FastORE ciphertexts' size.	90
Figure 32	High-level static data masking architecture.	134
Figure 33	High-level on-the-fly data masking architecture.	134
Figure 34	High-level dynamic (proxy-based) data masking architecture.	135
Figure 35	Outline of PPRL two-party protocols.	146
Figure 36	Outline of PPRL three-party protocols.	146
Figure 37	Example of PPRL protocol.	147
Figure 38	Ciphertext-Policy Attribute-Based Encryption (CP-ABE) encryption and decryption representation.	164
Figure 39	KP-ABE encryption and decryption representation.	164

---

**LIST OF TABLES**

---

Table 1	Seven foundation principles for privacy by design.	7
Table 2	Differences between the data protection directive 95/46/EC and the GDPR.	10
Table 3	ABE implementations.	64
Table 4	Response time limits for different user perceptions.	84
Table 5	Devices used on the results experiment.	85
Table 6	Collected information.	113
Table 7	Privacy controls.	116
Table 8	Goals of Data Collection.	119
Table 9	Data Processing – Legal grounds.	121
Table 10	Data Processing – Types of treatment.	122
Table 11	Data Sharing – Legal Grounds.	122
Table 12	Data Review/Update.	124
Table 13	Data exportation.	124
Table 14	Data removal – General information.	125
Table 15	Data removal – Details specified only by Google.	126
Table 16	Data Security.	127
Table 17	Cookies – Concept and utility examples.	128
Table 18	Cookies – Types of cookies, according to their features.	128
Table 19	Cookies – Management.	130
Table 20	Complaints.	130
Table 21	Data anonymization.	131
Table 22	Shared database.	132
Table 23	Different obligations under the GDPR regarding identified, pseudonymized and anonymized data.	137

---

## INTRODUCTION

---

This dissertation, under the theme “Privacy and Security in Data Mining”, is developed in the context of the 5<sup>th</sup> grade of Mestrado Integrado em Engenharia Informática (MIEI), University of Minho (UM), as well as in a business context, at the premises of the Altice Labs company.

Currently, information is constantly collected from multiple devices (smartphones, sensors, gateways, ...), which can vary from basic network statistics to sophisticated data and detailed user information. Although this new data collection paradigm offers great business opportunities, the growing amount of information, which can be quite sensitive, raises serious privacy and security issues.

Consumers are increasingly concerned about these issues. Businesses are looking for ways to alleviate these concerns, not only in their interactions directly with consumers, in a Business-to-Consumer (B2C) model, but also in a Business-to-Business (B2B) context. The increase of global awareness, demand, and regulation for privacy led to a resurgence of interest, advances, and commercialization in the area of Privacy Enhancing Technologies (PETs). They are a powerful category of technologies that enable, enhance, and preserve data privacy throughout its lifecycle, helping to achieve compliance with privacy policies or data protection legislation. An example of such legislation is the General Data Protection Regulation (GDPR), which will be further discussed in the subsequent chapter.

Data can be considered to have three stages in its lifecycle: at rest, in transit, and in use. The usage of PETs is mainly focused on the third stage, in which data assets are meaningfully used or processed, and also in the first stage, where the applied technologies somehow facilitate the use of stored data.

Organizations often need to perform operations such as sharing, searches, or analytics over data, creating data exposure points. PETs are designed to help eliminate this vulnerability by enabling data to be securely and privately processed. Examples of areas where these technologies can be applied are data analysis, machine learning, e-Health, cloud computation, and Internet of Things (IoT).

The desire for privacy is not a passing trend. Whether led by government regulation or consumer demand, organizations must be ready to operate in a world that prioritizes data security and privacy.

Thus, there is an increasing need to adapt existing data (e.g., big data) exploration mechanisms with security guarantees to real scenarios. This project has as main objectives the identification, evaluation, and implementation of some of these mechanisms, always considering their performance.

This dissertation should allow Altice Labs to understand what technologies could be used, with acceptable efficiency, to offer more privacy and security guarantees over the user data, especially regarding some scenarios useful for the company.

This document starts by presenting in chapter 2 state of the art regarding legislation, compliance with it, and technologies associated with data privacy and security. In section 2.1, some of the most important aspects of the GDPR are presented. Section 2.2 analyzes the current state of technologies and algorithms that ensure privacy and security are carried out. Then, several data exploration scenarios suitable for the company Altice/MEO are defined in chapter 3, being one of them chosen for implementation.

Chapter 4 describes some challenges regarding the chosen scenario, as well as possible solutions. Chapter 5 then presents the development decisions regarding the selected technologies and cryptographic algorithms, and describes the implemented system. The evaluation of this system's practicality is described in chapter 6, with the presentation and analysis of the respective results.

Finally, this dissertation's conclusion and prospects for future work are presented in chapter 7.

---

## STATE OF THE ART

---

This chapter describes the state of the art regarding ensuring personal data privacy and security, especially in terms of the existing data mining mechanisms.

To select the most appropriate mechanisms for different scenarios, an investigation about which guarantees must be given to customers at this level was carried out. So, the GDPR is the start point of this chapter.

Then, an investigation about how these guarantees can be assured to the user, especially during data processing, is carried out. This corresponds to the gathering of the main known PETs. For each PET, the concept, goals, advantages, disadvantages, maturity level, application phase, and application context are presented. Some additional information is also given in Annex C.

### 2.1 GENERAL DATA PROTECTION REGULATION

The General Data Protection Regulation 2016/679 (GDPR) is a regulation in European Union (EU) law on data protection and privacy for all individual citizens of the EU and the European Economic Area (EEA).

According to the EU GDPR Portal, this regulation was designed with the following main objectives:

- Simplify the regulatory environment for international business by harmonizing data privacy laws within the EU;
- Protect and empower all EU citizens' data privacy by giving them control over their personal data;
- Reshape the way organizations across the region approach data privacy.

This regulation applies to “personal data”, meaning any information regarding an identifiable person who can be directly or indirectly identified (e.g., by an identifier, name, identification number, location data, online identifier, or one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that person).

## 2.1. General Data Protection Regulation

The EU General Data Protection Regulation replaces the Data Protection Directive 95/46/EC. Although the key principles of data privacy still hold to the previous directive, many changes have been proposed to the regulatory policies. The key points of the GDPR, as perceived by EU GDPR Portal and ENISA – European Union Agency for Cybersecurity, are presented next:

- **Increased Territorial Scope** (extraterritorial applicability) – The GDPR has extended jurisdiction, as it applies to all companies processing the personal data of data subjects residing in the EU, regardless of the company's location. Processing is a broad term that covers just about anything that can be done with data: collection, storage, transmission, analysis, etc.

Previously, territorial applicability of the directive was ambiguous and referred to the data process “in the context of an establishment”. GDPR makes its applicability very clear.

The GDPR applies to the processing of personal data by controllers and processors (e.g., clouds) in the EU, regardless of whether the processing takes place in the EU or not. It also applies to the processing of personal data of subjects in the EU by a controller or processor not established in the EU, where the activities relate to offering goods or services to EU citizens, and monitoring of behavior that takes place within the EU. Non-EU businesses processing the data of EU citizens also have to appoint a representative in the EU. It is important to notice that these rules also apply to processors, which must sign adequate contracts with the controllers.

- **Penalties** – Organizations in breach of the GDPR can be fined up to 4% of annual global turnover or €20 Million (whichever is greater). This is the maximum fine that can be imposed for the most serious infringements (e.g., failing to comply with the basic principles for processing the consumer's personal data, including conditions for consent, or violating the core of Privacy by Design concepts).
- **Lawfulness** – This very basic principle of lawfulness is not internationally harmonized. While in several countries outside Europe, processing of personal data is permitted unless it is explicitly forbidden, in the EU, the processing is usually forbidden unless there is explicit permission, e.g., by the individual's consent or by statutory provisions. According to European data protection law, the processing of personal data is only allowed if:
  - a) the data subject has given consent to the processing of his or her personal data for one or more specific purposes;
  - b) processing is necessary for the performance of a contract which the data subject is or will be part;
  - c) for compliance with a legal obligation;

- d) to protect the vital interests of the data subject or another natural person;
  - e) for the performance of a task carried out in the public interest;
  - f) for legitimate interests pursued by the data processing entities, if such interests are not overridden by the data subject's fundamental rights and freedoms.
- **Consent** – To enable lawful data processing of individuals' personal identifiable information, individuals need to give specific, informed, and explicit indication of their intentions concerning the processing of their data. This means that the consent must be clear and distinguishable from other matters, as well as provided in an intelligible and easily accessible form, using clear and plain language. A declaration of consent is invalid if not all of these requirements are met. Hence, transparency is a prerequisite for consent.

Additionally, a data controller cannot refuse service to users who decline consent to processing that is not strictly necessary to use the service. Consent for children, defined in the regulation as being less than 16 years old (although with the option for member states to individually make it as low as 13 years old, such as the Portuguese case), must be given by the child's parent or custodian, and be verifiable.

Furthermore, it must be as easy to withdraw consent, with effect for the future, as it is to give it. Consent is related to the right to informational self-determination. It is an expression of the individuals' freedoms in general.

However, in practice, many individuals are not sufficiently informed, or the consent is not freely given. This situation can occur due to, for example, the asked consent being too complicated or the individuals' focus being on another topic at the moment that consent is requested. This observation has been made not only in the field of privacy and data protection with "take it or leave it" apps or contracts in *legalese*, but also when signing consent forms for medical measures or bank statements. Annex A explores this subject further, presenting a brief analysis of some unethical practices used by some companies to make it difficult for users to enforce their rights.

- **Purpose Binding** – Personal data obtained for one purpose must not be processed for other purposes that are not compatible with the original one. The purpose has to be legitimate, and it has to be specified and made explicit before collecting personal data. In many countries outside Europe, the principle of purpose limitation or purpose binding is unknown. Instead, it is encouraged to use data for multiple purposes. Big Data is one of the prominent trends that incorporates multi-purpose linkage and analysis of data instead of leaving them in separate domains.

- **Data Subject Rights** – Individuals have the right to access and rectify, as well as to block and erase their personal data. Further, they have the right to withdraw given consent with effect for the future. In case of a breach, data subjects have the right to be notified. They also have the right to data portability. These rights should be supported in a way that individuals can effectively and conveniently exercise their rights. Some specifications about these rights are presented next:
  - **Right to Access** – With the GDPR, data subjects have the right to obtain confirmation from the data controller as to whether personal data concerning them is being processed, where and for what purpose. Further, the controller shall provide a copy of the personal data, free of charge, in an electronic format. This change is a dramatic shift to data transparency and empowerment of data subjects.
  - **Right to be Forgotten or Data Erasure** – This right allows the data subject to have the data controller erase his/her personal data, cease further dissemination of the data, and potentially have third parties halt processing of the data. The conditions for erasure include the data no longer being relevant to the original purposes for processing or a data subject withdrawing consent. It should also be noted that this right requires controllers to compare the subjects' rights to "the public interest in the availability of the data" when considering such requests.
  - **Breach Notification** – Under the GDPR, breach notifications are now mandatory in all member states where a data breach is likely to "result in a risk for the rights and freedoms of individuals". This must be done within 72 hours of first having become aware of the breach. Data processors are also required to notify their customers, the controllers, "without undue delay" after first becoming aware of a data breach.
  - **Data Portability** – GDPR introduces the right for a data subject to receive his personal data, which he has previously provided in a "commonly use and machine-readable format". He also has the right to transmit that data to another controller.
- **Necessity and Data Minimization** – Controllers should collect, hold, and process only the data absolutely necessary for the completion of their duties, as well as limit the access to personal data to those needing to perform the processing. Consequently, personal data must be erased or effectively anonymized as soon as it is not needed anymore for the given purpose. Although data minimization at the earliest stage of processing is a core concept of Privacy Enhancing Technologies



(PETs) and has been mentioned explicitly in the Global Privacy Standard of 2006, it has not been well enforced yet.

- **Privacy by Design** – Privacy by design has existed as a concept since 1995 (Hustinx, 2010) and as a framework since 2009 (Cavoukian, 2009). GDPR incorporated this concept as a legal requirement. The principle of privacy or data protection by design is based on the insight that building in privacy features from the beginning of the design process is preferable over the attempt to adapt a product or service at a later stage. The involvement in the design process supports the consideration of the full data lifecycle. The 7 foundation principles of privacy by design are presented in Table 1. Some of these principles will be further explored next.

Table 1: Seven foundation principles for privacy by design.

<b>Foundation Principles</b>	<b>Privacy</b>	<b>Security</b>
1. Proactive, not Reactive; Preventative not Remedial	Anticipate and prevent privacy-invasive events before they happen. Do not wait for privacy risks to materialize.	Begin with the end in mind. Leverage enterprise architecture methods to guide the proactive implementation of security.
2. Privacy as the Default Setting	Build privacy measures directly into any given Information and Communications Technology (ICT) system or business practice, by default.	Implement “Secure by Default” policies, including least privilege, need-to-know, least trust, mandatory access control, and separation of duties.
3. Privacy Embedded into Design	Embed privacy into the design and architecture of ICT systems and business practices. Do not bolt it on after the fact.	Apply Software Security Assurance practices. Use hardware solutions such as a Trusted Platform Module.
4. Full Functionality – Positive-Sum, not Zero-Sum	Accommodate all legitimate interests and objectives in a positive-sum “win-win” manner, not through a zero-sum approach involving unnecessary trade-offs.	Accommodate all stakeholders. Resolve conflicts to seek win-win.

Table 1: Seven foundation principles for privacy by design (continuation).

<b>Foundation Principles</b>	<b>Privacy</b>	<b>Security</b>
5. End-to-End Security – Full Lifecycle Protection	Ensure cradle-to-grave secure lifecycle management of information.	Ensure confidentiality, integrity, and availability of all information for all stakeholders.
6. Visibility and Transparency – Keep it Open	Keep component parts of Information Technology (IT) systems and operations of business practices visible and transparent to users and providers alike.	Strengthen security through open standards, well-known processes, and external validation.
7. Respect for User Privacy – Keep it User-Centric	Respect and protect the interests of the individual, above all. Keep it user-centric.	Respect and protect the interests of all information owners. Security must accommodate both individual and enterprise interests.

- **Privacy by Default** – This principle means that the user is already protected against privacy risks in the default setting. This affects the choice of the designer about which parts are wired-in and which are configurable. In many cases, a privacy-respecting default would not allow an extended functionality of the product unless the user explicitly chooses to use it.
- **Information Security** – Information security addresses the protection goals of Confidentiality, Integrity, and Availability (CIA). These goals are also important from a privacy and data protection perspective that specifically requires that unauthorized access, processing, manipulation, loss, destruction, and damage are prevented. The data also has to be accurate. Moreover, the organizational and technical processes for appropriately handling the data and providing the possibility for individuals to exercise their rights have to be available whenever necessary. This principle calls for appropriate technical and organizational safeguards.
- **Accountability** – Accountability means to ensure and to be able to demonstrate compliance with privacy and data protection principles (or legal requirements). This requires clear responsibilities,

internal and external auditing, and controlling of all data processing. In some organizations, Data Protection Officers (DPOs) are appointed to perform internal audits and handle complaints. A means for demonstrating compliance can be a Data Protection Impact Assessment (DPIA) or Privacy Impact Assessment (PIA). On a national level, accountability is supported by an independent Data Protection Authority (DPA) for monitoring and checking as supervisory bodies.

- **DPO** – Under GDPR, it is not necessary to submit notifications/registrations of data processing activities to each local DPA, nor is it a requirement to notify/obtain approval for transfers based on the Model Contract Clauses (MCCs). Instead, there are internal record keeping requirements, as further explained below. Also, the DPO appointment is mandatory for those controllers and processors whose core activities consist of processing operations that require regular and systematic monitoring of data subjects on a large scale, or of special categories of data, or data relating to criminal convictions and offenses. Importantly, the DPO:
  - \* Must be appointed based on professional qualities and, in particular, expert knowledge on data protection law and practices;
  - \* May be a staff member or an external service provider;
  - \* Contact details must be provided to the relevant DPA;
  - \* Must be provided with appropriate resources to carry out their tasks and maintain their expert knowledge;
  - \* Must report directly to the highest level of management;
  - \* Must not carry out any other tasks that could result in a conflict of interest.
  
- **PIA** or **DPIA** – It was incorporated by the GDPR and refers to the controller's obligation to conduct an impact assessment and to document it before starting the intended data processing. It must be done when data processing could result in a high risk to natural persons' rights and freedoms (intersoft consulting, 2018).

Summarizing, the main differences between the previous data protection directive 95/46/EC and the GDPR are presented in Table 2.

Table 2: Differences between the data protection directive 95/46/EC and the GDPR.

<b>Data Protection Directive 95/46/EC</b>	<b>GDPR</b>
European reach only	Global reach
Local law divergence across 28 EU states	Regulation: uniform across EU
Multiple DPA exposure	One-stop-shop
Limited accountability	Accountability key
Controllers only	Controllers and Processors
Small penalties that differ between countries	Huge penalties
No obligation to report breaches	Obligated to report breaches without delay
No obligation to have DPO	DPO required for large organizations

In order to understand, from a legal perspective, which of these guarantees are given by big companies, including Altice, an analysis and comparison of the privacy policies of Google and Altice were carried out. This information can be found in Annex B. This analysis reveals these companies' concerns regarding compliance with the GDPR and ensuring user's privacy, emphasizing the need to use privacy and security mechanisms during data processing.

## **2.2 PRIVACY ENHANCING TECHNOLOGIES**

The idea of shaping technology according to privacy principles has been discussed for many years, addressing, among others, the principles of data minimization, anonymization, and pseudonymization (ENISA – European Union Agency for Cybersecurity, 2016b). This led to the term Privacy Enhancing Technologies (PETs), which covers the broader range of technologies and approaches (from a piece of tape masking a webcam to advanced cryptographic techniques) designed to help achieve compliance with data protection legislation or privacy policies, like the GDPR.

The purpose of PETs is to allow online users to protect the privacy of their Personally Identifiable Information (PII) provided to and handled by services or applications. While cybersecurity is focused on protecting data so other people cannot access it, PETs concentrate on enabling the derivation of useful results from data without giving other people access to all of the data (The Royal Society, 2019).

PETs have different maturity or readiness levels, which are defined by ENISA – European Union Agency for Cybersecurity (2018) as follows:

- Idea (lowest readiness level) – The PET has been proposed as an idea in an informal fashion, e.g., written as a blog post, discussed at a conference, described in a white paper or technical report.
- Research – The PET is a serious object of rigorous scientific study. At least one, preferably more, academic paper(s) have been published in the scientific literature, discussing the PET in detail and at least arguing its correctness, as well as security and privacy properties.
- Proof-of-concept – The PET has been implemented and can be tested for specific properties, such as computational complexity, protection properties, etc. This means that “running code” is available, but no actual application of the PET exists in practice, involving real users, nor is the implementation feature complete.
- Pilot – The PET is or has recently been used in practice in at least a small-scale pilot application with real users. The application scope and the user base may have been restricted.
- Product (highest readiness level) – The PET has been incorporated in one or more generally available products that have been or are being used in practice by a significant number of users. The user group is not *a priori* restricted by the developers.
- Outdated – The PET is not used anymore, e.g., because the need for the PET has elapsed, because it is dependent on another technology that is not maintained anymore, or because there are better PETs that have superseded that one.

The efficacy of a PET can be assessed based on whether sensitive information is effectively protected. In order to make this assessment, it is essential to consider who the attacker might be and what prior information they might have before the solution is deployed.

According to Denny et al. (2014), some feel that they are protecting privacy just by using PETs. Although this can be partially true, it is not entirely true. Even if the design is full of PETs, privacy will not be fully protected without well-written policies, standards, procedures, guidelines, and information about the data processing presented in an intelligible form, among other things. PETs are enablers, but they are not substitutes for privacy engineering. PETs can be just one of many design components but alone are not a privacy solution.

Some specific examples of PETs that seem promising to enable privacy-aware data collection, analysis, and dissemination of results, as well as privacy-preserving computation, are presented next.

### 2.2.1 Anonymization and Pseudonymization

**CONCEPT** Anonymization is defined by ENISA – European Union Agency for Cybersecurity (2015) as the process of modifying personal data so that individuals cannot be re-identified and no information about them can be learned. Pseudonymization is a method to replace identifiable or sensitive data with one or more reversible, consistent, and artificial identifiers, or pseudonyms (GDPR Report, 2017). There can be a single pseudonym for a collection of replaced fields or a pseudonym per replaced field.

**GOALS** Protect data classified as personal identifiable data, sensitive personal data, or sensitive commercial data, which can be accomplished by preventing re-identification, attribute disclosure, and/or the linkage of records corresponding to the same individual (Personal Data Protection Commission Singapore – PDPC, 2018). Pseudonymization only prevents re-identification by people who don't have access to additional secret information.

**TECHNIQUES** According to Personal Data Protection Commission Singapore – PDPC (2018), there are several anonymization and pseudonymization methods that can be used for different purposes. All these methods entail some information loss (data utility loss) in the resulting dataset. Some anonymization techniques are presented next:

- Attribute Suppression – Refers to the removal of an attribute to all records in a dataset.
- Record Suppression – Refers to the removal of an entire record in a dataset.
- Nulling Out or Deletion – Particular fields become *nulls* to anyone whose access is not authorized.
- Character Scrambling – The characters are jumbled into a random order.
- Character Masking – Change of the characters of a data value, e.g., by using a constant symbol, like “\*” or “x”. Masking is typically partial, i.e., applied only to some characters in the attribute.
- Pseudonymization or Coding – The replacement of identifying data with made-up values. As an anonymization technique, pseudonyms must be irreversible, meaning that the original values are disposed of properly. The main pseudonymization techniques are specified below – as anonymization techniques, the identity database or the key, according to the used method, are disposed of properly.
- Generalization or Recoding – A deliberate reduction in the data precision.
- Swapping, Shuffling, or Permutation – Rearrange of the data in the dataset such that the individual attribute values are still represented but generally do not correspond to the original records.

- Number and Date Variance – It consists of adding or subtracting a percent, number, or date to the original data, maintaining that data meaningfulness.
- Data Perturbation – The values from the original dataset are modified to be slightly different.
- Synthetic Data – Mainly used to generate synthetic datasets directly and separately from the original data, instead of modifying the original dataset.
- Data Aggregation – Convert a dataset from a list of records to summarized values.

In pseudonymization, pseudonyms are reversible (by the original data owner), which means that the original values are securely kept but can be retrieved and linked back to the pseudonym if the need arises. The main pseudonymization methods are:

- Pseudonym generators – Pre-generates a list of made-up values and randomly selects values from this list to replace the original ones. The made-up values should be unique and should have no relationship to the original ones. The identity database, which keeps the correlation between pseudonyms and original data, cannot be shared with the pseudonymized database recipient: it should be securely held, and only the organization can use it to resolve any specific queries (in controlled quantity).
- Encryption – It entails the use of a key to encode or protect a dataset. Consequently, encryption is mathematically reversible and subject to the complexities of key management. The encryption key cannot be shared and must be securely protected from unauthorized access, because the leak of such a key could result in a data breach by enabling the encryption's reversal.

In some cases, pseudonyms may need to follow the structure or data type of the original value. In such cases, special pseudonym generators may be considered to create synthetic datasets, or the so-called Format-Preserving Encryption (FPE) to create pseudonyms with the same format as the original data.

An accurate analysis of the disclosure risk is needed before any information release. Some measures can be considered to ensure that a certain risk threshold has not been surpassed, such as *k*-anonymity and differential-privacy, both addressed in more detail later on.

**APPLICATION PHASE** Addresses privacy at the data collection stage or when disclosing results. In the first case, as anonymization is made at an early stage of the data lifecycle, it is not possible to optimize it to a specific analysis.

**APPLICATION CONTEXT** Anonymization is useful in several security scenarios. Sometimes, the data doesn't need to look real or appear consistent. Other times, those are needs, because the data must

## 2.2. Privacy Enhancing Technologies

remain usable for different purposes. According to Watts (2018), some of the top reasons why enterprise businesses use anonymization are:

- Protect data from third-party vendors, such as marketers, consultants, and others;
- Minimize the impact of data breaches that are the result of an operator error;
- Perform operations where some data does not need to be real, such as conducting application tests.

Pseudonymization can be used for those same purposes if there is the need to recover the original data after the processing or analysis. It can also be one way to comply with the GDPR demands for secure storage of personal information.

ADDITIONAL INFORMATION Annex C.1.

### ***k*-Anonymity and its extensions**

CONCEPT *k*-Anonymization is an approach where certain variables in a dataset are suppressed or generalized in a manner that ensures the information from any single individual cannot be distinguished from at least  $k - 1$  other individuals in the dataset (The Royal Society, 2019). More technically, a dataset is said to satisfy *k*-anonymity for  $k > 1$  if there are at least  $k$  records sharing each combination of Quasi-Identifier (QI) attribute values in the dataset (ENISA – European Union Agency for Cybersecurity, 2015). This means that identity disclosure is limited to that group of  $k$  records.

ADVANTAGES

- For datasets with low to moderate dimensions (number of attributes), these anonymization methods can protect against re-identification based on crossing databases (ENISA – European Union Agency for Cybersecurity, 2014).
- *k*-Anonymity offers linkability at the group level, among groups of  $k$  records (ENISA – European Union Agency for Cybersecurity, 2015).

DISADVANTAGES

- *k*-Anonymity assumes that each record pertains to a different individual (Personal Data Protection Commission Singapore – PDPC, 2018). If the same individual has multiple records (e.g., visiting the hospital on several occasions), then *k*-anonymity will need to be higher than the repeated records.



Otherwise, the records may be not only linkable but also re-identifiable, despite seemingly fulfilling  $k$  equivalence classes.

- $k$ -Anonymity is not composable. This means linking two  $k$ -anonymous datasets is not guaranteed to yield a  $k'$ -anonymous dataset for any  $k' > 1$  (ENISA – European Union Agency for Cybersecurity, 2015). For example, if we have two  $k$ -anonymous datasets of patients from two different hospitals (e.g., including zip code, age range, and disease of the patient), it would be possible to identify a certain individual in the set if someone has information that this individual visited both hospitals, and his/her age and location are known. Overall  $k$ -anonymity cannot guarantee privacy if sensitive values in the dataset lack diversity and some further information is known to the attacker.
- For high-dimensional datasets,  $k$ -anonymity-like methods are hardly usable. A new class of statistical de-anonymization attacks against high-dimensional datasets was presented and illustrated on the Netflix Prize data (Narayanan and Shmatikov, 2008). This dataset contains anonymous movie ratings of 500 000 subscribers of Netflix. The attack showed that knowing a little bit about an individual subscriber is enough to find his record in the dataset. In such high-dimensional datasets, it is hard to split attributes between Quasi-Identifiers and confidential.  $\epsilon$ -differential privacy, which will be described next, could be used to prevent re-identification, but at the cost of a considerable utility loss.
- As big data becomes the norm rather than the exception, there is an increasing dimensionality of data, as well as more public datasets that can be used to aid re-identification efforts (Morland, 2017).

**MATURITY LEVEL** Despite the specified limitations,  $k$ -anonymity is one of the most accepted models for privacy in real-life applications and provides the theoretical basis for privacy-related legislation (Friedman et al., 2007). Regarding a priori anonymization, software offering it with  $k$ -anonymity (and some of its variants) guarantees is freely available, e.g., ARX. In what respects a posteriori anonymization, the official statistics community has produced some free software packages for microdata protection, such as  $\mu$ -ARGUS and sdcMicro (Danezis et al., 2014; Prasser et al., 2020).

**ADDITIONAL INFORMATION** Annex C.2.

### **Differential privacy and related models**

**CONCEPT** Differential privacy is a privacy model that seeks to limit the impact of any individual subject's contribution on the analysis outcome (ENISA – European Union Agency for Cybersecurity, 2015). This security definition means that when a dataset, result, or statistic is released, it should not give much more information about a particular individual than if that individual had not been included in the dataset. The introduction of differential privacy has provided a new approach to releasing statistical information on datasets in a privacy-preserving manner, together with mathematical proof (The Royal Society, 2019).

**ADVANTAGES**  $\epsilon$ -differential privacy is strongly composable: combining an  $\epsilon_1$ -differentially private dataset and an  $\epsilon_2$ -differentially private dataset yields an  $\epsilon_1 + \epsilon_2$ -differentially private dataset. Hence, differential privacy is still satisfied, although with a less strict parameter. Regarding linkability, differentially private datasets are not linkable if obtained using noise addition, but they can be made linkable if obtained using partially synthetic data generation (ENISA – European Union Agency for Cybersecurity, 2015). In general,  $\epsilon$ -differential privacy provides more strict privacy guarantees than  $k$ -anonymity and its extensions.

**DISADVANTAGES** Using differential privacy to maximize learning while providing a meaningful degree of privacy requires judicious choices concerning the privacy parameter  $\epsilon$  and careful selection of other factors (Dwork and Kohli, 2019). Today, there is little understanding of the optimal value of  $\epsilon$  for a given system or class of systems, purposes, data, ... There is no clear consensus on how to choose  $\epsilon$ , nor agreement on how to approach this and other critical implementation decisions. Typically,  $\epsilon$ -differential privacy also entails more utility loss than  $k$ -anonymity-like models.

**MATURITY LEVEL** Some academic systems implement general-purpose data processing systems with differential privacy guarantees. Differential privacy has also been piloted by tech companies and governmental organizations that hold large datasets, where differentially private mechanisms are most likely to yield both a useful and privacy-preserving result (The Royal Society, 2019).

For example, the Microsoft Research PINQ project defines queries in a LINQ-like syntax, executes them, and protects the results using a differentially private mechanism. The Airavat system uses the sample-and-aggregate differentially private mechanism to protect map-reduce query results using differential privacy. Other systems include Fuzz (University of Pennsylvania) and GUPT (University of California, Berkeley), as mentioned in Danezis et al. (2014). Apple and Google also implemented their own versions of distributed differential privacy for collecting statistics from end-users (The Royal Society, 2019). However, most statis-

ticians in industry and government rely on existing software, and such legacy systems mean that they cannot implement a sophisticated approach such as differential privacy in a straightforward manner.

ADDITIONAL INFORMATION Annex C.3.

### **2.2.2 Privacy-Preserving Record Linkage**

CONCEPT Privacy-Preserving Record Linkage (PPRL) supports the matching and integration of person-related data without compromising privacy (Franke et al., 2019). It is based on the encoding of sensitive attribute values needed for matching and often requires trusted parties for linkage.

GOALS Record linkage aims to link records that refer to the same real-world person (Sehili et al., 2018).

ADVANTAGES In many situations, record linkage is an efficient way to collect and correlate data. The benefits of combining data from different sources have already been demonstrated (Ariel et al., 2014). For example, it allows answering research questions that are very difficult to answer using data from just one source. Record linkage can also reduce the inconvenience of asking sensitive questions.

DISADVANTAGES The challenge in record linkage is to link records that belong to the same individual from different sources. Missed links lead to the same problems as nonresponse in surveys. If certain groups of individuals are more difficult to link, estimations could be biased. Similarly, incorrect links, defined as combining the information of two different persons into one record, lead to errors similar to measurement error. The quality of linkage procedures and the datasets' reliability are difficult to determine, which constitutes a significant issue in record linkage (Ariel et al., 2014).

MATURITY LEVEL There are already some products available, such as a suite of technologies known as Anonlink, that allows two organizations to carry out private record linkage, i.e., to find matching records between their datasets without disclosing Personally Identifiable Information (Data61, 2017).

However, there is still research on several components of this technique (Sehili et al., 2018). In the last few years, there has been a focus on improving the performance and scalability of PPRL methods. For future work, it is planned to simplify the use of PPRL by investigating approaches to automatically find suitable parameter settings, as well as investigate multi-party PPRL approaches with more than two sources. These approaches often require finding subset matches, i.e., sets of matching records that are only in a subset of the sources. Finally, the integration of PPRL into practical applications and use cases continues.

## 2.2. Privacy Enhancing Technologies

APPLICATION PHASE Addresses privacy when disclosing results.

APPLICATION CONTEXT Typically, there is a lack of global identifiers. Therefore, the linkage can only be achieved by comparing available QIs, such as name, address, or birth date (Sehili et al., 2018). However, in many cases, data owners are only willing or allowed to provide their data for such integration if there is enough protection of sensitive information to ensure data subjects' privacy. PPRL addresses this problem by providing techniques to match records while preserving their privacy, allowing the combination of data from different sources for improved data analysis and research. PPRL approaches can be applied in many areas, such as public health surveillance, demographical studies, and marketing analysis.

ADDITIONAL INFORMATION Annex C.4.

### 2.2.3 Privacy-Preserving Storage

Storage privacy refers to the ability to store data without anyone being able to read (let alone manipulate) it, except the party who stored the data (data owner) and whoever he authorizes. If the data owner stores the data locally, then physical access control might help, but it is not enough if the computer is connected to a network: a hacker might succeed in remotely accessing the stored data. If the data owner stores it in the cloud, then he cannot control physical access to the data. Hence, technology-oriented countermeasures are needed, such as:

- Operating system controls – User authentication and access control lists managed by the operating system.
- Local encrypted storage – Encryption is a fundamental security technique, which transforms data in a way that only authorized parties can read it. It is a strong protection measure for personal data. Full disk encryption or filesystem-level encryption, for example, can be used to achieve this purpose.
- Format-preserving encryption – It encrypts a plaintext of some specified format into a ciphertext of identical format (e.g., encrypting a valid credit card number into a valid credit card number).
- Steganographic storage – Steganographic file systems allow storing private information while hiding its very existence. This mechanism gives the user a very high privacy level: the data are steganographically embedded in cover data.
- Secure remote storage – Encrypted and steganographic storage can also be used to achieve private storage in remote settings, such as cloud storage.

While storing encrypted data locally or remotely is a good way to achieve private storage, it implies sacrificing functionality to security and privacy. For example, the search for documents with a keyword would have the overhead of download and decrypt time. Big data analytics need to allow for search and other computations over the stored data without decrypting it (which contradicts the very idea of encryption). The ability to search on encrypted data is very attractive, especially in cloud systems, as the remote user can delegate the search to those systems, which usually have more computing power and hold the encrypted data locally.

### **Traditional Encryption**

**CONCEPT** Encryption is the usage of a mathematical function, with a secret value (the key), to encode data so that only users with access to the decryption key can read the information (Information Commissioner's Officer, 2018). There are two types of encryption schemes: symmetric and public-key (or asymmetric). Symmetric encryption uses the same key to encrypt and decrypt the ciphertext, while public-key encryption uses the concepts of a public and a private key.

Because this technology does not allow the derivation of useful results from the encrypted data, traditional encryption is not strictly a PET. However, it will be further explored to introduce the concept of encryption, which is related to other PETs presented later.

**GOALS** Encryption provides an appropriate safeguard against the unauthorized or unlawful access and processing of personal data (Information Commissioner's Officer, 2018).

**ADVANTAGES** Encryption main advantages are (SpamLaws, 2009):

- Confidentiality – A lot of organizations are required to meet specific confidentiality requirements and other associated regulations. Encrypting data means that it can only be accessible by recipients who have the decryption key.
- Limited Impact of Data Breaches – Data encryption ensures the protection of intellectual property and other similar data types. So, it minimizes the potential complications that accompany data breaches.
- Separation – Data encryption allows the data to remain separate from the device security where it is stored. Because the encryption is on the data itself, the data is secure regardless of how it is stored or transmitted. In other words, security is included with the encryption, which allows the storage and transmission of data via unsecured means.

## 2.2. Privacy Enhancing Technologies

Another advantage, specific to symmetric cryptography, is related to quantum computation, which is a new phenomenon. While modern computers store data using a binary format called a “bit”, in which a “1” or a “0” can be stored, a quantum computer stores data using a quantum superposition of multiple states. These multiple-valued states are stored in “quantum bits” or “qubits”. This allows the computation of numbers to be several orders of magnitude faster than traditional transistor processors. Numbers that would typically take billions of years to compute could only take a matter of hours or even minutes with a fully developed quantum computer. Most current symmetric cryptographic algorithms and hash functions are considered relatively secure against attacks by quantum computers. Generally, doubling the key size is enough to keep the same security level in quantum computers as in the conventional ones.

**DISADVANTAGES** Encryption main disadvantages are (SpamLaws, 2009; tutorialspoint, 2015):

- **Encryption Keys** – Without a doubt, data encryption is a monumental task for an IT specialist. The more data encryption keys there are, the more difficult IT administrative tasks for maintaining all of them can be. If the encryption key is lost, the data associated with it is also lost.
- **Not Enough** – Encryption (or cryptography) by itself does not guard against vulnerabilities and threats that emerge from the poor design of systems, protocols, and procedures. These need to be fixed through proper design and the set up of a defensive infrastructure.
- **Costly** – Data encryption comes at a cost in terms of time (performance loss) and money. Regarding the monetary cost, the systems that maintain data encryption must have the capabilities to perform such tasks. A financial budget is also required to set up and maintain the public key infrastructure used with public-key schemes.
- **Compatibility** – Data encryption technology can be tricky when layering it with existing programs and applications. This can negatively impact routine operations within the system.
- **Security** – Usually, the security of public-key encryption techniques is based on the computational difficulty of mathematical problems. Any breakthrough in solving such problems or increasing the computation power can render an encryption technique vulnerable.

Regarding quantum computation, modern public-key cryptography will have to look for computationally harder problems or devise completely new techniques to achieve the goals it presently serves.

**MATURITY LEVEL** Different encryption algorithms can be included in different maturity levels. For example, Rivest Cipher 4 (RC4) and Data Encryption Standard (DES) algorithms are outdated because they were broken or more secure algorithms came up. The algorithms Salsa20, ChaCha, Advanced Encryp-

tion Standard (AES), Diffie-Hellman (DH), and Rivest-Shamir-Adleman (RSA), on the other hand, are still commonly used by several products. There are also several Elliptic Curve Cryptography (ECC) implementations available. At last, quantum cryptography (the science of exploiting quantum mechanical properties to perform cryptographic tasks) is still in the research phase.

APPLICATION PHASE Encryption can be applied to data transfer or storage.

APPLICATION CONTEXT Encryption has long been used by militaries and governments to facilitate secret communication. It is now commonly used to protect information within many kinds of civilian systems (Wikipedia, the free encyclopedia, 2017a).

Encryption can be used to protect data at rest, such as information stored on computers and storage devices, and data in transit, such as data transferred via networks, mobile telephones, wireless microphones, wireless intercom systems, Bluetooth devices, and bank automatic teller machines. Sensitive or private data must be encrypted when transmitted across networks to protect it against the eavesdropping of network traffic by unauthorized users.

ADDITIONAL INFORMATION Annex C.5.

### **Attribute-Based Encryption**

CONCEPT Attribute-Based Encryption (ABE) combines access control with public-key cryptography, in a way that both the user's secret key and the ciphertext depend upon certain attributes (e.g., the individual's country, job, or habit). The decryption of the ciphertext is possible only if the set of attributes of the user's key matches the ciphertext attributes (ENISA – European Union Agency for Cybersecurity, 2015).

GOALS Share data among different user groups while preserving users' privacy.

ADVANTAGES Besides the traditional encryption advantages (confidentiality, limited impact of data breaches, and separation), ABE also allows flexible access control over the encrypted data. Any user who possesses a certified attribute set that matches the ciphertext's attribute set can decrypt the data using its own private key, without interactions with the data owner (Hoang et al., 2019).

DISADVANTAGES

- Space and Computational Cost – Both the size of the produced ciphertexts and the computational cost grow linearly with the complexity of the access policy or the number of attributes, which brings pressure for users with limited computing resources, like mobile-device users (Hoang et al., 2019).

## 2.2. Privacy Enhancing Technologies

Some ABE schemes with constant-size ciphertexts have been proposed to solve this problem, but they have some restrictions regarding access policies (Zhang et al., 2019; Tao et al., 2019). Several ABE schemes with outsourcing computing have also been proposed (Liu et al., 2018).

- Scalability – Some ABE schemes fail to perform with a large number of users and hierarchical relationships among them, while others directly address this problem (Sethi et al., 2019).
- Key Escrow – ABE requires the existence of an assumed-trusted authority to generate and distribute the users' secret keys. When this is a single authority, it holds each user's attribute key and has the power to decrypt every ciphertext (Zhang et al., 2018). There is, for example, a scheme that relies on the interpolating polynomial to prevent the authority from abusing the members' private keys (Zhou et al., 2019). Multi-Authority Attribute-Based Encryption (MA-ABE) is also an effective way to solve the key escrow problem (Zhang et al., 2019).
- Revocation – ABE lacks efficient user and attribute revocation mechanisms. Users may share some common attributes which could change over time. Therefore, without an effective revocation mechanism, revoking a certain user or some of its attributes may result in the revocation of others in the system. There are already several schemes proposed to solve this problem, but they usually entail some computational overhead (Hoang et al., 2019).
- Search Mechanism – Traditional ABE fails to provide an efficient keyword-based search on encrypted data, which somewhat weakens the power of this encryption technique, as search is usually the most important approach to quickly obtain data of interest from a large-scale dataset (Yin et al., 2019). There is a solution proposal for this problem that combines ABE technology with Searchable encryption (presented below).

**MATURITY LEVEL** Currently, no ABE scheme addresses all referred disadvantages. However, it is possible to find some combinations that might be useful for practical applications.

For example, there is a recent decentralized multi-authority attribute-based cryptosystem proposal, with white-box traceability, outsourcing decryption, and policy update for access control in the cloud (Sethi et al., 2020). The results show that this cryptosystem performs reasonably well in terms of execution time with varying attributes and parameters of attribute authorities. There is also a proposed multi-authority attribute-based encryption scheme with constant-size ciphertexts and user revocation for threshold access policy (Zhang et al., 2018). Another example is an existing proposal for a so-called Enhanced Ciphertext-Policy Attribute-Based Encryption (ECP-ABE) scheme, which reduces the storage and revocation costs, as well as improves computational performance (Venkata Rao et al., 2020). Regarding quantum security,



some proposed schemes already address this issue, especially lattice-based constructions (Dong et al., 2020).

So, there are already some ABE schemes with practical applications proposed and implemented.

APPLICATION PHASE ABE can be applied to data transfer or storage.

APPLICATION CONTEXT In recent years, ABE has been increasingly applied for supporting flexible and fine-grained access control of sensitive data in mobile computing, cloud computing, social networks, and IoT. Some more concrete application scenarios are Pay-TV, e-Health, Cloud Storage, among others.

ADDITIONAL INFORMATION Annex C.6.

## 2.2.4 Privacy-Preserving Storage and Computation

While a large amount of data is being generated and used for driving novel scientific discoveries, the effective and responsible usage of that data remains a big challenge (Chen et al., 2019). This issue might be alleviated by outsourcing data to public cloud service providers with intensive computing resources. However, the privacy and security problem of the outsourced data and its analysis remains.

In the past few years, significant progress has been made on cryptographic techniques for secure computation. Among these techniques, Searchable Encryption can be used to search over encrypted data, for example. Homomorphic Encryption and Multi-Party Computation have also received increasing attention for secure computation due to technical breakthroughs.

### **Searchable encryption**

CONCEPT Searchable Encryption (SE) is a cryptographic technique that allows data users to search over encrypted data without decrypting it (Varri et al., 2019). So, it enables semi-trusted environments to retrieve ciphertexts while preserving the privacy of data and search query.

GOALS Securely store and retrieve data from a semi-trusted environment, which can search directly over the encrypted information (Wang et al., 2020).

ADVANTAGES Besides the traditional encryption advantages (confidentiality, limited impact of data breaches, and separation), SE schemes also allow secure and efficient search over encrypted data. The traditional data encryption technique hides the contents of the data and thus effectively protects data confidentiality. However, it doesn't allow, for example, to search for the ciphertexts containing certain keywords. It would

require the user to fetch all data, decrypt it, and then search for the needed information, which is not an efficient solution. SE solves this problem, allowing the user to authorize a third-party data storage server to retrieve his/her ciphertexts without leaking either what is searched for or the contents of the searched data (Lu et al., 2019).

### DISADVANTAGES

- Query expressiveness – Most SE schemes proposed have poor expressive ability. Although several more expressive SE schemes were proposed, they entail high computation and communication costs (Shen et al., 2020). So, there is a need for further research on other search mechanisms for richer query types, such as subset or range queries.
- Quantum Security – Most existing SE schemes are based on conventional cryptographic hardness assumptions, which are vulnerable to adversaries equipped with quantum computers in the near future. In this case, lattice-based cryptography attracts a lot of attention because it is secure against quantum attacks. Although some recent schemes have been proposed to query over encrypted data based on lattice assumptions, these constructions still have reduced query expressiveness. Hence, new schemes must be built considering quantum attacks (Wang et al., 2020).
- Secure dynamic search – Most methods proposed so far use the concept of index search. The addition of new keywords or removal of old ones from an already constructed index is not possible in static SE schemes without re-indexing the entire data. Dynamic schemes address this issue, allowing users to perform dynamic update operations on already outsourced data, but they usually entail more information leakage than static schemes (Varri et al., 2019). Recent attacks exploiting this leakage problem is driving the rapid development of new SE schemes revealing less information while performing updates, which are known as forward and backward private SE schemes. However, those advanced schemes reduce SE's efficiency, especially in the communication cost between the client and the server. There is also a proposal for a more efficient search method that doesn't include any index construction but can only perform basic search operations over multiple conditions. So, the construction of secure dynamic schemes is still an area of study.
- Key-exposure – Most existing SE schemes suffer from key-exposure problems. Once a data receiver's private key is compromised, adversaries may be able to reveal the contents of trapdoors that the data receiver previously submitted and further violate the confidentiality of the outsourced data. In an IoT scenario, for example, these problems become more evident once there is a substantial usage of mobile intelligent terminal devices with limited key protection. There are already

scheme proposals that incorporate a key update mechanism to resist key-exposure (Zhang et al., 2019a).

- Verifiability – Besides data privacy, data integrity is also important to make SE schemes more secure. Many authors proposed different verifiable schemes to check if the cloud results aren't correct, but with the sacrifice of essential functionalities like dynamic data updates and some critical search functionalities (Varri et al., 2019). Moreover, the verification cost should also be nominal and affordable to users regardless of the large data collection. Hence, verifiable search schemes with minimal cost and without sacrificing essential functionalities are still needed.
- Deployment difficulties – SE schemes are rarely deployed because most of them require database modifications. There is a proposal for an SE scheme that can work natively with off-the-shelf databases and supports wildcard-based pattern search on encrypted data (Chung et al., 2019). Although in this scheme there isn't the possibility of obtaining theoretical indistinguishability between indexed items, it does provide adequate confidentiality protection and performs much better than other existing wildcard SE schemes in terms of query performance, storage overhead, and deployment complexity.

A disadvantage specific to Symmetric Searchable Encryption, an SE variation described in Annex C.7, is the key distribution and management problem. This problem occurs due to the property of symmetric cryptography. These schemes need an extra secure channel to share the private key among owners and users. However, there is no guarantee that the secure channel is not compromised (Lu and Li, 2019). Besides that, when a user is revoked, the data owner needs to re-encrypt the data with a fresh key and distribute the new key to the rest of the legitimate users.

Additionally, there are other disadvantages specific to Public Searchable Encryption (SE variation described in Annex C.7):

- Performance – The most computationally expensive part of Public Searchable Encryption (PSE) is to retrieve target data from the entire outsourced database, as the storage server is required to perform a test algorithm for each keyword over the database during the search process (Zhang et al., 2019a). Most existing PSE schemes introduce a significant end-to-end computation delay due to costly public-key cryptographic operations. So, these schemes are confronted with a performance bottleneck on the storage server side. There are some schemes with better performance, usually entailing other disadvantages, such as reduced query expressiveness.

## 2.2. Privacy Enhancing Technologies

- Certificate management – Several PSE schemes are constructed based on the Public-Key Infrastructure (PKI) cryptosystem, which entails the need for certificate management. There have been some PSE scheme proposals based on a new definition of Certificateless Public Key Cryptography (CL-PKC), which removes the need to use certificates (Ma et al., 2020).
- Multi-user environments – The key management is taken care of by the data owner, which needs to be online every time a new user registers in the system. Also, user authentication, user access control, and user revocation are difficult operations in PSE schemes. The basic solution to share data between several users is to generate a different ciphertext for each one of them, thus leading to an increase in the database's size proportionally to the number of recipients, an increase in the network traffic, higher computational costs on the client-side, and the replication of encrypted data, which could affect security. However, several PSE variations aiming to solve these problems have been proposed (Varri et al., 2019; Al-Maytami et al., 2020).
- Variations problems – In the PSE domain, there are also the Identity-Based Searchable Encryption (IBSE) and Attribute-Based Searchable Encryption (ABSE) variants, which contributed with security metrics such as user revocation and fine-grained access control. However, these solutions respectively inherit Identity-Based Encryption (IBE) and ABE issues (Varri et al., 2019).
- Security threats – Most PSE schemes are subject to great security threats, such as Keyword Guessing Attacks (KGAs). Such attacks are based on the observation that keywords are always selected from a small space, and receivers usually use well-known keywords for searching files. There are some PSE schemes with resistance against KGAs (Qin et al., 2020; Zhang et al., 2019a,b).

**MATURITY LEVEL** Currently, no SE scheme addresses all referred disadvantages. There is still a lot of research about different aspects of this technology. However, it is possible to find some variations that might be useful for practical applications, especially if the queries to perform aren't very complex.

For example, there is a proposal for a fine-grained authorized keyword secure search scheme (Yin et al., 2020), which supports not only privacy-preserving keyword-based search over encrypted data but also flexible and fine-grained data privilege control (by leveraging ABE). Moreover, this scheme can achieve fine-grained search permission update with very small communication and computation cost. It can also flexibly perform disjunctive, conjunctive, and threshold operations among attributes. Extensive experiments demonstrate the correctness and practicality of this scheme.

Besides keyword-based searches, the execution of range queries in a database is also useful. A simple solution is to encrypt the database's contents using an Order-Preserving Encryption (OPE) scheme, which

is an encryption scheme that supports comparisons over encrypted values. However, Naveed et al. (2015) has shown that databases encrypted with OPE are extremely vulnerable to inference attacks. There is also a related primitive called Order-Revealing Encryption (ORE), which is a generalization of OPE that allows for stronger security. There are already some practical ORE schemes available (Lewi and Wu, 2016).

As technology advances, blockchain is attaining more attention from the SE approach (Varri et al., 2019). Some proposals used blockchain technology in SE to offer more security guarantees. Nevertheless, additional studies and experiments are required to develop more mature schemes in the future.

So, there are already some SE schemes with more simple practical applications proposed, but there are still many research areas regarding this encryption technique.

APPLICATION PHASE SE can be applied to data storage and computation.

APPLICATION CONTEXT SE schemes have important application value (Lu et al., 2019), which can be employed by many practical applications, such as encrypted email systems, cryptographic cloud storage, encrypted audit logs, Internet of Things, electronic health/medical systems, among others.

ADDITIONAL INFORMATION Annex C.7.

### **Homomorphic encryption**

CONCEPT Homomorphic Encryption (HE) is an encryption scheme that allows for computations on ciphertexts, with the result, when decrypted, matching the result of the corresponding operations on the plaintexts (Ma et al., 2018). So, a third party can process the encrypted data without any key, and the process itself does not reveal any original information. The user with the key can decrypt the processed data and get exactly the plaintext processing result.

GOALS Enable industry and government to provide capabilities for secure outsourced computation (Homomorphic Encryption Standardization, 2019).

ADVANTAGES Besides the traditional encryption advantages (confidentiality, limited impact of data breaches, and separation), HE schemes provide the following ones:

- Secure computation over encrypted data – HE allows securely computing over encrypted data in a third-party server. Fully Homomorphic Encryption (FHE) can even execute arbitrary computation on ciphertexts, facilitating the construction of programs for any desirable functionality (Kalyani and Chaudhari, 2019).

## 2.2. Privacy Enhancing Technologies

- Packing techniques – It is possible to encrypt multiple plaintext values into a single packed ciphertext and use the Single Instruction, Multiple Data (SIMD) techniques to perform operations on these values in parallel. Hence, HE schemes with packing techniques have good amortized complexity per plaintext value. They have been applied, for example, to privacy-preserving big data analysis (Chen et al., 2019).
- Quantum security – In recent years, it was shown how to construct FHE schemes based on standard cryptographic assumptions, including those assumed secure against quantum adversaries (Brakerski, 2018). In particular, it was shown that FHE schemes could be based on the hardness of the Learning with Errors (LWE) problem. LWE was proven to be as hard to solve as finding approximate shortest vectors in arbitrary worst-case lattices, a task for which no significant quantum speedup is known.

### DISADVANTAGES

- Ciphertext size – Encryption can entail a substantial increase in data size, which can cause a significant bandwidth problem (McCarthy and Fourniol, 2020).
- Noise accumulation – In several Somewhat Homomorphic Encryption (SHE) and FHE schemes (detailed in Annex C.8), noise accumulates in ciphertexts when computations are performed on them. As this noise grows, the ciphertexts eventually can't be decrypted (Yamada et al., 2019). So, this noise growth typically limits the size of computations that could be performed with HE. However, there is a special method used by some FHE schemes called bootstrapping, which reduces the noise embedded in a ciphertext, with the drawback that the bootstrapping operations are extremely computationally intensive. Noise-free schemes represent a different direction in coping with the noise in FHE ciphertexts. While common noisy SHE/FHE schemes are based on well-known and scrutinized mathematical problems, such as the LWE problem, noise-free schemes usually rely on less common algebraic trapdoors, which do not have widely investigated reductions to hard problems.
- Performance – Compared with computing on unencrypted data, HE is extremely computationally expensive and has lower throughput. In FHE, the running time increases dramatically with the number of operations (additions or multiplications) due to the bootstrapping step. These concerns are the subject of ongoing research (Chakarov and Papazov, 2019). The previously referred packing techniques, for example, intend to address this issue.

- Expert knowledge required – HE takes extensive expert knowledge to design meaningful and useful programs constructed from atomic HE operations. There are already some proposals for simplifying this process, making HE more widely available and usable by the programmers' population (Archer et al., 2019).
- Multiple data providers – Traditional HE schemes only allow computation on ciphertexts decryptable under the same secret key. Therefore, HE does not naturally support secure computation applications involving multiple data providers, each using its own secret key. A cryptographic primitive called Multi-Key Homomorphic Encryption (MKHE) was introduced and supports arithmetic operations on ciphertexts that are not necessarily decryptable with the same secret key. Most studies about MKHE schemes are purely abstract, with no given implementation. There is one proof-of-concept implementation of an MKHE scheme that supports packed ciphertexts (Chen et al., 2019).
- Trust managing – Considering current developments in HE, it may be hard for the client to verify that the server performed the function it said it would (The Royal Society, 2019), which is another subject of ongoing research.

**MATURITY LEVEL** Different implementations and architectures were developed to facilitate practical HE schemes: hardware architectures, such as cryptoprocessors and GPU accelerators (e.g., cuHe), as well as software libraries, such as HElib, PALISADE, SEAL, TFHE, and NFLlib (Mert et al., 2020). There is also the iDash competition, which occurs each year, challenging participants to produce a privacy-preserving solution using homomorphic encryption (Curtis and Player, 2019). Furthermore, the HomomorphicEncryption.org consortium has begun an effort to standardize both an API and advice on secure parameter selection. An important output of the consortium is the Homomorphic Encryption Security Standard, which recommends parameter sets achieving certain target security levels. These parameter sets do not necessarily reflect implementation choices in the most commonly used HE libraries.

However, Homomorphic Encryption is not currently appropriate when analysts wish to carry out arbitrary computations (Nguyen-Van et al., 2019). The ones relying primarily on adding encrypted data items, and performing only a limited depth of multiplications on secrets, can be executed in practice. So, Partially Homomorphic Encryption (PHE) schemes, described in Annex C.8, are widely used (product maturity level), while SHE and FHE are the subject of current ongoing research (some solutions already evaluate relatively simple arithmetic circuits efficiently).

**APPLICATION PHASE** HE can be applied to data storage and computation.

## 2.2. Privacy Enhancing Technologies

**APPLICATION CONTEXT** HE can be used to analyze data in circumstances where all or part of the computational environment is not trusted and sensitive data should not be accessible, such as the cloud and untrusted computers or service providers. This technology provides confidentiality and is applicable where the computation required is known and relatively simple.

Most applications, such as industrial and machine monitoring, personal healthcare, electronic voting systems, and privacy-preserving machine learning, can take advantage of HE to process their sensitive data, which cannot be exposed to any third party (Alabdulatif et al., 2020).

**ADDITIONAL INFORMATION** Annex C.8.

### 2.2.5 Privacy-Preserving Computation

#### **Secure Multi-Party Computation**

**CONCEPT** Secure Multi-Party Computation (MPC) is a class of cryptographic techniques that allow for secure computation over sensitive datasets. In MPC, a set of distrustful participants jointly compute over the data they hold individually (Volgushev et al., 2019). As long as some parties, typically a majority, honestly follow the protocol, no party learns anything beyond the computation's final output.

**GOALS** Allow some participants to securely evaluate a function on their private inputs so that no information other than an agreed-upon output is available to the participants (Bayatbabolghani and Blanton, 2018).

**ADVANTAGES**

- Multiple data providers – Some advanced cryptographic algorithms provide data privacy using encryption and support computation on such encrypted data, such as Homomorphic Encryption algorithms. MPC allows collaboratively executing data mining on encrypted data from different parties (Jiang et al., 2020).
- Distributed computation – Each participant performs the agreed-upon computation over its data (Kaiser et al., 2019).
- No trusted central authority – MPC removes the necessity of unconditionally trusting one single party, which would perform the computation or analysis by pooling together all participants' data and decrypting it (Kaiser et al., 2019).



## DISADVANTAGES

- Protocols uniqueness – Using MPC protocols or sharing intermediate results often require changing or adapting the data mining algorithms (ENISA – European Union Agency for Cybersecurity, 2014). Hence, each cryptographic privacy-preserving data mining protocol is designed for a specific data mining computation, and, in general, it is not valid for other computations.
- Expert knowledge required – Implementing a performant and secure MPC protocol currently requires domain-specific expertise that makes it impractical for most data analysts. This and other limitations have led researchers to build oblivious query processors that generate and execute secure query plans. These query processors improve the accessibility of MPC, as they allow data analysts to write convenient relational queries (Volgushev et al., 2019).
- Performance – Currently, MPC significantly increases the time it takes to compute a given function, in part due to delays incurred in communicating encrypted data across the network (latency). The computing time has been cut down since the first implementations came out and still needs further improvement to make MPC more practical (The Royal Society, 2019).
- Scalability – Current MPC algorithms scale poorly with data size, making MPC on big data prohibitively slow and inhibiting its practical use. Many relational analytic queries can maintain MPC's end-to-end security guarantee without using cryptographic MPC techniques for all operations. When parties trust others with specific subsets of the data, new hybrid MPC-plaintext protocols have been applied to run additional steps outside of MPC and improve scalability further. The previously referred query processors also use this solution to mitigate the scalability problem, especially for operations that are notoriously slow under MPC, such as joins and grouped aggregations (Volgushev et al., 2019).

**MATURITY LEVEL** During recent years, MPC techniques have experienced dramatic advances in their performance (Bayatbabolghani and Blanton, 2018). For example, an MPC-enabled query compiler, called Conclave, makes MPC on big data accessible and efficient (Volgushev et al., 2019). Data analysts write relational queries as if they had access to all parties' data in the clear. The compiler turns the queries into a combination of efficient local processing steps and secure MPC steps. Conclave delivers results with near-interactive response times (i.e., a few minutes) for input data several orders of magnitude larger than existing systems can support. It is designed to withstand passive (or semi-honest) adversaries.

In short, while MPC has been applied in a limited number of products, research and development are ongoing, and other applications are at a proof-of-concept stage (The Royal Society, 2019).

### 2.3. Summary

APPLICATION PHASE MPC can be applied to data computation.

APPLICATION CONTEXT Many examples show the importance of secure MPC constructions in practice, such as privacy-preserving decision making on distributed medical or financial data, privacy-preserving machine learning, auctions, online poker, the private intersection of sets belonging to different organizations, among others (Bayatbabolghani and Blanton, 2018).

ADDITIONAL INFORMATION Annex C.9.

## 2.3 SUMMARY

The GDPR introduces higher and stricter privacy requirements and heavy fines for noncompliance. It is applied to all organizations processing the personal data of subjects within the EU, regardless of their location. The GDPR key points are presented in Figure 1.

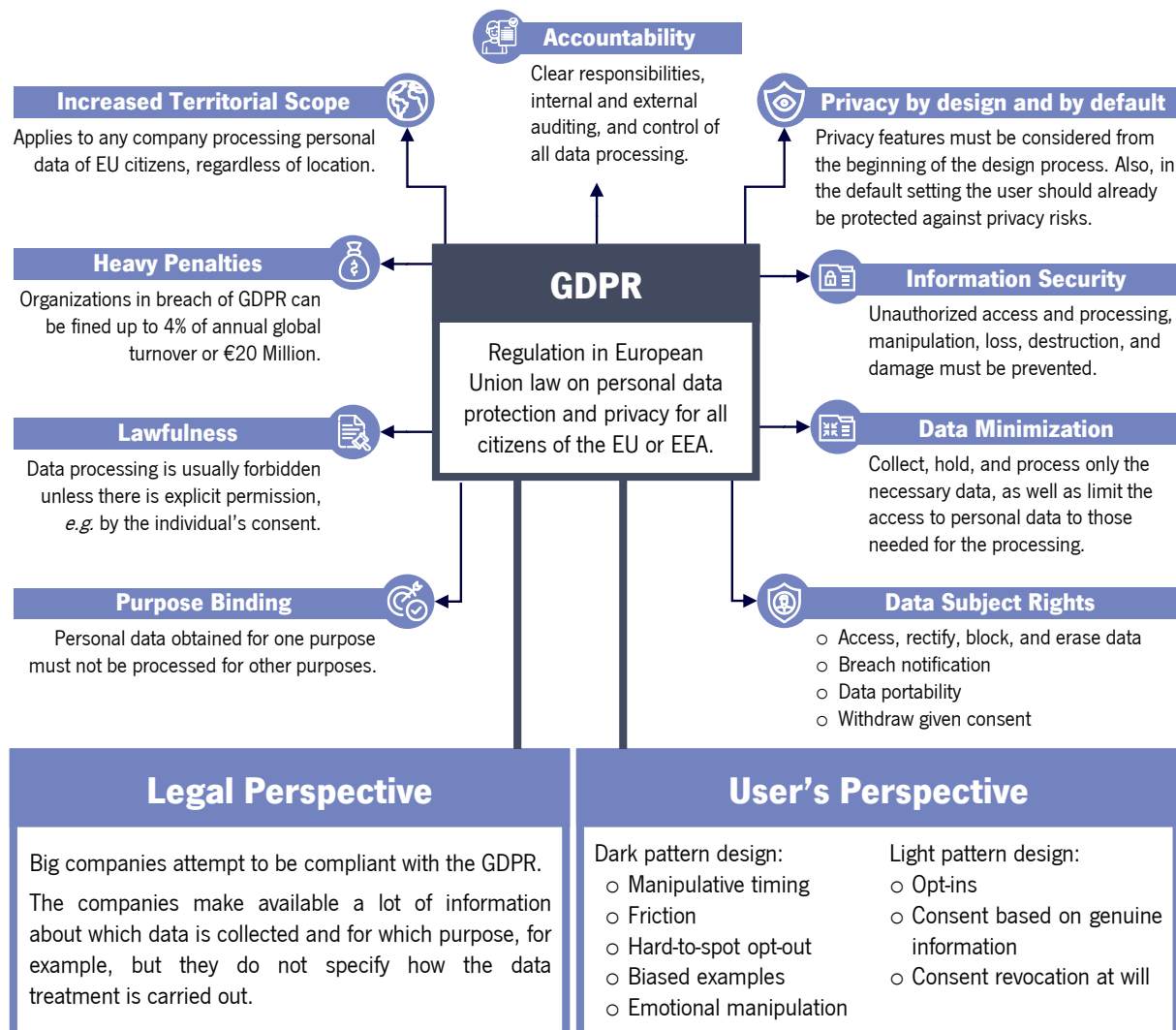


Figure 1: GDPR summary.

PETs are technologies or approaches designed to help achieve compliance with privacy policies or data protection legislation, like the GDPR. A schematic summary of the studied PETs is presented in Figure 2.

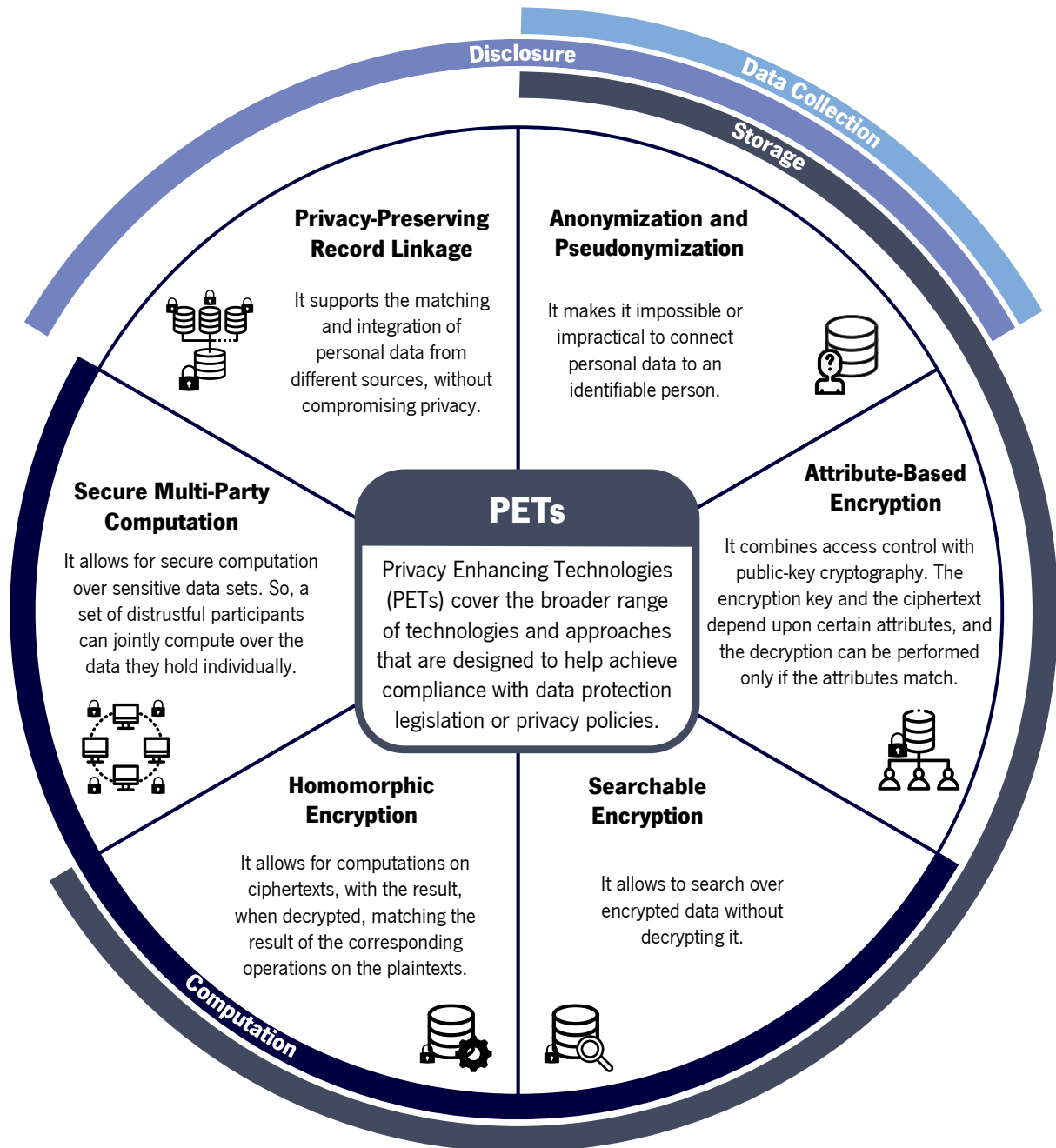


Figure 2: PETs summary.

---

## SCENARIOS

---

In this chapter, several data mining scenarios from Altice/MEO that require privacy and security guarantees are defined. Among those, the most interesting one in terms of the application of the studied technologies will be selected.

### 3.1 CROSS-SERVICE PROFILING

**CONTEXT** The main goal is to profile a user across several services, without losing the security guarantees offered by each of them. Therefore, if the user allows sharing of his information with third parties without ever revealing his identity unless he is also a customer of their service, it would be possible to offer him a more personalized experience on each service. There may not exist a global identifier across all services, so it would be necessary to use QIs to link records.

**SECURITY REQUIREMENTS** To ensure the user's privacy, some security guarantees must be verified:

- Each service cannot give information that allows identifying its clients, except for their consuming habits.
- Each service should only provide those consuming habits if the user authorizes sharing that information with third parties.
- Each service cannot receive information about a user that isn't its client.

**APPLICATION PHASE** In this scenario, the security guarantees must be ensured during data correlation.

**APPLICATION EXAMPLE** Alicia is a MEO client that uses several services of that provider:

- Location – Tracks the location of its users.
- MEO TV – Allows the user to watch television.
- NetQ – Stores information about the contracts with the clients and provides client support.

Those services allow MEO to obtain the following information:

- Alicia travels every Monday from Porto to Lisboa and every Friday from Lisboa to Porto (Location).
- Alicia likes to watch series and movies (MEO TV).
- Alicia doesn't have a suitable mobile device to watch series and movies (NetQ).

Alicia authorized each service to share her consuming habits with third parties to obtain a more personalized experience. So, MEO intends to use this information to promote its services without compromising its clients' privacy. Examples of campaigns that could be provided to Alicia are:

- Discounts on MEO Go.
- Upgrade of the 5G plan.
- Discount on the purchase of a mobile device with better features.

**SOLUTION SKETCH** The solution to this problem could be based on PPRL or secure MPC. The former technique is used in this case, as presented in Figure 3, because there are no global identifiers. An MPC-centric approach could also be used, removing the need for a trusted third party, but it was not further explored within the scope of this project.

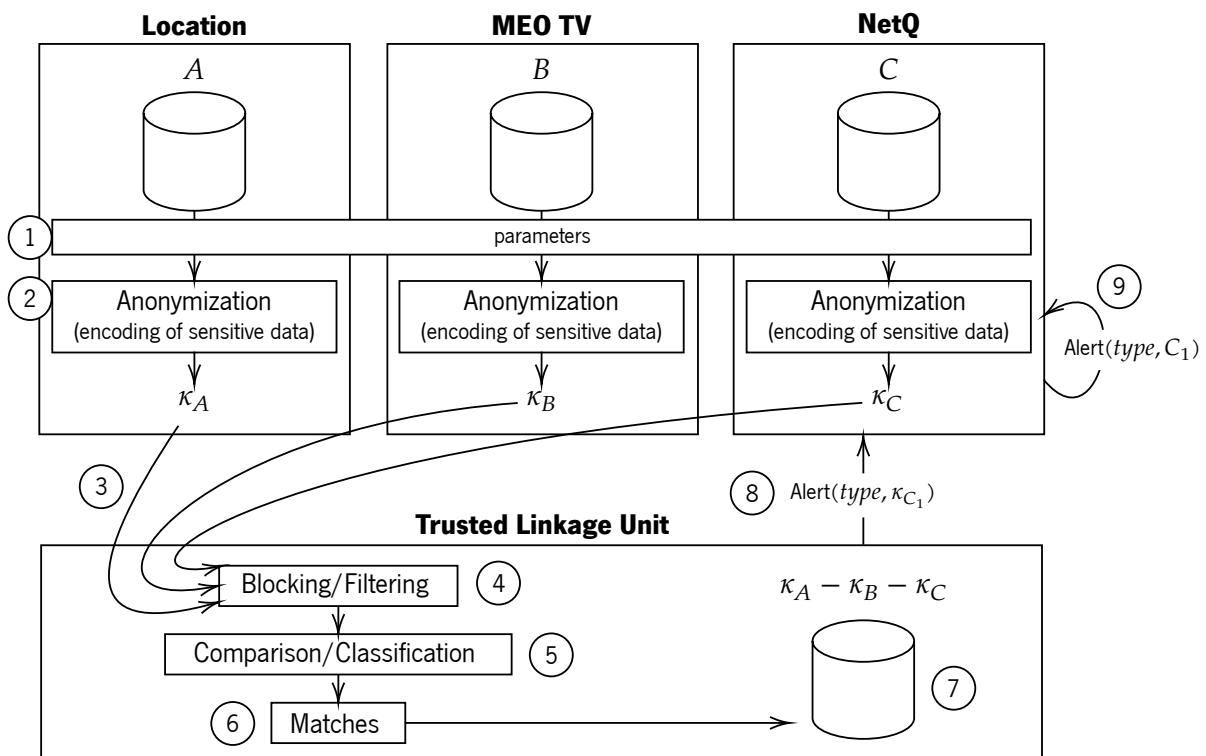


Figure 3: Diagram representing the proposed solution based on PPRL.

Each step of the proposed PPRL-based solution is specified next:

### 3.2. Health – Assisted Living Platform

1. The data holders exchange some parameters about the attributes to use in the linkage and define how to clean and standardize them. In particular, they exchange the encoding parameters.
2. Each data holder then conducts a data cleaning and standardization step, which is essential to achieve a high linkage quality. After that, the data holders anonymize their records (the ones that are authorized to be shared with third parties), each of which only includes the necessary information to link the records (encoded/masked) and the consuming habits (clean).
3. The data holders send their set of anonymized records,  $\kappa_A$ ,  $\kappa_B$ , and  $\kappa_C$ , to the linkage unit.
4. To avoid the need for every record to be compared with every record of the other sources, blocking and/or filtering techniques are used to exclude dissimilar record pairs from detailed comparisons.
5. Similarity functions are used to compare pairs of records and derive the pairs of matching records. To identify matching records, most PPRL methods follow a simple threshold-based approach such that two records are considered to represent the same real-world entity if their similarity meets or exceeds a threshold.
6. Each record can only be included in a unique match (the one with the biggest similarity value). The match may include the data of just some data holders.
7. The anonymized records are linked according to the matches and stored in a database. Those records are monitored to detect some specific patterns and generate alerts based on that pattern to the services.
8. When a certain new pattern that fits the requirements established by a service is detected, an alert is sent to the respective service with the necessary information (user's encoded data from that data holder and needed consuming habits).
9. The data holder that receives the alert decodes the data to identify its client and takes action accordingly to the alert.

## 3.2 HEALTH – ASSISTED LIVING PLATFORM

**CONCEPT** The assisted living platform, here named WebAL, is a social and health support Information and Communications Technology (ICT) solution that includes telemonitoring of vital signs and support for daily activities related to people's health, well-being, and safety.

**FEATURES** The platform is accessible via different technologies within reach of the user, flexible enough to apply the solution across multiple possible scenarios. At the center of the system is a back-office that

manages all users with different profiles, devices/sensors, activities/tasks, notifications/reminders/alerts, and related data (e.g., vital signs, video content, quizzes). The main features provided by this assisted living platform are:

- Measurement of vital signs;
- Integration of different medical devices with automatic measurement recording (with the possibility of manual insertion);
- Real-time professional distance monitoring;
- Management and scheduling of daily health and wellbeing activities and tasks;
- Automatic generation of alerts if there are indicators outside the defined thresholds;
- Configuration of custom alerts;
- Centralized management of the patient’s clinical information.

**PROFILES** There are several user profiles in the assisted living platform, the most important of which are presented in Figure 4. Each user can only perform operations or access the data that is required to fulfill his profile functions. The system administrator is responsible for managing institutions and administrators. So, this is the only entity that can manage data from all institutions. Administrator and administrative users can create other entities inside their institution. The administrative user can also create relationships between formal caretakers and patients from his institution.

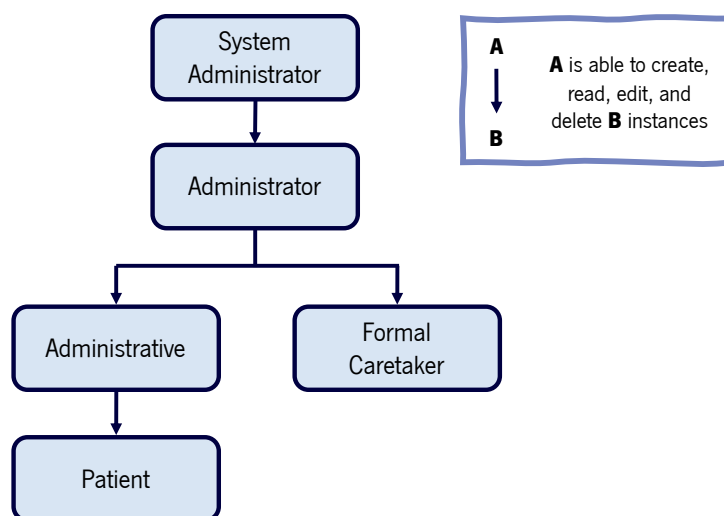


Figure 4: Assisted living platform profiles hierarchy.

**ARCHITECTURE** The diagram in Figure 5 represents the general platform logical architecture.

### 3.2. Health – Assisted Living Platform

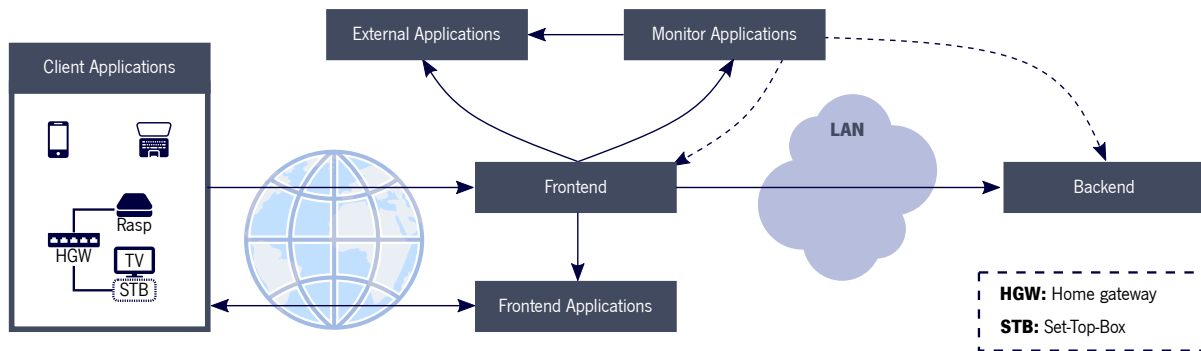


Figure 5: Assisted living platform logical architecture.

**DATA PRIVACY AND SECURITY** This platform processes personal data, which includes sensitive medical information. So, a DPIA was performed to understand all privacy risks that should be addressed. This assessment led to the emergence of new scenarios where an intervention would be useful to improve data privacy and security:

- Secure communication – Ensure that the communication between medical devices and the recipient of the transmitted data has some security guarantees, such as confidentiality, integrity, and authentication;
- Data anonymization – Ability to outsource useful medical data without compromising patients' privacy;
- Database encryption – Ensure the confidentiality of the stored sensitive data, considering performance and key management issues;
- Data Enrichment with Third Parties – Ability to compute new data about a patient using third-parties information, without data sharing between the involved entities.

Between these scenarios, the last two will be the subject of further analysis.

#### 3.2.1 Database Encryption

**CONTEXT** Encryption is mentioned by the GDPR as one way to secure stored personal data (EU GDPR Portal, 2018) and is considered the most suitable one to be implemented over the assisted living platform's database. The main goal is to protect the user's privacy while keeping an acceptable performance on data access and manipulation.

Considering the sensitivity of the information stored on this platform and the need to share it with other users, another goal is to give users more control over who can access their data, making it impossible to read unencrypted data by unauthorized entities, including the platform's server itself.



SECURITY REQUIREMENTS The following basic security requirements need to be verified in this scenario:

- The personal data stored in the database must be encrypted and the key stored in a safe location.
- Only authorized users can trigger the decryption of the stored data.

There are also some additional requirements, which would improve the platform's compliance with the GDPR, such as:

- Each user should be able to define who can access his/her personal data and change those permissions at any time (access policy update). However, this assisted living platform has a business requirement, implying that an administrator should perform this control. The users approve this on a consent given by them on the first login. Nevertheless, the proposed solution should be easily adapted so that the users themselves could define who has access to their data.
- Only currently authorized users own a key that allows them to decrypt certain personal data.
- The platform's server doesn't have access to the user's personal data unless the user has given the organization explicit consent for the access, with a specific purpose (e.g., diseases' pre-diagnosis).
- Each data access or manipulation must be securely registered (accountability).

APPLICATION PHASE Data storage and, additionally, data processing and transfer.

SOLUTION SKETCH In this scenario, there are several possible solutions or approaches, some with better performance, others with more security guarantees. Regardless of the approach being used, only the information considered personal or sensitive should be encrypted. To evaluate the best security level possible, without compromising too much the general performance of the platform, two types of solution are proposed:

- Basic encryption – There is only one key to encrypt and decrypt the whole database. The key should be securely stored on the server-side, which could be accomplished using a Hardware Security Module (HSM). It is a physical computing device that safeguards and manages digital keys for strong authentication and provides cryptoprocessing. Therefore, this approach requires the usage of specialized hardware. Another possible approach is to encrypt the key using an administrator password.

There are several levels where encryption could occur. To this platform, the most meaningful options are database encryption or application encryption.

This solution addresses only the above-described basic security requirements.

### 3.2. Health – Assisted Living Platform

- Users-based encryption – Each user can only decrypt data to which he/she was specifically authorized to access. This extends to the assisted living platform server, which cannot decipher users' medical data unless it was explicitly authorized by a manager, for some purpose.

This approach includes the entities represented in Figure 6. The patients are the data subjects of health information. The data producers and data consumers respectively write and read data about the patients. The Key Generation Center (KGC) issues the users' ABE keys. Regarding personal information, while the Data Storage saves users' encrypted data, the Security Storage saves encrypted keys and all key access requests.

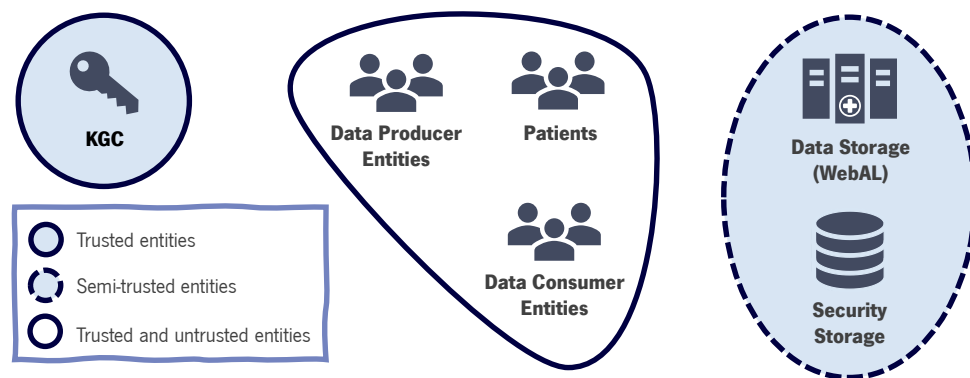


Figure 6: Entities involved in the Database Encryption scenario solution.

The proposed solution, based on the scheme presented by Pournaghi et al. (2020), consists of the following 3 phases:

- Setup – This is an initial phase in which some keys and administrative entities are created. First, a default user with the profile System Administrator is created in all servers, WebAL, KGC, and Security Storage. Then, this user creates other ones, which in turn perform more user registration operations, all according to each user permissions shown in Figure 4.
- Write Data – In this phase, data is inserted, encrypted, and stored. This process is shown in Figure 7 (encryption). Before anyone entering data about a particular patient, an administrative user should insert in the KGC a relationship between the patient and the user entering the data. Only then the data producer will be authorized to enter the patient's medical data. That data is encrypted with a private symmetric key and stored on the WebAL database. Finally, the symmetric private key is encrypted using ABE (so that only authorized users can decrypt it) and stored in the Security Storage.
- Read Data – In this phase, data is read, decrypted, and presented. This process is shown in Figure 7 (decryption). First, the encrypted symmetric private key is obtained from Security Storage and decrypted using the data consumer ABE private key. This private key is generated

and provided by the KGC upon authentication. Then, the encrypted data is obtained from the WebAL database and decrypted using the symmetric private key.

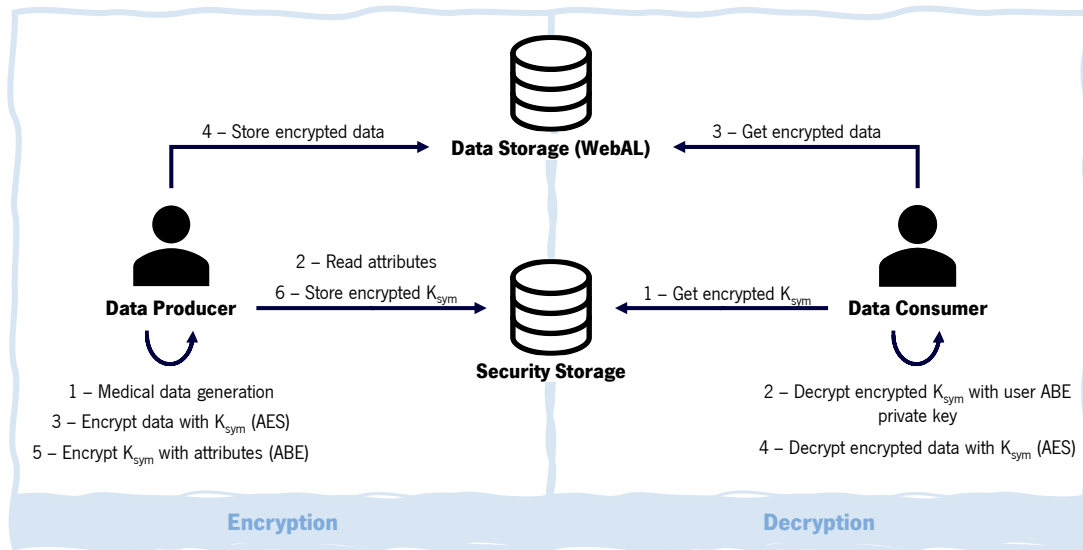


Figure 7: Data writing and reading in the presented solution.

The KGC entity needs to be fully trusted because the user must trust it with his plain data (e.g., credentials) and data processing (e.g., the user ABE private key issuance). If this entity is somehow compromised, the whole system is endangered. One of this scheme's main goals is to minimize the data saved and operations performed by this trusted entity. Also, KGC should be generic enough so it could be used for more platforms alongside this one. An example of this kind of entity is the Certification Authorities (CAs).

The users' universe is composed of both trusted (authorized) and untrusted (unauthorized) entities. So, authentication and permissions validation is mandatory for each access to the servers that may reveal private data. This scheme includes an intrinsic access control mechanism (ABE) for reading and writing data. That is, only authorized users, according to the attributes used in the symmetric key encryption, can decipher the encrypted information. However, it is still needed to perform access control at the application level to prevent data from being improperly created, removed, or replaced.

All sensitive data sent to the Data Storage and the Security Storage should be encrypted, so these entities must be semi-trusted: the user doesn't trust these entities with his clean data but trusts them to keep their encrypted data secure and to perform the expected operations or data processing. If one or both entities are compromised, no clean data can be accessed by the attacker. However, he can create, delete, or corrupt data. So, the security of these entities is still crucial.

### 3.2. Health – Assisted Living Platform

Regarding the ABE system, both its variants Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and Key-Policy Attribute-Based Encryption (KP-ABE), described in Annex C.6, could be used. However, the KGC trusted entity should be the one to define which messages can be accessed by the users, instead of the users being the ones that determine who can access the messages encrypted by them. As a result, the KP-ABE variant was selected to assure that the KGC entity would provide more flexibility to establish access control.

The method of encrypting keys with ABE and data with a symmetric key is used due to the greater efficiency of symmetric encryption and decryption operations. So, this method may have its advantages if there is a large amount of data to encrypt and decrypt.

In addition, this approach has the great advantage of allowing the usage of other types of ciphers as an alternative to the symmetric cipher. Thus, it will be possible for the server to carry out searches or other operations on the encrypted data using, for example, SE and HE. Examples of this kind of operation are the computation of the encrypted Body Mass Index (BMI) value from the user's encrypted weight and height, or the comparison of encrypted measurements with threshold values to trigger alerts.

This solution addresses all basic and additional security requirements. Regarding the accountability requirement, it can be ensured through signatures performed by the users over their requests.

#### **3.2.2 Data Enrichment with Third Parties**

**CONTEXT** The idea is to use third-party data together with the assisted living platform data to draw conclusions about the patients' health in a way that each entity's data privacy is not compromised. A concrete example would be to use the measurements from the assisted living platform and analysis from a hospital to warn doctors about the risk of a specific patient having some disease, taking into account the available data.

Each patient would have to give his/her consent for this data processing to occur and for the assisted living platform to receive the result data, but each entity's data should remain private.

**SECURITY REQUIREMENTS** To ensure the user's privacy, some security guarantees must be verified:

- The user's data from each entity must remain private.
- Only the assisted living platform receives the result from the data processing.
- User identifying data should only be used on entities to which the user has given consent.

APPLICATION PHASE Data processing.

SOLUTION SKETCH This solution proposes to diagnose the patient's disease using Euclidean distance, to get the similarity values between the patient medical data and certain known diseases. To keep the patient's data from each entity private, Secure Multi-Party Computation is applied to data processing. The presented solution is based on an article written by Li et al. (2019a).

In this scheme, the patient is assumed registered in all third-party entities he authorized to perform the collaborative data processing. When the patient asks about his/her illness, the assisted living platform first authenticates the patient in those third-party entities, using the OAuth2<sup>1</sup> protocol, for example.

Then, the computation stage occurs, aiming to obtain the most probable disease(s). For the sake of simplification, this stage explanation will be about the calculation of the similarity between a disease trait vector and the medical data from a hospital and the assisted living platform. The shift to more third-party entities is trivial, as well as the calculation of the similarity for more diseases. The following steps could be improved to simultaneously calculate the similarity of various diseases, thus minimizing the communication between entities. Also, it should be noted that different entities must not provide information about the same health parameter.

So, in the computation stage, an Asymmetric Additively Homomorphic Encryption scheme will be used. A public-key encryption scheme  $Enc$  is additively homomorphic if it has a valid operation  $\otimes$  that is not dominated by any secret keys. For any pair of plaintexts  $x_1, x_2$ :

$$Enc(x_1) \otimes Enc(x_2) \equiv Enc(x_1 + x_2) \quad (1)$$

It should be noted that ciphertext of  $nx$ , which adds  $x$  with  $n$  times, can be expressed as:

$$Enc(x)^n \equiv Enc(nx) \quad (2)$$

For this explanation, the following variables and equations will be considered:

- $n$  – Number of medical data values used from WebAL (W) to diagnose the disease ( $n \geq 0$ ).
- $m$  – Number of medical data values used from the Hospital (H) to diagnose the disease ( $m > 1$ ).
- $Q$  – Vector with the useful patient's medical data for the disease diagnosis.

$$Q = (\underbrace{q_1, q_2, \dots, q_n}_{Q_W}, \underbrace{q_{n+1}, \dots, q_{n+m}}_{Q_H}) \quad (3)$$

<sup>1</sup> OAuth 2.0: <https://oauth.net/2/>

$\mathbf{Q}_W$  is the patient's medical data in the WebAL platform, and  $\mathbf{Q}_H$  is the patient's medical data in the Hospital, both used for the disease diagnosis.

- $\mathbf{T}$  – Disease trait vector.

$$\mathbf{T} = \underbrace{(t_1, t_2, \dots, t_n)}_{\mathbf{T}_W}, \underbrace{(t_{n+1}, \dots, t_{n+m})}_{\mathbf{T}_H} \quad (4)$$

$\mathbf{T}_W$  and  $\mathbf{T}_H$  are the disease trait vectors with the health data values related to WebAL and the Hospital, respectively, both stored in WebAL.

- $D$  – The square of the Euclidean distance between the patient's medical data vector  $\mathbf{Q}$  and the disease trait vector  $\mathbf{T}$ .

$$\begin{aligned} D = D(\mathbf{Q}, \mathbf{T}) &= \|\mathbf{Q} - \mathbf{T}\|^2 \\ &= \sum_{j=1}^{n+m} (q_j - t_j)^2 \\ &= \sum_{j=1}^{n+m} (q_j)^2 + \sum_{j=1}^{n+m} (t_j)^2 + \sum_{j=1}^{n+m} (-2q_j t_j) \end{aligned} \quad (5)$$

The  $D$  value can be computed using the distance between the patient's medical data vector and the disease trait vector for the WebAL ( $D_W = D(\mathbf{Q}_W, \mathbf{T}_W)$ ) and the Hospital ( $D_H = D(\mathbf{Q}_H, \mathbf{T}_H)$ ):

$$\begin{aligned} D = D(\mathbf{Q}, \mathbf{T}) &= \|\mathbf{Q} - \mathbf{T}\|^2 \\ &= \sum_{j=1}^{n+m} (q_j - t_j)^2 \\ &= \sum_{j=1}^n (q_j - t_j)^2 + \sum_{j=n+1}^{n+m} (q_j - t_j)^2 \\ &= \|\mathbf{Q}_W - \mathbf{T}_W\|^2 + \|\mathbf{Q}_H - \mathbf{T}_H\|^2 \\ &= D_W + D_H \end{aligned} \quad (6)$$

- $sim(\mathbf{Q}, \mathbf{T})$  – Similarity between the patient's medical data vector  $\mathbf{Q}$  and the disease trait vector  $\mathbf{T}$ .

$$sim(\mathbf{Q}, \mathbf{T}) = \frac{1}{1 + D(\mathbf{Q}, \mathbf{T})} \quad (7)$$

- $(pk_H, sk_H)$  – Key pair generated by each third party (the Hospital, in this case) to perform HE.  $pk_H$  is the public key used for encryption, and  $sk_H$  is the private key used for decryption.

- $M_1$ ,  $M_2$ , and  $M_3$  – Parts computed by WebAL to calculate the encrypted  $D_H$  value, using the Hospital encrypted information.

$$\begin{aligned}
 Enc(D_H) &= Enc\left(\sum_{j=1}^m (q_{Hj})^2 + \sum_{j=1}^m (t_{Hj})^2 + \sum_{j=1}^m (-2q_{Hj}t_{Hj})\right) && \text{by (1)} \\
 &= Enc\left(\sum_{j=1}^m (q_{Hj})^2\right) \cdot Enc\left(\sum_{j=1}^m (t_{Hj})^2\right) \cdot Enc\left(\sum_{j=1}^m (-2q_{Hj}t_{Hj})\right) && \text{by (1)} \\
 &= Enc\left(\sum_{j=1}^m (q_{Hj})^2\right) \cdot Enc\left(\sum_{j=1}^m (t_{Hj})^2\right) \cdot \prod_{j=1}^m Enc((-2t_{Hj})q_{Hj}) && \text{by (2)} \\
 &= \underbrace{Enc\left(\sum_{j=1}^m (q_{Hj})^2\right)}_{M_1} \cdot \underbrace{\prod_{j=1}^m Enc(q_{Hj})^{-2t_{Hj}}}_{M_2} \cdot \underbrace{Enc\left(\sum_{j=1}^m (t_{Hj})^2\right)}_{M_3} && (8)
 \end{aligned}$$

Hence, after authentication, the steps presented in Figure 8 are executed to compute the similarity value between the patient medical data and some disease trait vector.

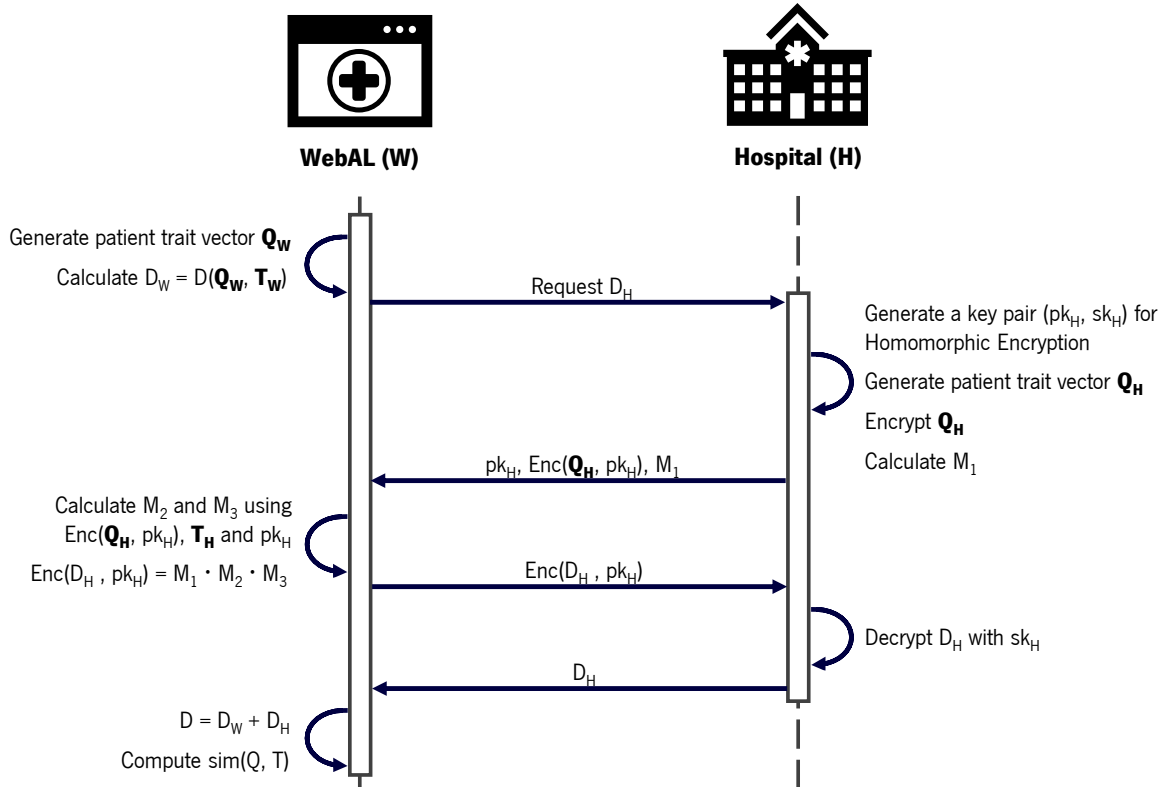


Figure 8: Computation stage sequence diagram for data enrichment with the third parties scenario.

### 3.3 FINAL CONSIDERATIONS

All the proposed scenarios have very interesting challenges, both in terms of utility in the big data context, such as the Cross-service Profiling scenario, and in terms of mathematical and cryptographic challenges, such as the Data Enrichment with Third Parties scenario.

However, the users-based encryption solution from the Database Encryption scenario was selected because of the related technical challenges and its relevance at the business level. Among the technical challenges, there are the management of encryption keys and the possibility of decryption only by authorized users, which requires the server to perform some operations on the encrypted data if it is not authorized to decrypt that data. All of this should be possible without compromising the practical use of the assisted living platform, i.e., the increase in response time and processing power requirements should be acceptable. The importance of this scenario at the business level is evident in the context of user privacy, which is reinforced by the GDPR.

### 3.4 SUMMARY

In this chapter, several PETs application scenarios considered relevant for the Altice Labs company were presented, one to be applied to several services, and two applied to an assisted living platform, which will be referred to as WebAL in this document.

The first scenario is named Cross-service Profiling and some basic information about it is presented in Figure 9. In this scenario, if the user consents to the linkage of some of his information between different services, it will allow to offer him a more customized experience on each one of them. However, for privacy and security purposes, the entities involved shouldn't be able to access the user's consumption habits of the other services.

To solve this issue, two approaches are proposed using the studied PETs. The first approach is based on Privacy-Preserving Record Linkage and requires a

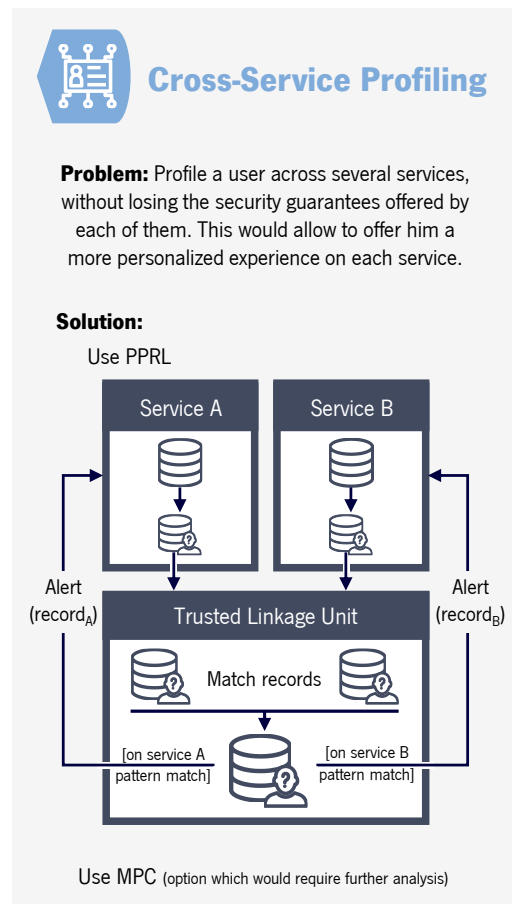


Figure 9: Cross-service Profiling scenario summary.



trusted entity to perform the records linkage. This entity will have access to the encrypted identifiable information used in the linkage process and each user's consumption habits. The other possible approach is the usage of Secure Multi-Party Computation, which would require further analysis.

The other scenarios were created in the context of an assisted living platform, which stores the users' personal information, including the patients' medical data. This medical data can be created or accessed by other users, as long as they are related to the patient.

A DPIA was developed to understand all data privacy risks which should be addressed in this platform. In this context, the scenarios Database Encryption and Data Enrichment with Third Parties emerged. A summary of each one of these scenarios is presented in Figure 10.

The Database Encryption scenario has a solution based on ABE (possibly also SE and HE), while the Data Enrichment with Third Parties scenario has a solution based on MPC with HE.

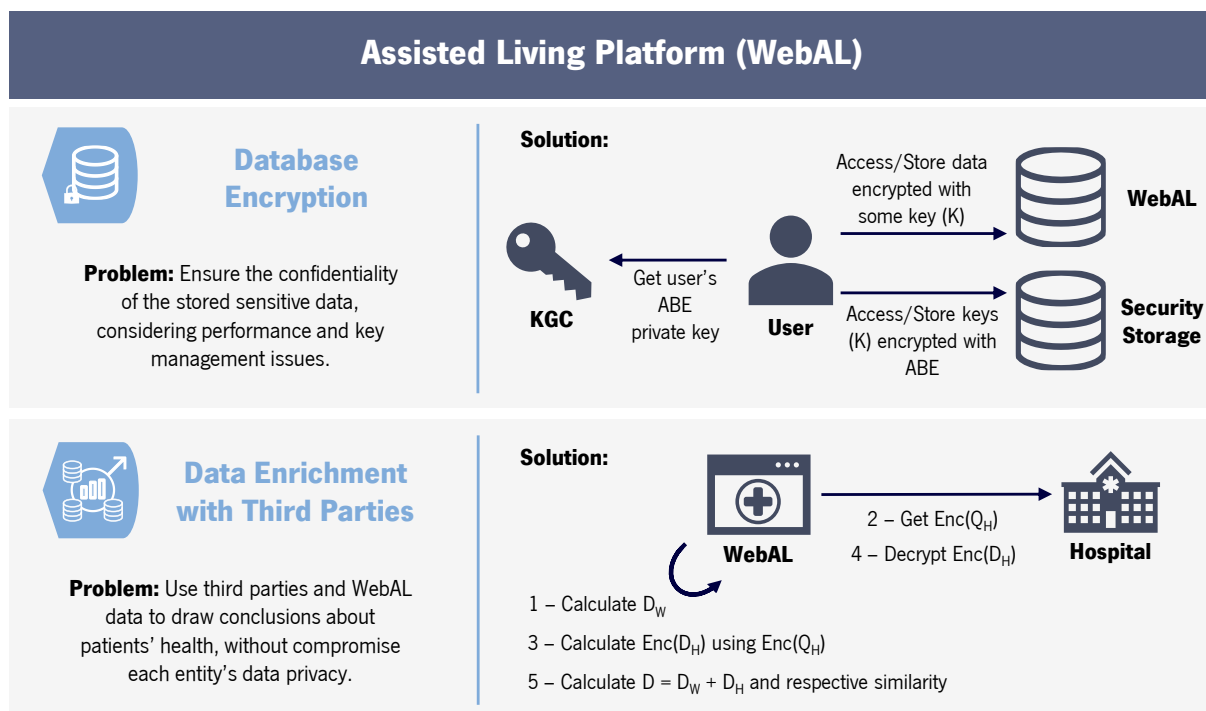


Figure 10: WebAL related scenarios summary.

For the development phase, the scenario Database Encryption was selected because of the related technical challenges and relevance at the business level.

---

## CHALLENGES

---

The previous chapter covers some useful scenarios for the company Altice Labs, which could use PETs based solutions to address data privacy and security use cases. This chapter will present some challenges related to the Database Encryption scenario, the chosen one for the implementation phase.

This scenario aims to improve an assisted living platform's privacy and security features, allowing only authorized entities to decipher the users' medical data. This extends to the platform server, which shouldn't be able to decrypt users' medical data. Suppose some operation needs to be performed by the server over the encrypted data, and the usage of PETs is not enough to accomplish that or its cost is too high. In that case, the manager can explicitly authorize the platform to decipher the data for some specific purpose.

In this chapter, the main challenges regarding that scenario and the proposed solution will be highlighted.

### 4.1 CLIENT DEVICES WITH LIMITED RESOURCES

Users can interact with the assisted living platform using the Android application, browser, or TV application. Because data must be encrypted on the client-side, the encryption algorithms will have to be used in all these environments. Among them, the browser is the most controversial one to implement security features, and the TV is the most difficult one to apply these features due to resource limitations.

In the browser, the only way to manipulate the data before sending it to the server is by using JavaScript. JavaScript cryptography is considered harmful by several sources, such as Ptacek (2011) and Arcieri (2013). According to the last author, there are ways to develop cryptographic applications to enforce the interests of the web application creator, but not the user:

*“If ample precautions are taken (which includes a large laundry list of things like TLS, CSP, CORS, proper HTTP headers, JS strict mode, and more), this can allow for the suc-*

*successful development of cryptographic applications that attempt to enforce the interests of the web application creator. But what about the user?*

*Do programs in the browser represent the interests of the user? According to Commander Douglas Crockford [...] the answer is a resounding NO. This is not the traditional threat model of the browser.*

*[...]*

*Unfortunately, it [the web] does not rely on a comprehensive cryptographically secure signature system to determine content is authentic, but instead just trusts whatever is sitting around on the server at the time you access it. This is worsened by the fact that web browsers give remote servers access to wide-ranging local capabilities exposed via HTML and JavaScript. This creates an environment that is not particularly safe or stable for use in creating, storing, or sharing encryption keys or encrypted messages.”*

In addition to the requirements mentioned in the previous excerpt to successfully develop cryptographic applications in the browser, there is also the need to build a secure User Interface (UI) avoiding script interference and Cross-Site Scripting (XSS) attacks.

However, even if all these requirements are satisfied, the JavaScript files are loaded in runtime. This means that although the user does not trust his data to the server, he will have to rely on it to guarantee that data's security. Anyone who has sufficient access can inject a malicious payload into the code at any point in time. They could even selectively target users, so the rest of the world would think that the code is fine, but a particular victim would receive the malicious payload.

Browser extensions could be used to mitigate this problem, but they also present their own security issues. Nevertheless, it would be an improvement to study and possibly apply in the future.

Regardless of this improvement, it is worth noting that this assisted living platform will continue to require users to trust that it will carry out the operations it claims to perform. However, the fact that the code is on the client-side makes data processing more transparent, allowing users with an understanding of the subject to check whether the platform actually does what it advertises. This solution also helps to achieve GDPR compliance, as the server does not have access to data that is not needed since it is encrypted. That data is available and decipherable only by the authorized users.

Nevertheless, users should be advised to use the native application instead of the web application for security reasons.

Regarding the TV application, most set-top boxes do not have the resources to perform custom encryption and decryption operations. So, an additional client-side or server-side component would be needed to

execute these operations for the user. Most recently, the MEO Android TV boxes were introduced, allowing to perform the operations on the box without additional components.

## 4.2 CROSS-DEVICES USAGE

The same user must be able to access the assisted living platform from different devices without a previous configuration. However, the presented solution requires the user to have access to an ABE private key. Also, to guarantee the non-repudiation characteristic for some operations, each user must have exclusive access to his signature private key, and all entities must have access to the users' public keys. So, there is the need for a centralized trusted entity to make these keys available to each user or entity. The KGC is a trusted entity and already issues ABE private keys for each user. Therefore, this was the chosen entity for managing users' keys.

A new ABE private key can be issued for a specific user every time he/she needs one. So, there is no need for the KGC to store ABE private keys. When the user logs in, it can issue a new appropriate ABE key to be used in that session.

Regarding the signature key pair, its generation and storage process is represented in Figure 11.

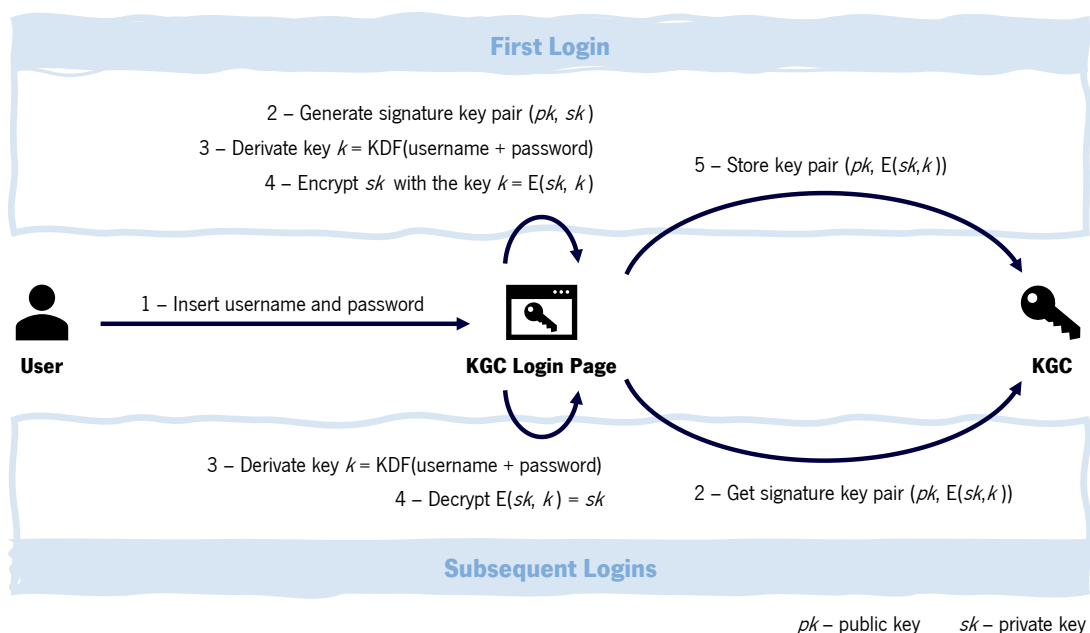


Figure 11: User's signature key pair generation, storage, and retrieval.

The signature key pair is generated on the client-side, on the first login. The user's username and password are then used to encrypt the user's private key so that the KGC won't have access to it. First, those credentials are used as input for a Key Derivation Function (KDF) to generate a symmetric key. Subsequently, that key is used to encrypt the signature private key. The pair public key and encrypted

signature private key is then sent to the KGC, which stores it. On the next login attempts, the encrypted private key is sent to the authenticated user, which can decipher it using his/her username and password.

For this key encryption to fulfill its purpose, the KGC can never access the original user's password. So, the password sent to the KGC for authentication will result from a KDF over the user's original password. This process still has the advantage that equal passwords in different applications, with varying parameters for the KDF, end up being different for the servers of those applications. Therefore, if a user adopts the same username and password for several applications and those credentials are leaked from one that uses this method, the leaked username and password will not allow for authentication on the other applications.

### **4.3 ACCESS POLICY UPDATE**

**ADD RELATIONSHIP BETWEEN USERS** The insertion of a new users' relationship, such as between a patient and a doctor, implies that each involved user should begin to have access to some past and future data about the other user. To allow this operation, the respective keys must be re-ciphered with the attributes that will authorize the related user. This re-encryption is performed by the trusted entity that issues users' ABE keys, the KGC. This entity can generate a new ABE key to decipher the encrypted symmetric key and cipher the latter with the new attributes.

**REMOVE RELATIONSHIP BETWEEN USERS** The removal of an existing relationship implies that each involved user should stop being able to access any past or future data about the other user. To allow this, all symmetric keys regarding the data of the related users would have to be re-ciphered with new attributes. Nevertheless, suppose one of the users stored the other user's symmetric key at some point, which will not be the assisted living platform's usual behavior. In that case, the first user will still be able to decipher the encrypted data related to the second one. To avoid this, it would also be needed to re-cipher all data about both users.

However, if the user stored the other user's symmetric key, he could also have stored that user's data to which he had access at that time. Thus, it is considered enough to generate a new symmetric key to encrypt future data and leave past data encrypted with the previous keys. For this reason, each symmetric key and encrypted data will have an associated version, which will allow matching the key used to encrypt certain data with that data. Past keys should still be re-encrypted by KGC with the new attributes. The new key will be generated by the next user trying to access it and with permission to do so.

It should also be noted that there is still the application permissions validation, which would not allow users to access unauthorized data. Besides, if there is a data breach in the assisted living platform or the

Security Storage, only users who currently have access to data or had access to certain data at some point will be able to decipher that data.

### 4.4 SERVER OPERATIONS

There are several basic operations currently performed by the assisted living platform server that, with the proposed scheme, will have to be executed by the client over the plaintext data or by the server over the encrypted data. If some operation has to be performed by the server and currently cannot be performed over encrypted data, the data owner or the administrative entity would have to authorize the platform server to decrypt the needed data for that specific purpose. The platform server would then have an ABE key that would grant it access to the data key and, hence, the data itself.

The operations most commonly performed by the assisted living platform server were identified and can be summarized and solved as follows:

- Measurement comparisons – For each inserted measurement, the respective thresholds for that patient are obtained from the database and compared ( $>$ ,  $<$ ,  $\geq$ , and  $\leq$  operations) with the measurement value. This will allow assigning a color (green, orange, or red) to the inserted measurement, representing a warning level. If the inserted value is not between the expected thresholds (green), an alert is sent to the patient's formal caretaker.

The comparison operation can be performed over encrypted data if ORE is the encryption scheme used.

- BMI computation – When the patient's weight is inserted, his BMI is also computed and stored.

$$BMI = \frac{weight}{height \times height}$$

To perform this operation, multiplicative HE could be used. For that, the value  $\frac{1}{height \times height}$  would have to be computed and encrypted by the client-side when setting the patient's height.

However, similarly to the previous item, this measurement value must be compared with certain thresholds after computation. For that, it must be encrypted with an ORE scheme. So, the BMI value must be precomputed on the client-side in plaintext, encrypted with ORE, and then sent to the server to perform the needed comparisons.

- Calories computation – When a quantity of steps is inserted for a patient, the calories burned are also computed and stored.

$$distance = 0.414 \times \frac{height}{100} \times steps$$

$$calories = distance \times weight \times \frac{1.036}{1000}$$

To perform these operations, multiplicative HE could be used. However, as explained on the previous topic, it must be precomputed on the client-side in plaintext, encrypted with ORE, and then sent to the server to perform the needed comparisons.

- Dates operations – Several operations with dates and times are performed by the platform server, such as sum, subtraction, comparison, conversion between formats, and sorting. These operations involve dates and times regarding the same patient, thus being encrypted with the same key.

Dates' sum or subtraction can be performed using additive HE. Comparison between dates or sorting them can be achieved by resorting to ORE. If there is the need to execute both kinds of operations over the same date, two versions can be kept in the database, one with its components encrypted with HE and another with ORE.

The conversion of dates between formats is used to present them to the user in a more appealing form. So, this operation can be performed directly on the client-side, using the decrypted date.

- Database queries – Several queries are performed over the database, such as simple WHERE queries. More complex queries are also performed, using LIKE, or ORDER BY and LIMIT clauses. Generally, these kinds of queries can be performed over encrypted data using different techniques of SE.

However, the overhead of performing such searches over encrypted data may be too big. It would have to be compared in terms of performance and security with other options, such as sending all unfiltered/unordered data to the client-side (which would perform the needed operations), allowing the platform server to decrypt that data, or allowing it to have only the required data unencrypted.

- Alarms – Some alerts are fired at a certain time and use the patient's personal information. These alerts are used as a reminder for taking medication, for example. The notification service must know the needed data, such as username, contact information, alert message, and alert times. So, this service must have access to a key that decrypts the mentioned data.

## 4.5 SUMMARY

In this chapter, the main challenges regarding the scenario and solution chosen for implementation were described, as well as how they will be addressed in the development phase. A summary of those challenges and their solution is presented in Figure 12.

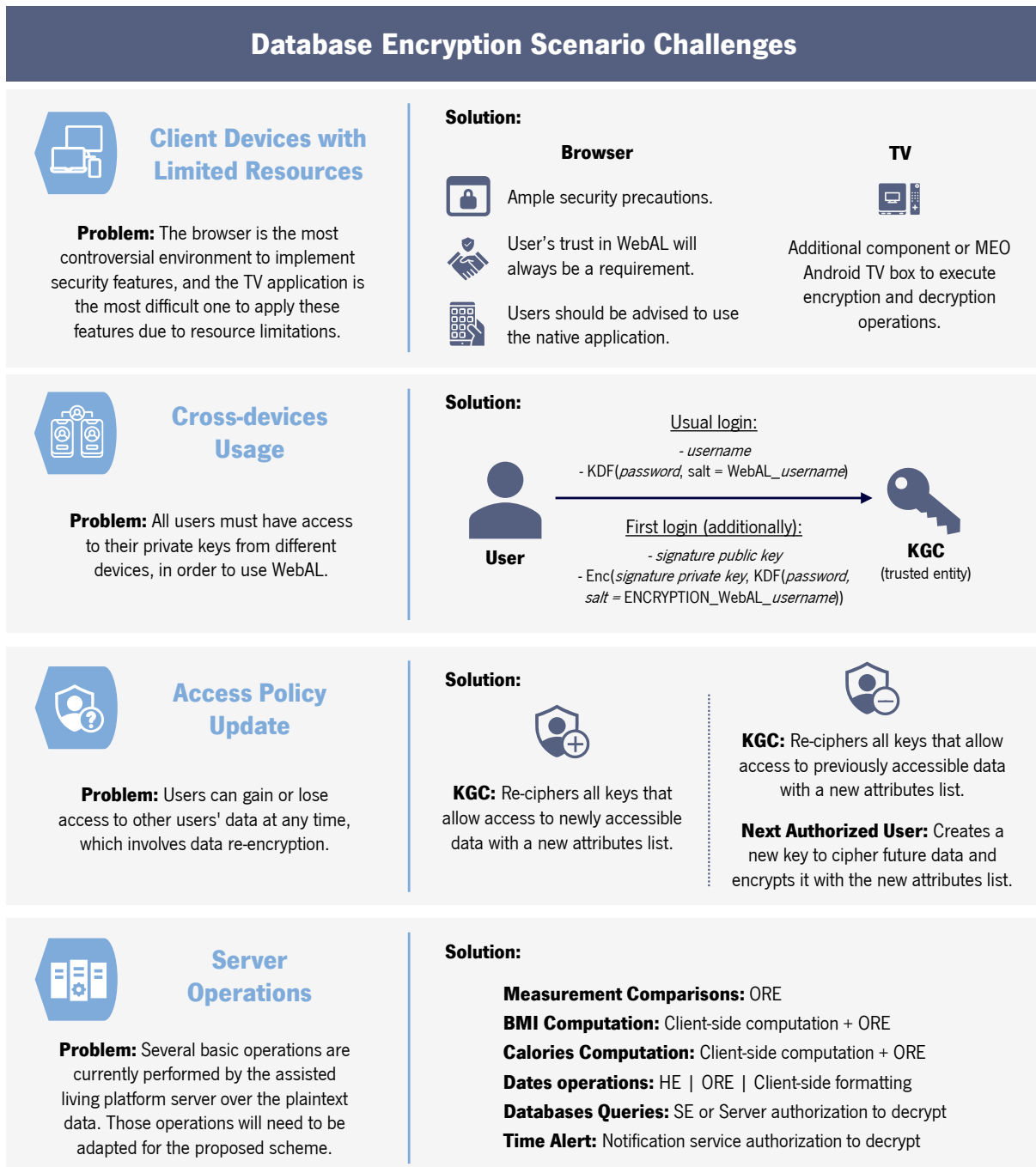


Figure 12: Database Encryption scenario challenges summary.



---

## DEVELOPMENT

---

This chapter will address the project development phase, which consists of implementing the proposed solution for the scenario Database Encryption. The major implementation decisions will be presented first. The implemented scheme will then be described, starting with the system architecture, going through the stored data, and ending with the main operations communication flow.

### 5.1 DECISIONS

The current section will address the major implementation decisions regarding technologies used for each service and authentication, as well as cryptographic algorithms and approaches.

For the proposed scheme to be applied in practice, the client-side operations must be implemented on the browser, as well as the Android and TV applications. The applications do not entail as many challenges as the browser, both regarding private keys' storage and integration of cryptographic algorithms. Indeed, the browser is considered a less secure environment to perform cryptographic operations (or store private keys). Also, the existing implementations of cryptographic algorithms are more challenging to integrate with JavaScript than with Android or a TV additional component using, for example, Java.

Aiming to solve these additional challenges related to applying the proposed scheme on the browser, this was the chosen client-side environment for the prototype implementation.

#### 5.1.1 REST Services

For the sake of simplicity, each entity (WebAL, KGC, and Security Storage) has a service with an available REST API, which is used by the other services. The chosen programming language for each service is Java, mainly because it is the primary language used by the assisted living platform's current implementation. Also, Apache Tomcat is being used for the web server and PostgreSQL for the databases. REStEasy is the framework used to help build the RESTful Web Services.

## 5.1. Decisions

To test those services, Swagger UI is being used as UI for the WebAL service. That project includes HTML, CSS, and JavaScript code. This code was customized to perform encryption and decryption operations over some data on the client-side. To allow executing cross-domain requests, the Tomcat CORS filter was used in each service.

### 5.1.2 Authentication

The need for access control at the application level in this scheme was already explained in section 3.2.1. So, authentication and permissions validation (authorization) must be performed by all involved entities that may allow accessing or manipulating private data. However, the same credentials must not be used to access all services. If that were the case, the semi-trusted entities WebAL and Security Storage would be able to access data from the trusted entity KGC, which should only be accessed by the user himself. Nonetheless, it cannot be expected for the user to be willing to authenticate differently on the three involved entities to use the assisted living platform.

So, developing an application that uses different services, requiring the secure identification of users in several of them, raises the need to centralize their authentication systems for better management and security. Because the KGC already needs to be a trusted entity, the authentication process will be delegated to it. Currently, according to Karaki (2019), the three major protocols for federated identity are SAML, OAuth2, and OIDC.

Security Assertion Markup Language (SAML) is an XML-based open-standard used for Single Sign-On (SSO) implementations. SAML is used for authentication and authorization between two parties: a Service Provider (SP) and an Identity Provider (IdP). The SP agrees to trust the IdP in the authentication process. This is done through a SAML XML document containing the user authorization and authentication sent by the IdP to the user once he has proven his identity. That document is then redirected to the SP.

OAuth2 is an open standard used to provide applications with delegated authorization. Unlike other frameworks that offer authentication, OAuth only authorizes devices, services, or servers, using access tokens rather than credentials. It works over HTTPS and defines four roles:

- Resource Owner – Generally the user;
- Resource Server – Server hosting protected data;
- Client – Application requesting access to a resource server;
- Authorization Server – Server issuing the access token to the client.

When the client needs some information from the resource server, which trusts the authorization server in the authorization process, the user is redirected from the client page to the authorization server page. If the user authorizes access, the authorization server sends an authorization code to the client in the callback response. Then, the client will exchange that code by an access token with the authorization server. That token will be used to access the user data on the resource server.

OpenID Connect (OIDC) is a simple identity layer on top of the OAuth 2.0 protocol that allows for federated authentication. The OIDC process flow is similar to the OAuth2 authorization flow, with the major difference being that the authorization code is exchanged by an access token and an id-token that allows the user authentication. The authorization server response format on both protocols is JSON.

Between SAML and OIDC, SAML is the most mature protocol, being trusted by many organizations. OIDC, being newer and evolving, is still lagging behind SAML in terms of features (Auth0, nd). However, for many applications where there is a simple requirement for basic identity data, particularly in the consumer space, OIDC is very attractive, as it is far easier to use than SAML and doesn't require the XML handling heavyweight. So, OIDC or OAuth2 with a custom authentication mechanism were the chosen protocols for this scheme's authentication and authorization process, with the KGC assuming the role of an authorization server.

An overview of OAuth2 and OIDC implementations was carried out in July 2020, taking into account aspects such as useful tutorials, maintenance, popularity, activity in the community, and type of license. Some implementations were immediately discarded due to reduced maintenance (nowadays)<sup>1</sup>, such as Apache Oltu, Curity OAuth, MITREid Connect, and OpenId Connect for Shibboleth Identity Provider. Other discarded implementations were the Connect2id Server because of the license type, as well as the Oracle Access Management and the Authlete since their authentication and authorization processes were provided as an external service.

The next attributes to be considered were popularity and activity in the community, considering GitHub stars, Maven usages, and the number of questions asked and answered on StackOverflow. The implementations considered more interesting given those parameters were Spring Security, Connect2id (Nimbus), and Keycloak. A deeper analysis was also carried out on those implementations, and Keycloak was the chosen one because it is the most complete and allows for quick integration in the project.

Keycloak is made available in different environments, such as directly on OpenJDK or with Podman, Openshift, Docker, or Kubernetes. The OpenJDK version was used so that later the Keycloak code could be customized if needed. The Keycloak runs by default on a WildFly server. It includes Java code for the

---

<sup>1</sup> Based on the latest update on GitHub and the last libraries release on Maven.

backend and HTML, CSS, and JavaScript for the frontend. Freemarker is used to create templates with HTML code and some variables, which will be replaced by values from the Java code. The Freemarker templates are the ones that will be customized with JavaScript code to perform additional client-side operations on login and password updating.

Keycloak comes with its own embedded Java-based relational database called H2. This is the default database that Keycloak uses to persist data and only exists so that the authentication server can be initialized out of the box. It is highly recommended to replace the H2 database with a more production-ready external database. PostgreSQL is the object-relational database system chosen for this purpose.

### 5.1.3 Cryptography

#### **Hash Functions**

Arias (2019) states the importance of protecting user passwords using the following sentences:

*“ A strong password storage strategy is critical to mitigating data breaches that put the reputation of any organization in danger. Hashing is the foundation of secure password storage. ”*

In cryptography, a hash function is a mathematical algorithm that maps data of any size to a bit string of a fixed size. Hash functions used in cryptography have the following fundamental properties:

- It's easy and practical to compute the hash but difficult or impossible to re-generate the original input if only the hash value is known.
- It's difficult to create an initial input that would match a specific desired output.

Thus, in contrast to encryption, hashing is a one-way mechanism. The data that is hashed cannot be “unhashed” in a practical way.

There are many well-known hashing algorithms, such as MD5, SHA-1, SHA-2, SHA-3, NTLM, RIPEMD, and WHIRLPOOL. However, fast to compute hashes make it possible to determine user passwords using brute-force. So, password hashing requires slow to compute and long hashes, as well as the usage of a salt value to prevent rainbow table attacks.

Several hash algorithms can be considered for password hashing (Preziuso, 2019), such as:

- Password-Based Key Derivation Function 2 (PBKDF2) – This algorithm exists for a long time, but it is easily parallelized on multi-core systems (GPUs) and is trivial for tailored systems – Field-Programmable Gate Arrays (FPGAs)/Application-Specific Integrated Circuits (ASICs).

- BCrypt – It has been out there since 1999 and does a better job at being GPU/ASIC resistant than PBKDF2, but it doesn't shine in a threat model with offline cracking. It hasn't gained a lot of popularity and consequentially enough interest from the FPGA/ASIC community to build a hardware implementation. Also, Malvoni et al. (2014) have written a paper describing a hybrid system of Advanced RISC Machine (ARM)/FPGA System On Chips (SOCs) to attack the algorithm.
- SCrypt – It has been in the field for 10 years and has a better design than BCrypt, especially regarding memory hardness. SCrypt has been used for many cryptocurrencies, and there are a few hardware (both FPGA and ASIC) implementations of it.
- Argon2 – It won the Password Hashing Competition in July 2015. Argon2 is particularly resistant to ranking tradeoff attacks, making it much more difficult to cheaply optimize on FPGAs: even though recent FPGAs have embedded RAM blocks, memory bandwidth is still a constraint.

Taking into account all the previous considerations, Argon2 was the chosen algorithm for hashing passwords. There are three variants of Argon2: Argon2i, Argon2d, and Argon2id. The differences are described by Biryukov et al. (2020):

*“ Argon2d uses data-dependent memory access, which makes it suitable for cryptocurrencies and proof-of-work applications with no threats from side-channel timing attacks. Argon2i uses data-independent memory access, which is preferred for password hashing and password-based key derivation. Argon2id works as Argon2i for the first half of the first pass over the memory, and as Argon2d for the rest, thus providing both side-channel attack protection and brute-force cost savings due to time-memory tradeoffs. Argon2i makes more passes over the memory to protect from tradeoff attacks. ”*

So, Argon2i is being used for password hashing on the server-side in this project. It will also be used as a password-based KDF on the client-side, with the purpose already explained in section 4.2. According to Maddox and Moschetto (2019), if a modern and secure hashing algorithm is used, repeated hashing does not reduce entropy.

Regarding the Argon2 implementation, it is needed for both Java on the server-side and JavaScript on the client-side. The used Java implementation is the one provided by Kammerer (2020), and the JavaScript is the one supplied by Witkowski (2020).

## **Signatures**

Some operations in the implemented scheme will have the non-repudiation characteristic, so it will be necessary to sign some data with the user's private key on the client-side. There are several cryptographically secure digital signature schemes available (Nakov, 2019), the most popular being presented below:

- RSA – Public-key cryptosystem based on the math of modular exponentiations and the difficulty of the Integer Factorization Problem (IFP).
- Digital Signature Algorithm (DSA) – Based on the math of modular exponentiations and discrete logarithms, as well as the difficulty of the Discrete Logarithm Problem (DLP).
- Elliptic Curve Digital Signature Algorithm (ECDSA) – Relies on the math of the cyclic groups of elliptic curves over finite fields (ECC) and on the difficulty of the Elliptic-Curve Discrete Logarithm Problem (ECDLP).
- Edwards-curve Digital Signature Algorithm (EdDSA) – Fast digital signature algorithm that uses elliptic curves in Edwards form, like Ed25519 and Ed448-Goldilocks, and relies on the difficulty of the ECDLP problem.

All the above-mentioned signature schemes are quantum-breakable, which means that powerful enough quantum computers may calculate the signing key from the public key. Quantum-safe signatures, such as CRYSTALS-Dilithium, FAsT-Fourier Lattice-based COmpact Signatures Over NTRU (FALCON), and Rainbow, are not massively used, because of long key length, long signatures, and slower performance compared to ECDSA and EdDSA.

Both DSA and ECDSA require a parameter usually called  $k$  to be completely random, secret, and unique. In practice, that means that if the user machine has a poor random number generator and the same  $k$  happens to be used twice, for example, an observer of the traffic can figure out the user's private key. A deterministic DSA and ECDSA variants are defined in RFC 6979, which calculates the random number  $k$  as the Hash-based Message Authentication Code (HMAC) from the private key, the message hash, and few other parameters. Those deterministic versions are considered more secure.

The last decade trend is to move from RSA and DSA to elliptic curve-based signatures, like ECDSA and EdDSA. Modern cryptographers and developers prefer ECC signatures for their shorter key length for the same security level, shorter signature, and better performance.

Between these schemes, EdDSA is more simple, more secure, and designed to be faster than ECDSA for curves with comparable key length. So, Ed25519 (EdDSA) was the selected scheme to implement these project signatures. The chosen implementation for Java is the one provided by Bouncy Castle and for JavaScript is the one supplied by *Forge*.

All signatures will include the generation timestamp to register the moment the operation was performed.

## **Certificates**

The KGC will issue certificates to authenticated users, aiming to ensure other entities that the user's public key is the one registered in the certificate. These certificates will also indicate the user's permissions, avoiding multiple future communications between the other entities and the KGC to validate the authenticated user's permissions, thus improving the scheme's overall performance.

These certificates make use of a customized structure presented in Figure 13. It will consist of the user (owner) identifier, public key, permissions, access structure, and the certificate issue and expiration date. The certificate will also have this information signature, using the previously chosen algorithm Ed25519.

The certificate permissions attribute will be a structure containing several data types, each related to multiple operations. In turn, each operation will be associated with a list of policies. Each list of policies will determine which users can perform the respective operation over the associated data type. Both the data type and the operation are enumerations whose values are specified in Figure 13.

Regarding the policies list for a particular data type  $DT$  and some operation  $O$ , all policies from that list must be verified so the certificate owner can perform the operation  $O$  over data with the type  $DT$ . There will be three available policy types, according to the assisted living platform needs. Each policy type is associated with a different kind of permissions list:

- Profile Policy – Related to a list of profiles. The profile is an enumeration whose values are specified in Figure 13. This policy is verified if the data in question is about a user with one of the profiles indicated by this policy. E.g.: The creation of new entities.
- Profile Pair Policy – Related to a list of profile pairs. This policy is verified if the data in question is about two users whose profiles match one of the profile pairs indicated by this policy. E.g.: The creation of a new relationship between two users.
- Related Ids Policy – Related to a list of user identifiers (strings). This policy is verified if the data in question is about a user with one of the identifiers listed by this policy. E.g.: Access to the patient measurements is only possible between related users.

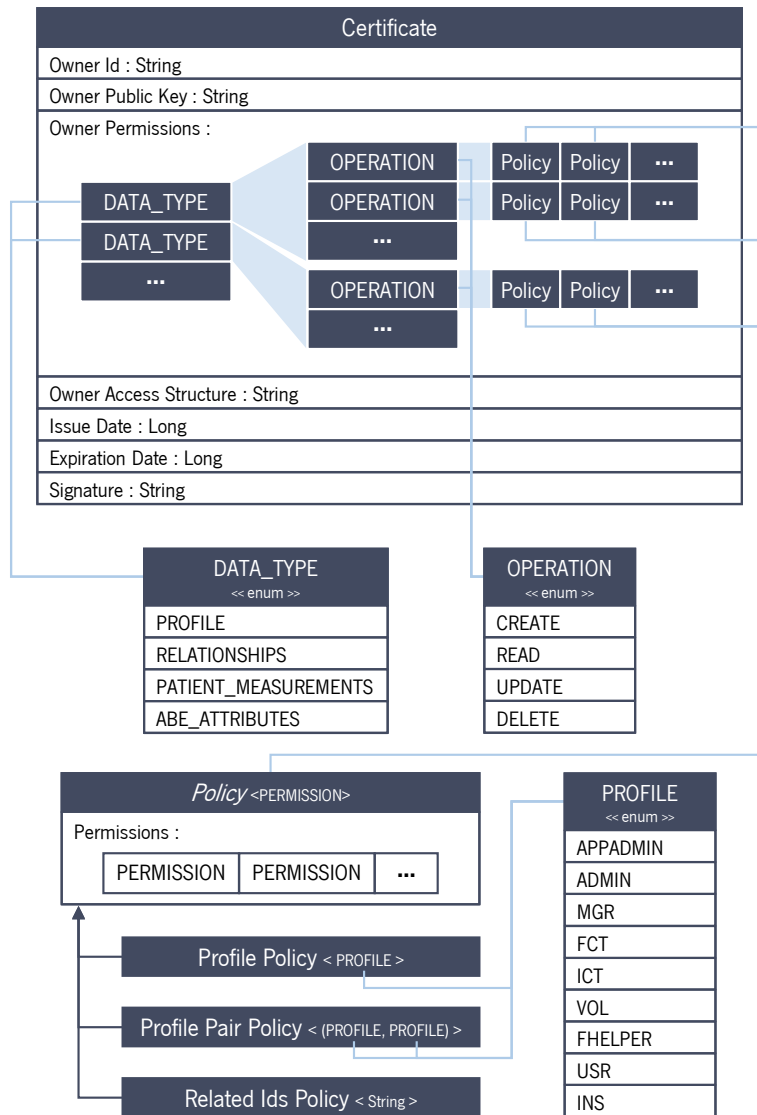


Figure 13: KGC certificate structure.

It should be noted that if there are many relationships regarding one user (e.g., a doctor with many patients), that user could have a large certificate. So, this way of specifying users' permissions should be evaluated and compared with other possible solutions, such as the communication with the KGC every time there is the need to validate the user's permissions. However, this analysis is outside the scope of this project.

So, on login, the KGC issues the authenticated user certificate to the WebAL client-side, which sends it to the WebAL Server and Security Storage. These servers store the certificate as a session variable if the authenticated user identifier matches the certificate owner identifier and the certificate signature is valid. That certificate is needed to validate signatures sent by the authenticated user, as well as his permissions to perform certain operations on subsequent requests. Therefore, this process avoids sending the certificate with each request.



## Encryption

The solution to be implemented includes multiple encryption schemes that will be used over different types of information:

- AES-GCM – AES is one of the most widely used symmetric encryption schemes, and it is one of the two block cipher techniques currently approved by NIST. The Galois/Counter Mode (GCM) is specified by NIST as a high-throughput authenticated encryption mode.

This scheme is used to encrypt sensitive data over which no operation needs to be performed by the server that stores it. On the client-side, AES-GCM is used to encrypt users' private data stored on WebAL and signature private keys stored on the KGC. On the server-side, it is used to cipher the signature private key of the KGC administrative user. The chosen implementation for JavaScript on the client-side is the one supplied by *Forge* and for Java on the server-side is the one provided by *Java Cryptography Architecture (JCA)*.

- KP-ABE – It will be used to encrypt the symmetric keys used for data encryption. The chosen scheme will have to be used on the server-side (Java) and client-side (JavaScript). There are several available KP-ABE implementations and no standard established. So, the available algorithms will be discussed and compared next to find the better choice.
- ORE – It is a Symmetric Searchable Encryption cryptographic primitive which allows for efficient range queries, sorting, and threshold filtering on encrypted data. This primitive could be used to compare the patient measurements with certain threshold values on the server-side to alert the doctor if some values are out of the usual range. The most popular ORE implementation is *FastORE*, which implements the schemes “Practical Order-Revealing Encryption with Limited Leakage” and “Order-Revealing Encryption: New Constructions, Applications, and Lower Bounds”, respectively named *FastORE* and *FastORE\_BLK*.

The *FastORE\_BLK* scheme operates on blocks (where a block is a sequence of bits), and the additional leakage is the position of the first block in which two messages differ. For instance, if blocks are byte-sized (8 bits), then the *FastORE\_BLK* scheme only reveals the index of the first byte that differs between the two messages. In contrast, the *FastORE* construction always reveals the index of the first bit that differs. Thus, *FastORE\_BLK* construction provides significantly stronger security at the cost of longer ciphertexts.

So, *FastORE* is the implementation chosen to integrate into this project. It will be used on the server-side (Java) to compare encrypted measurement values and on the client-side (JavaScript) to

encrypt and decrypt measurement values. This implementation is written in C language, so *Java Native Interface* will be used to integrate the library with Java. Regarding the JavaScript integration, the available choices are discussed below.

Regarding the KP-ABE implementations, a deeper search was needed. This search included parameters such as the used programming language, the repository's last update, and the popularity (measured by the project GitHub Stars). The main KP-ABE implementations found are presented in Table 3.

Table 3: ABE implementations.

Implementation	Scheme	Language	Last Update	GitHub Stars
<i>KP-ABE YCT14</i>	“A lightweight attribute-based encryption scheme for the Internet of Things”	C++	26/01/2018	10
<i>OpenABE</i>	Implements several schemes. KP-ABE scheme: Key Encapsulation Mechanism (KEM) variant of the “Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data” scheme.	C++	05/10/2020	123
<i>Rabe</i>	Implements several schemes. KP-ABE schemes: “FAME: Fast Attribute-based Message Encryption” and “Revocation Systems with Very Small Private Keys”.	Rust (with C bindings) <sup>2</sup>	27/08/2020	34
<i>upb.crypto.craco</i>	“Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data”	Java	05/10/2020	2

Continued on next page

<sup>2</sup> There is also a REST API version available: <https://github.com/Fraunhofer-AISEC/rabe-keyserver>.

Table 3: ABE implementations (continuation).

Implementation	Scheme	Language	Last Update	GitHub Stars
	Implements several schemes.			
<i>Predicate Based Encryption Library</i>	KP-ABE scheme: “ <i>Revocation Systems with Very Small Private Keys</i> ”.	Python	24/10/2016	17

No KP-ABE JavaScript implementation was found. So, it was necessary to search how to use the presented implementations with the intended languages (JavaScript and Java).

Regarding JavaScript, other languages’ code can be used if compiled to WebAssembly (Wasm). C/C++ code can be compiled to Wasm using *Emscripten* or *Wasienv*, being the first one more popular according to GitHub stars. Rust code can be compiled to Wasm using *Wasm-Pack*. Python can be transformed into JavaScript using *Transcrypt*. Any language code could also be used in JavaScript if there was a server written on that language providing the needed services through a REST API. However, because those services need to be performed on the client-side, that server would have to be running on the client machine. These are just possible solutions, being the most interesting one the compilation to WebAssembly, since it suits better the proposed solution architecture, does not require extra components, and is more reliable.

Regarding Java, both C/C++ and Rust code can be used through the *Java Native Interface*. Python code can be called from Java using *jython* or *jvabridge*, for example.

So, it should be possible to use all referred implementations based on the programming language parameter. However, some of them must be excluded due to reduced current maintenance. Among the other implementations, the most popular one was chosen, which is the OpenABE library. This library will be integrated with the project using *Java Native Interface* for server-side usage (Java) and *Emscripten* for client-side usage (JavaScript). It should be noted that if a problem occurs when compiling OpenABE using *Emscripten*, the usage of a REST service running on the client-side, for example, is still possible.

Another decision regarding KP-ABE was choosing the access structures to be used on the users’ ABE keys generation. Taking into account the assisted living platform profile hierarchy and each profile permissions, the most suitable access structures for each profile are the ones presented in Figure 14.

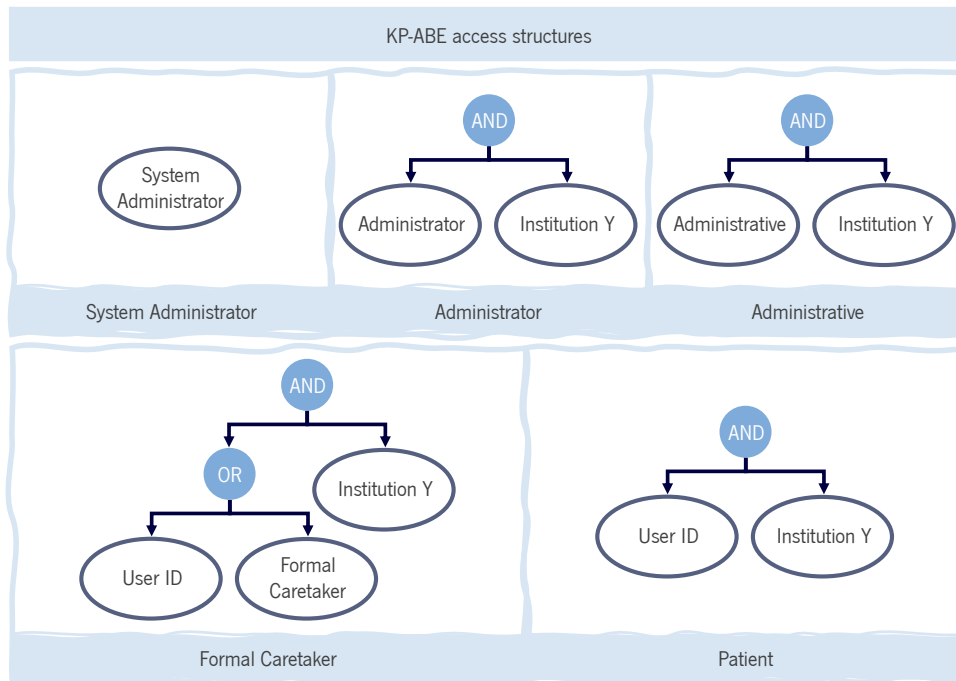


Figure 14: KP-ABE access structure for each profile.

The System Administrator profile is responsible for managing all institutions and Administrators. So, this is the only entity whose access structure is not dependent on the associated institution. Administrator and Administrative entities can create other entities inside their related institution. However, they are not related to any patient, so their data access will be conditioned by their profile but not their identifier. Formal Caretakers can access data inside their institution. If that data has the Formal Caretaker profile as an attribute, it is accessible to all Formal Caretakers. If it has their identifiers as attributes, the data is only accessible to the respective Formal Caretakers. Patients can only access data specifically authorized for them, inside their institution.

Summing up, AES-GCM will be used to encrypt data that won't be subject to server operations, FastORE to encrypt data that the server will need to compare with other values, and OpenABE to cipher the keys used in both those encryption schemes.

### **Key Management**

Both server and user keys must be securely managed. The former will be stored and used on the server-side, while the latter will be stored on the server-side and used on the client-side.

On the server-side, the most secure way of managing the server keys would be to use a Hardware Security Module (HSM). However, many vendors fail to disclose specifics about their HSM solutions, which require physical access for deployment, maintenance, and configuration. Besides, these devices can be rather expensive, depending on their level of functionality and security. This approach won't be used

for the proposed scheme's implementation since open-source solutions without associated costs will be privileged.

For HSM simulation and testing, *SoftHSM* could be used, for example. It is an open-source software implementation of a generic cryptographic device with a PKCS#11 interface. This interface specifies how to communicate with cryptographic devices, such as HSMs and smart cards. However, in a production environment, an HSM should be used to achieve the intended security level. Because *SoftHSM* by itself does not provide additional security regarding other software-based solutions, a simplified approach, without additional dependencies, was adopted for this prototype.

So, the KGC private keys, for signature and ABE, will be stored after encryption with AES-GCM. The AES key will be derived from an administrator password. That password and the decrypted keys should only exist in memory. The password will be placed in an environment variable from the Tomcat server, and the keys will be read and decrypted once on the respective Java class load.

It should be noted that there are different concerns regarding these keys, as far as the ability to recover them in case of a possible loss. The inability to recover the KGC signature private key will only require the generation of a new signature key pair, issuance of new user certificates, and replacement of the KGC signature public key on all entities. However, the KGC ABE master private key's loss would make it impossible to generate new user ABE keys and, consequently, to decipher the keys used to encrypt data and the data itself. In other words, this would cause the loss of all data. So, precautions such as data backups must be taken to prevent keys loss, particularly regarding the KGC ABE master private key.

The users' signature private keys will always be encrypted on the KGC server and will only be decrypted on the client-side. The KGC server must ensure that no user keys are lost, through redundancy and frequent backups, for example. Other crucial security properties must be verified on the client-side regarding the user's signature and ABE private keys, such as confidentiality and integrity. There are several ways of storing data on the client-side (browser), each one with its advantages and disadvantages:

- HTTP Cookie – Small piece of data stored on the user's computer by the web browser while visiting a website. The cookie is sent to the server on each client's request, which would compromise the non-repudiation characteristic if used to store the decrypted private keys. Also, cookies accessible via JavaScript do not protect sensitive information from XSS and Cross-Site Request Forgery (XSRF) attacks.

XSS attacks enable malicious entities to inject client-side scripts into web pages viewed by other users. A method of protecting cookies from such an attack is to use a flag that tells the web browser that the cookie can only be accessed through HTTP (HttpOnly flag). However, this solution does

## 5.1. Decisions

not apply to this scheme, given that the encryption and decryption operations that use the keys are performed by JavaScript on the client-side.

XSRF attacks allow users trusted by the web application to submit unauthorized commands. This type of attack is prevented by the Same-Origin Policy (SOP) and Cross-Origin Resource Sharing (CORS).

The most secure way of using cookies to store the private keys would be if those keys were encrypted so that even the KGC server would not be able to decrypt them. However, the same problem would remain regarding where to store the key derived from the user authentication credentials that allow decrypting the user's signature and ABE private keys.

- Web Storage – API that provides mechanisms by which browsers can store key/value pairs. The keys and the values are always strings and are never transferred to the server. There are two mechanisms within Web Storage: *sessionStorage* and *localStorage*. A different *Storage* object is used for the *sessionStorage* and *localStorage* of each origin. These objects operate and are controlled separately.

The *sessionStorage* mechanism stores data available for the duration of the page session, that is, until the browser or tab is closed. Its storage limit per domain is at most 5 MB, which is larger than a cookie. The *localStorage* does the same but persists data even when the browser is closed and reopened, with no expiration date. It only gets cleared through JavaScript or by clearing the browser cache or the locally stored data. The *localStorage* storage limit is usually larger than the *sessionStorage* one.

The *sessionStorage* mechanism has the disadvantage of not being able to share data between tabs located on the same origin. The *localStorage* has the disadvantage of not allowing to establish an expiration date for the stored data. Sensitive information should not remain on *localStorage* indefinitely. Even if an event listener is created to clean the *localStorage* when exiting the last opened tab on that domain, a system crash could result in sensitive information stored on *localStorage* until the user's next visit to the platform website.

Besides those independent disadvantages, the Web Storage API usage is frequently discouraged to store sensitive data because if the website is vulnerable to XSS attacks, Web Storage is not safe. However, that problem will exist with most storage approaches on the client-side since JavaScript must access the private keys to perform encryption and decryption operations. The most effective way to deal with this problem is to work on finding and preventing XSS vulnerabilities on the website.

- IndexedDB – Low-level API for client-side storage of significant amounts of structured data, including files/blobs. While Web Storage is useful for storing smaller amounts of data, it is less useful for larger amounts of structured data. Regarding keys storage, the Web Cryptography API has been referred to be used with IndexedDB for in-browser keys' security. An example is provided by Engelke (2014), who explains how securely store cryptographic keys in the browser:

*“ The Web Cryptography API can provide those safeguards by keeping private keys opaque. That is, the API lets applications create and use keys without ever being able to see their actual values. If code inside the browser can't see the values, it can't disclose them. All cryptographic keys are stored as a CryptoKey which does not provide access to their values, which are stored outside the browser environment. It is up to the browser vendor to make that storage as secure as possible; in any case, it is inaccessible from inside the browser.*

*There are cases where code inside the browser might need to access the actual value of key. [...] That was possible because the [...] key was created with its extractable property set to true. To keep stored keys secure, set their extractable property to false when they are created or imported.*

*Aside: the API always allows public keys to be exported, regardless of how they are created or imported. That's because public keys are not sensitive information. They are intended to be shared.*

*Because CryptoKey objects are opaque they can't be placed in localStorage, which only supports simple types. Instead the key store will use IndexedDB, which can clone and store opaque variables. ”*

So, this would be the most secure approach for the current use case. However, even if the keys cannot be exported, XSS attacks could still use those opaque keys to sign, encrypt, or decrypt data on the user machine. Also, the Web Cryptography API's latest published version is from 2017 and does not include the chosen password hashing algorithm Argon2 and the signature algorithm Ed25519.

- Operating System and Browser Certificate/Key Stores – Several operating systems and browsers provide certificate/key stores. These are software-based databases that store the user public/private key pair and certificate locally on the user machine (Yadav, 2017). Chrome and Internet Explorer both use the Windows Certificate Store, while Firefox uses its own certificate store by default

## 5.1. Decisions

(provided by Mozilla). This means that if a key is imported into the Windows Certificate Store, for example, Chrome and Internet Explorer will find the certificate, but Firefox won't. Also, in Chrome, for example, a Java applet or NativeClient SDK project would have to be built to access the Certificate Store from JavaScript.

Among these possible solutions, cookies won't be used because they would be shared with the KGC server. The Web Cryptography API with IndexedDB would be a more secure solution regarding key management, but the implemented algorithms are not the ones that are most advised today. So, this was not the chosen approach. The Operating System and Browser Certificate/Key Stores also won't be the used solution because of the need for different approaches for each browser and operating system to access the desired keys.

So, the chosen solution to store the keys on the client-side was the Web Storage, combining both its mechanisms to overcome the disadvantages of each one. The *localStorage* will be used on login to set the decrypted keys across all tabs. Those values are then immediately deleted from *localStorage*. A *storage* event triggered by the setting of the decrypted keys on the *localStorage* will be detected by all WebAL opened tabs, each of which sets those keys in its *sessionStorage*. So, the keys will exist on *sessionStorage* and won't exist on *localStorage* because they were immediately deleted.

If a new WebAL tab is opened, it just needs to request the user keys using the *localStorage*. When any opened WebAL tab detects this request and has the user keys, it sets the keys on the *localStorage*, deleting them next. Then, the new WebAL tab detects the keys set on *localStorage* and saves those keys on *sessionStorage*.

The user keys will be decrypted on login, but the login operation is performed by the KGC server. To allow communication between different domains, the WebAL server will allow KGC to access a specific HTML page using CORS. Upon successful user authentication, the KGC login page will open an *iframe* with that WebAL page and use the JavaScript method *postMessage* to send it the user keys. That page should only accept messages from the KGC origin. Then, the WebAL page will set the keys on *localStorage* and then delete them.

Using this approach, there is only a very short period between insertion and removal in which the user keys can be persistently registered on *localStorage* if the client machine crashes. Also, XSS vulnerabilities can still exist, so finding and preventing them in a real environment is required. However, this won't be addressed for the prototype.



## 5.2 IMPLEMENTATION

This section will describe the implemented scheme, including the system architecture, the data stored by each entity, and the main operations communication flow.

### 5.2.1 System Architecture

A diagram of the proposed solution architecture, with the used technologies for each entity, is presented in Figure 15. These technologies are the ones analyzed and justified in the previous section.

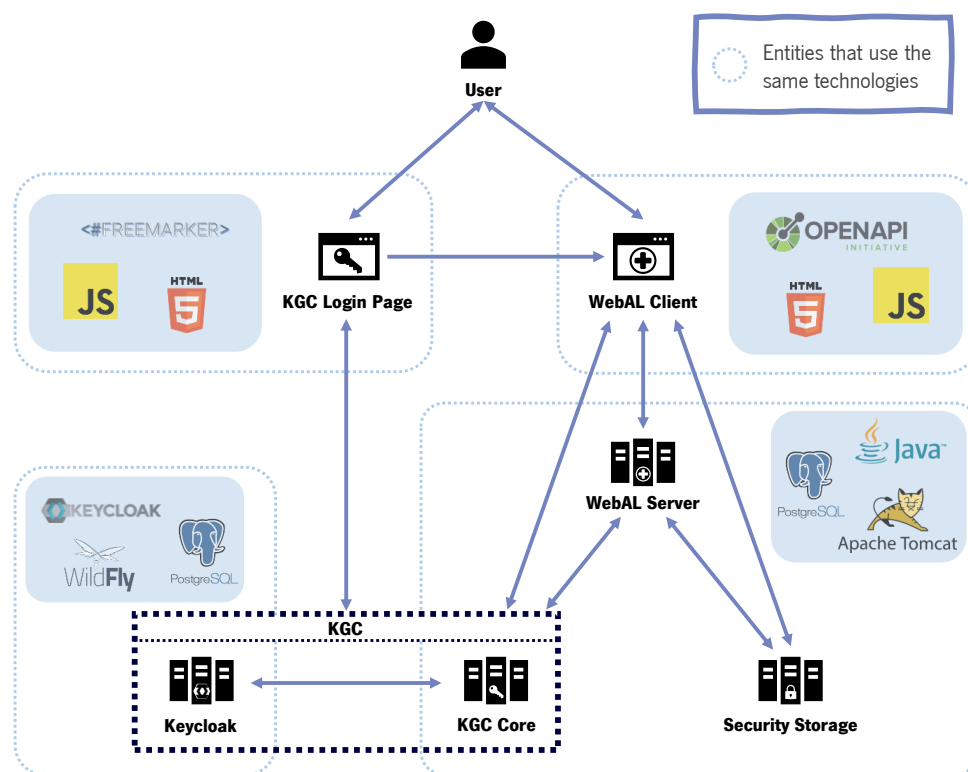


Figure 15: System architecture.

So, this solution includes the KGC, the WebAL, and the Security Storage entities. The KGC consists of the Keycloak and the KGC Core servers, the former providing a page where the user logs in. All other user interactions occur over the Swagger UI API page (WebAL Client) provided by the WebAL Server. Regarding the integration of the OpenABE library with the WebAL Client, it was accomplished by using Emscripten to generate WebAssembly.

The KGC login page mainly communicates with the KGC entity for authentication and to get the user's private keys and certificate. It also sends the decrypted user private keys and the certificate to the WebAL Client after login.

## 5.2. Implementation

The WebAL Client communicates with all three entities KGC, WebAL Server, and Security Storage. The WebAL Server also communicates with the other two entities, KGC and Security Storage. On user registration, for example, the WebAL Client communicates only with the WebAL Server, which will then take care of the user insertion in the KGC and the Security Storage entities, ensuring the data consistency between entities. On relationship insertion, the WebAL communicates directly with the KGC to create the relationship and with the Security Storage to update the related ABE data attributes. There is no need for the WebAL Server to know the existent relationships once the permissions validation is ensured by the authenticated user certificate validation. So, the information about the addition of a relationship is not sent to the WebAL Server.

### **Keycloak Configuration**

The Keycloak server was configured to use a PostgreSQL database for data storage. For the intended platform, a realm named *assisted\_living* was created on Keycloak. An email account was configured to allow, for example, sending emails for users to update their passwords. A custom password policy was also added to Keycloak to enable using the hashing algorithm Argon2 for password storage.

This realm default web pages (themes) were customized to perform some cryptographic operations on the client-side upon successful authentication, such as the password hashing, the encryption and decryption of the user's signature private key, and the generation of the signature key pair on the first login. There was also the need to create new clients inside the *assisted\_living* realm, such as *KGC*, *KGC-keycloak-auth*, and *WebAL*. All these clients use the protocol OpenId Connect for authentication and authorization.

The first client is the KGC Core, which uses the Keycloak Admin REST API for users' management. For that, the KGC Core authenticates itself on Keycloak as a realm administrator user, created for that purpose. This user, named *kgc\_admin*, is related to the roles *manage-users*, *view-realm*, and *view-users* from the default client *realm-management*. Besides the realm administrator credentials, the *KGC* client must use its client credentials since its access type is configured as confidential. In the future, the servers Keycloak and KGC Core could be merged into one to avoid the need to resort to this API to perform the users' management, once both servers are part of the same entity.

The *KGC-keycloak-auth* client corresponds to the web pages made available by Keycloak. This client is used by the login and reset password pages to communicate with the Keycloak and the KGC Core servers to perform the needed operations, such as getting or setting the user's signature key pair and getting the ABE key. However, the *KGC-keycloak-auth* client is used on the client-side, so its access type is public, and

the utilized credentials are the ones inserted by the user who is performing the authentication. This client has a special scope named *user\_keys*, which allows it to set or get the users' private keys.

The *WebAL* client is used by the WebAL Swagger UI API page to perform the user authentication based on OpenId Connect. This client's access type is also configured as public.

The users' roles are also defined in the realm context. Some examples are *APPADMIN* (system administrator), *ADMIN* (administrator), *MGR* (administrative), *FCT* (formal caretaker), and *USR* (patient).

### 5.2.2 Data Storage

This section will present the data stored by each entity. The KGC entity includes the Keycloak and KGC Core servers. While the Keycloak server stores data such as username, password hash, and profiles, the KGC Core stores the user's access policy, encrypted private key, public key, and relationships. To ensure the non-repudiation characteristic of the relationship creation, the user performing the operation generates a signature with the relationship data, which is stored on the KGC Core. This characteristic is important since the relationship addition will allow the involved users to access and decrypt data about each other. The physical Entity Relationship Diagram (ERD) of the KGC Core database is presented in Figure 16.

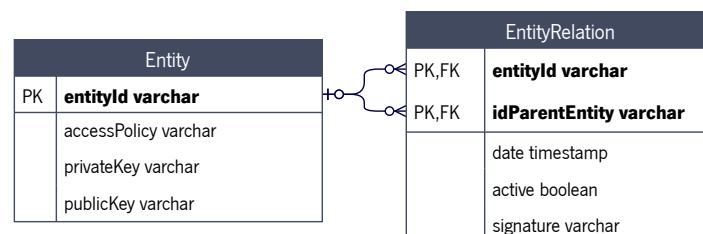


Figure 16: KGC Core database structure.

The Security Storage saves information about the ABE attributes and encrypted keys for each user and data type. The users' profiles are also stored to validate their permissions using the certificates issued by KGC. A key log is also kept to register every key access. All Security Storage data will have the non-repudiation characteristic, which means that the user will have to sign every request sent to the Security Storage and that signature will be stored. However, on user creation, the administrative won't have access to the user's KGC identifier before creating it on KGC. Also, the user creation operation is performed by the WebAL Server on both the KGC Core and Security Storage, as explained before. In this special case, the user creation signature will be provided by the KGC, which is a trusted entity, after creating the user. The physical ERD of the Security Storage database is presented in Figure 17.

## 5.2. Implementation

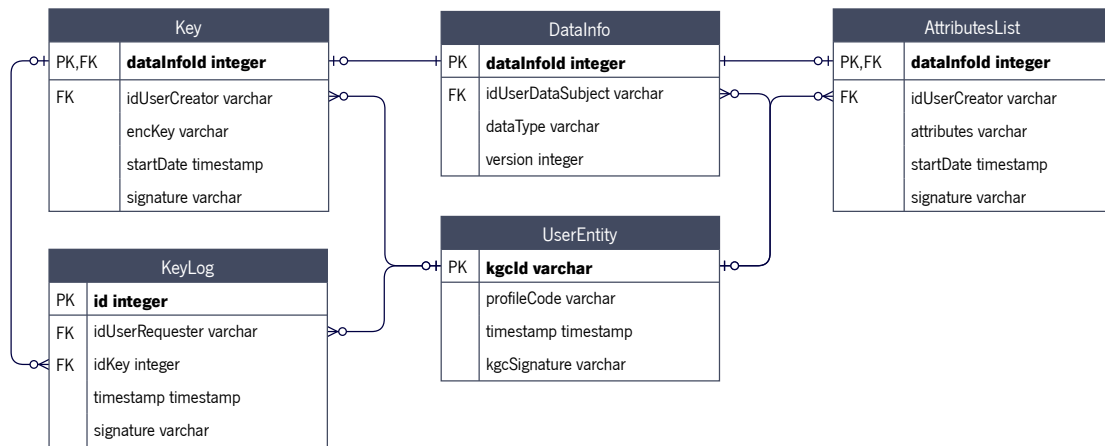


Figure 17: Security Storage database structure.

The WebAL Server stores users' personal data, such as username, date of birth, height, gender, KGC identifier, profile, and health measurements. The physical ERD of the WebAL database is presented in Figure 18.

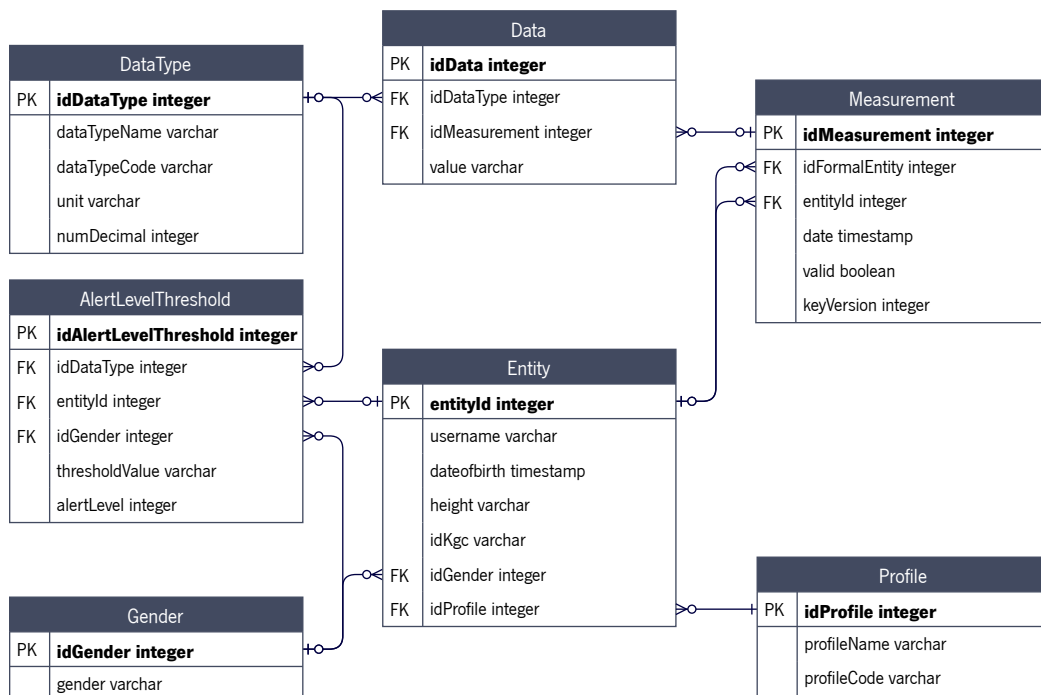


Figure 18: WebAL database structure.

The user's KGC identifier is kept in all entities' database so the same user can be identified between them. The WebAL database contains other duplicated data from the KGC entity, such as username and profile, to reduce the number of requests made to KGC.

### 5.2.3 Communication Flow

In this section, for each main operation, the communication flow between entities will be presented. Each communication flow assumes that every operation is successfully performed. If an error occurs, the respective message would be propagated to the end-user. The main operations are user login, user registration, relationship registration, data encryption, and data decryption.

Regarding the login operation, the sequence diagram representing its communication flow is presented in Figure 19. The black rectangles on that diagram represent the KGC Core, and the grey rectangles represent the Keycloak, both components of the KGC entity.

The login operation on the WebAL Client starts with the user being redirected to the KGC login page, where he inserts the username and password. The KGC login page then generates the password hash, using the Argon2 algorithm but with a deterministic value instead of a salt. The usage of a deterministic value, which consists of the application name and the username, aims to avoid an additional server request to get the previously used salt. This approach is considered adequate because the goal is to prevent equal credentials on different application servers when the user adopts the same credentials on several applications, as well as not to let the KGC know what password was used to encrypt the private key.

The generated hash becomes the password sent to Keycloak for authentication, alongside the username. If the user authentication form submission occurs successfully, the KGC login page requests Keycloak the access token for the *KGC-keycloak-auth* client. The KGC login page then requests the user's private keys and certificate to the KGC Core or creates the signature key pair if it doesn't exist, always using the access token for authentication. The user signature private key from the KGC Core is encrypted, so it is then decrypted using a password hash, generated with a new salt value. The user certificate, as well as the signature and ABE private keys, are sent to the WebAL Client for storage on the client-side (*sessionStorage*).

Then, the KGC login page redirects the user to the WebAL Client with the authorization code, which is exchanged with Keycloak by the WebAL client access token. This token is used as authentication in all entities for the following operations. Then, the WebAL Client sends the user certificate for the WebAL Server and the Security Storage, each of which stores it as a session variable.

Regarding the logout operation, it consists of cleaning the *sessionStorage* and sending a logout request to the WebAL Server, which sends the same request to the KGC Core and the Security Storage. Both the WebAL Server and the Security Storage invalidate the current session on logout, unbinding the associated certificate. The KGC Core performs the logout operation on Keycloak, destroying the respective session.

## 5.2. Implementation

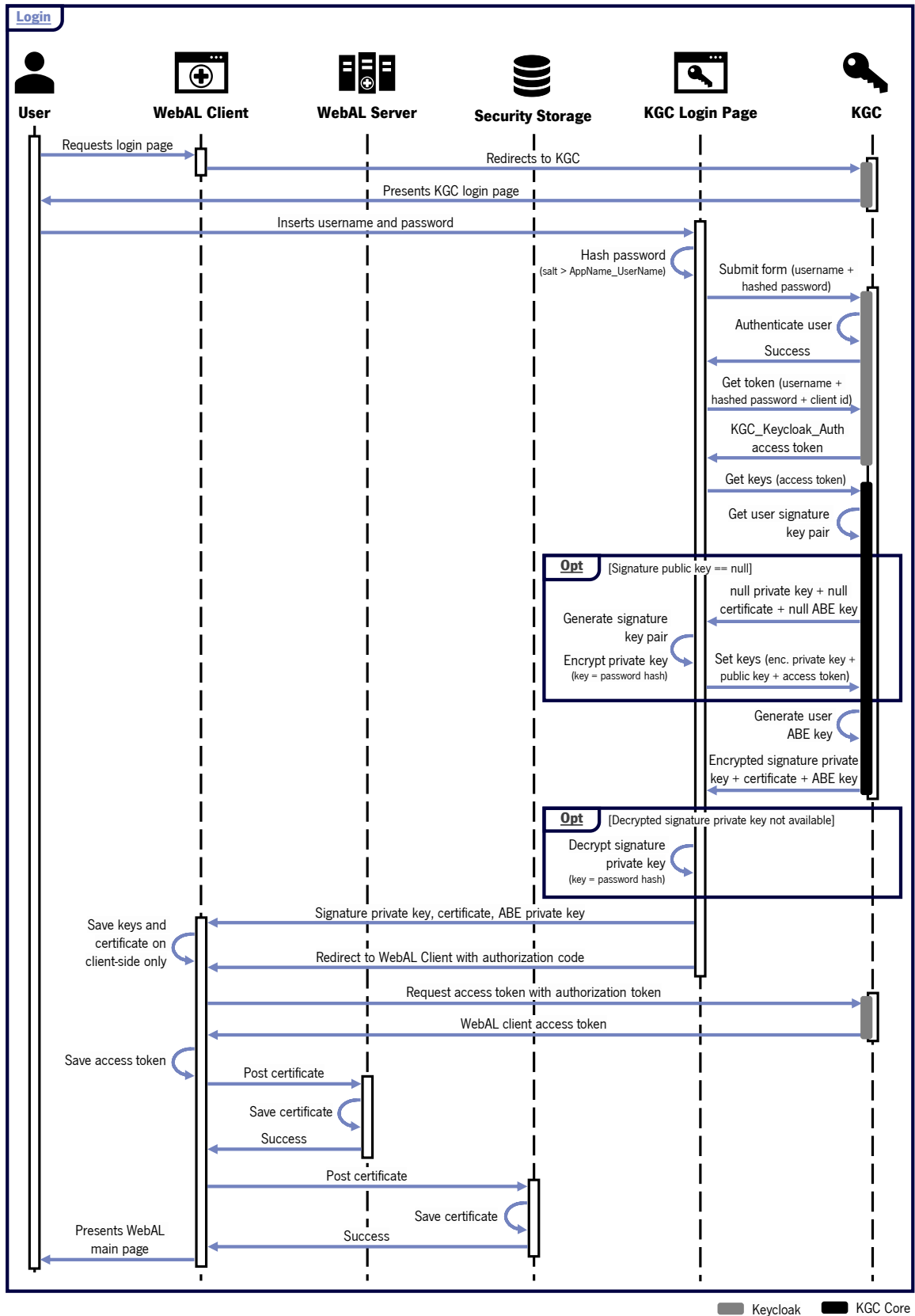


Figure 19: Login sequence diagram.

Regarding the user registration, the sequence diagram representing its communication flow is presented in Figure 20.

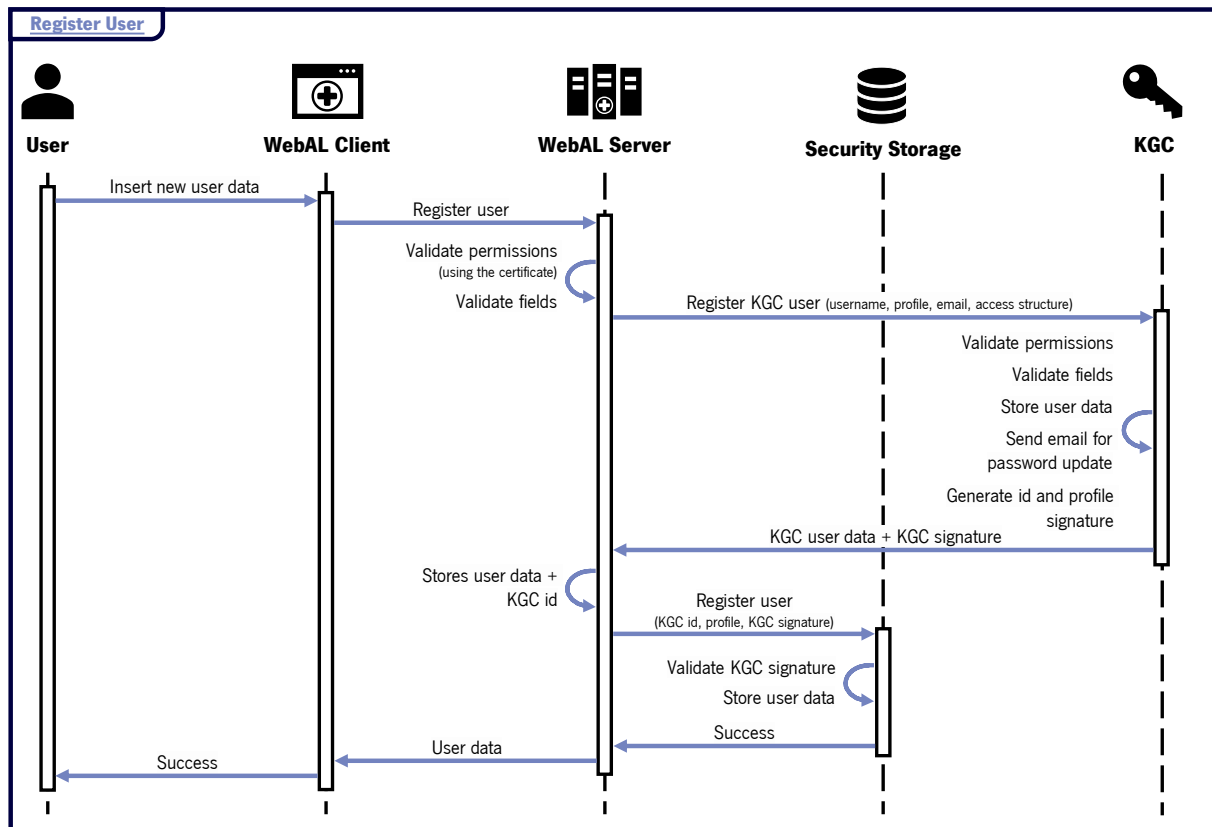


Figure 20: User registration sequence diagram.

The user registration operation is initiated by an administrative user, who inserts the new user data on the WebAL Client. The WebAL Client registers the new user on the WebAL Server, which validates the authenticated user permissions using his certificate and the new user inserted data. Then, the WebAL Server registers the needed data about the new user on the KGC, from which it obtains the new user KGC identifier and a KGC signature confirming the insertion of the new user. The WebAL Server registers the new user data and the KGC identifier on its database and sends the new user's required data and KGC signature to Security Storage. After Security Storage registers the new user, a successful response is returned to the WebAL Client and the authenticated user.

It should be noted that if the user registration fails on any entity, the ones that previously had created that user will revert the operation and a failure response will be returned to the user.

The user creation is an operation that involves all three entities and must be led by one of them that is responsible for ensuring that the new user is created in all three entities or none of them. The WebAL Server oversees this operation because it must have access to the majority of the inserted information, and the data that is not intended for WebAL can be known by that entity. So, WebAL is the entity that delegates the user creation on KGC and Security Storage, as well as removes the user from the entities where he has already been inserted if an error occurs during the user creation.

## 5.2. Implementation

Regarding the relationship creation, the sequence diagram representing its communication flow with successful responses is presented in Figure 21.

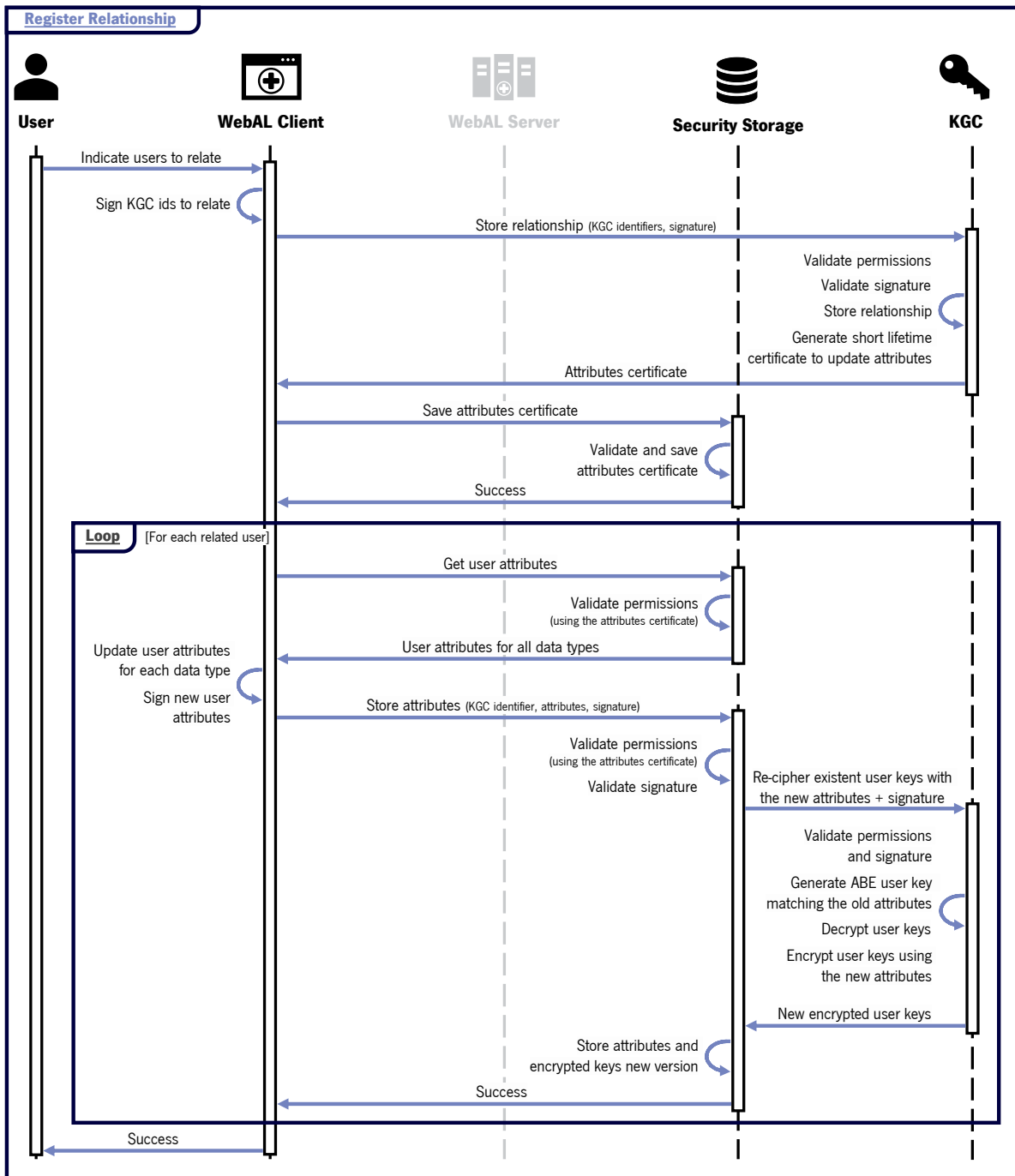


Figure 21: Relationship registration sequence diagram.

The relationship creation does not include the WebAL Server because there is no need for this entity to access that data. Because this operation only involves the other two entities, there is no need for the relationship creation to be led by one of them. The relationship is first created on the KGC. If that insertion succeeds, KGC will issue a short lifetime certificate authorizing the authenticated user to update ABE attributes about the related users. That attributes certificate is then used by Security Storage to validate



the user permissions when the WebAL Client updates the related user attributes. So, if the relationship insertion on KGC fails, the WebAL Client won't have access to the attributes certificate issued by KGC to perform the update of the attributes on Security Storage.

Concerning the data encryption and decryption operations, they should be implemented in all user sensitive data. However, as an example, they will only be performed over the measurement values, which will be encrypted before WebAL Client sends it to WebAL Server and decrypted before WebAL Client shows it to the user. The symmetric scheme AES-GCM will be used to perform these encryption and decryption operations. So, the server won't be able to compare the measurement values with threshold values. For that, FastORE should be used. It was already integrated with Java using *Java Native Interface* but not yet with the proposed scheme implementation.

The symmetric encryption and decryption operations share a part of the communication flow about getting the intended symmetric key, given its version. The sequence diagram representing this communication flow is presented in Figure 22.

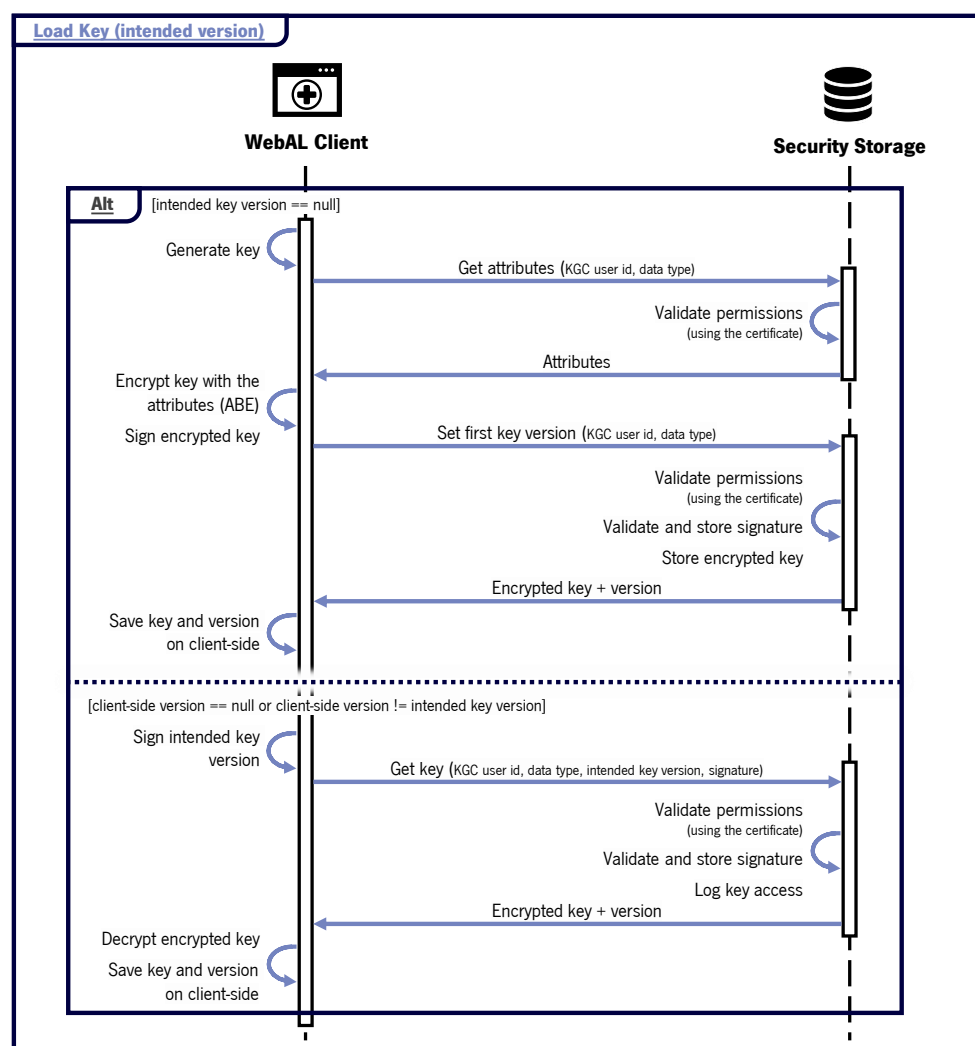


Figure 22: Load key sequence diagram.

## 5.2. Implementation

According to the presented diagram, if no key version is provided, a new key will be generated, encrypted using ABE, and stored on Security Storage. If there is a key version and that key isn't saved on the client-side, the respective encrypted key is obtained from the Security Storage and decrypted using the user ABE key. In both cases, the decrypted key and its version are saved on the client-side (*sessionStorage*) for performance purposes. The encryption and decryption operations will then use the key with the intended version stored on *sessionStorage*. The sequence diagrams representing the data encryption and data decryption communication flows are presented in Figures 23 and 24, respectively.

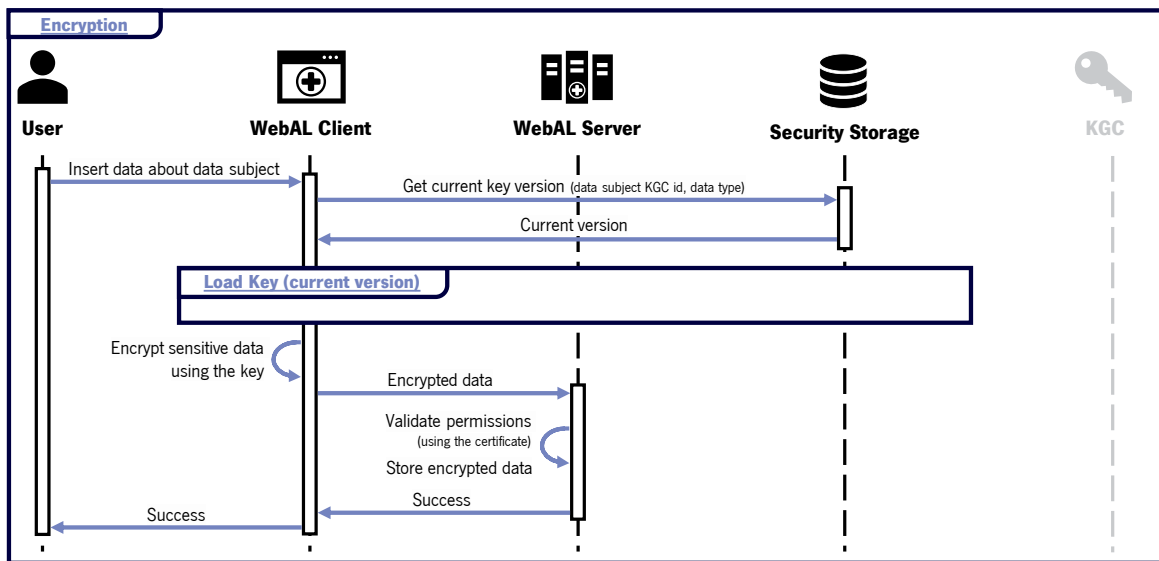


Figure 23: Encryption sequence diagram.

On data encryption, the current key version is retrieved from Security Storage and the respective key loaded as previously described. The sensitive data is then encrypted and stored on the WebAL Server, alongside the encryption key version.

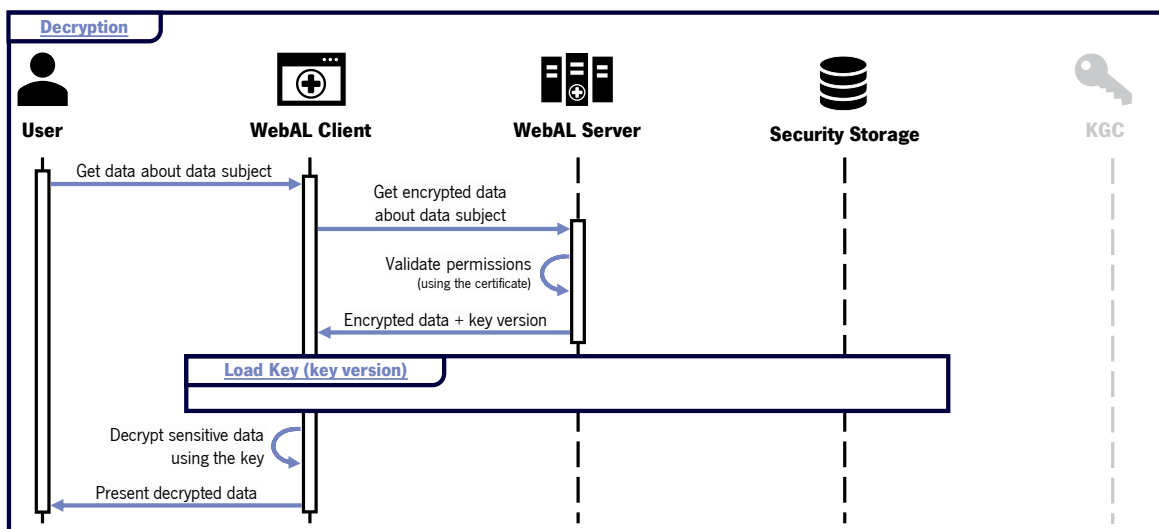


Figure 24: Decryption sequence diagram.

On data decryption, the intended data and its encryption key version are retrieved from the WebAL Server. Then, the key with that version is loaded as previously described, and the sensitive data decrypted with that key.

### 5.3 SUMMARY

This chapter addressed the development of the proposed solution for the scenario Database Encryption, using the browser as UI. The entities involved in this solution are the KGC, which includes the Keycloak and KGC Core servers, the WebAL, and the Security Storage. The technologies and cryptographic algorithms used in each entity are presented in Figure 25.

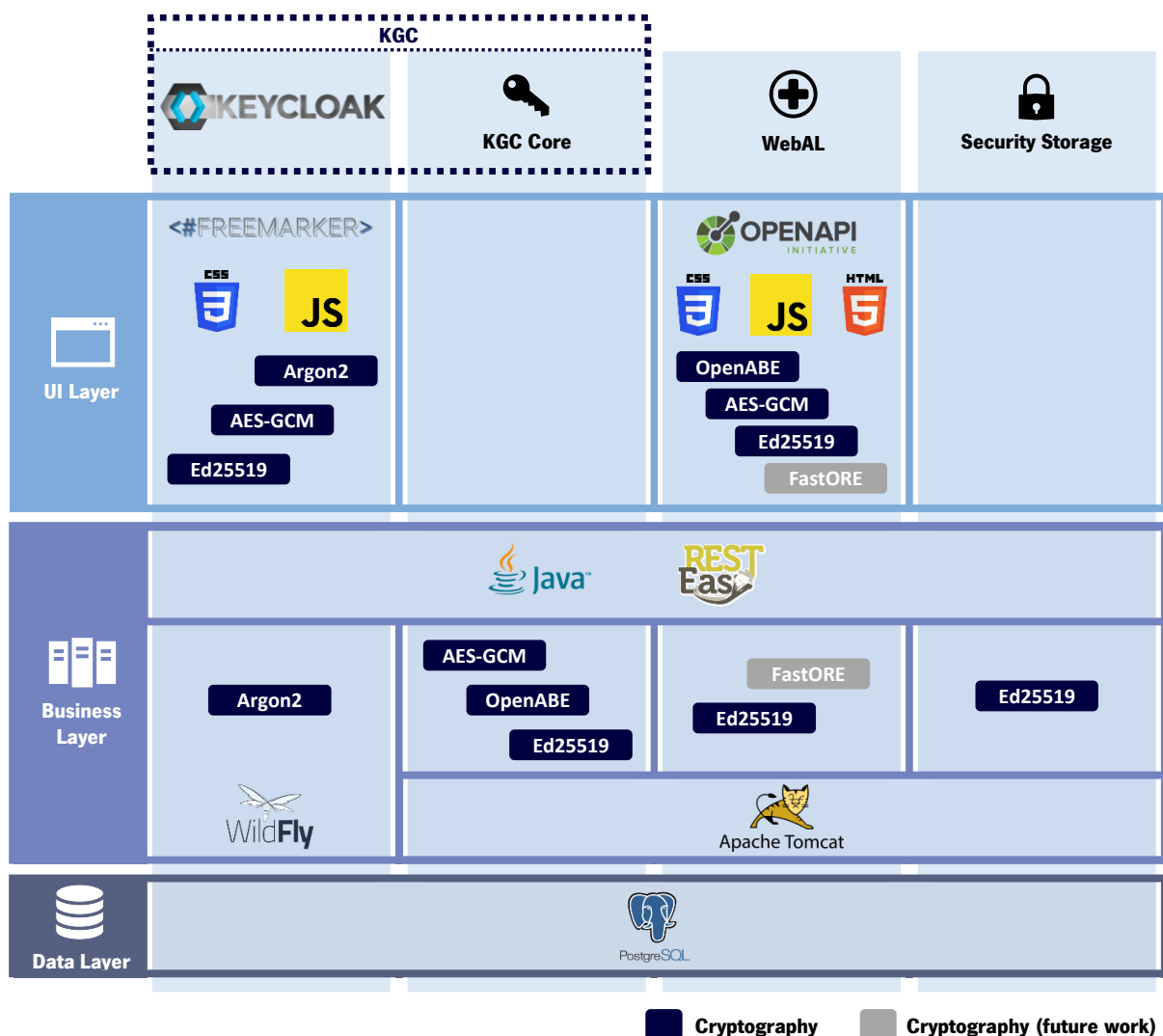


Figure 25: Technologies and cryptographic algorithms used in each entity.

Keycloak is an identity and access management solution used by all entities for user authentication through the OpenId Connect protocol. Keycloak was configured to use a PostgreSQL database for data storage. It is written on Java, uses RESTEasy for its REST API, and runs on a WildFly server by default.

### 5.3. Summary

A custom password policy was added to Keycloak to allow the usage of the hashing algorithm Argon2 for password storage.

Regarding the User Interface, Keycloak uses Freemarker for the HTML generation with java variable values, as well as CSS and JavaScript. The only Keycloak pages presented to the user are the login and reset password pages. The JavaScript (JS) code from those pages was customized to perform some cryptographic operations, such as the password hashing on the client-side with the algorithm Argon2, the encryption and decryption of the user signature private key using AES-GCM, and the generation of the signature key pair on the first login using the algorithm Ed25519.

The KGC Core is used to issue ABE keys and store users' signature key pairs and access structures. WebAL is the assisted living platform, which stores users' personal and medical data, some of which is encrypted. Security Storage saves the keys used to encrypt the WebAL data. Those keys are encrypted with ABE, and the Security Storage also saves the attributes used to encrypt those keys. Only the users with the appropriate ABE key can decrypt those keys, as well as the respective WebAL data.

The KGC Core, WebAL Server, and Security Storage use PostgreSQL for data storage. They are written in Java, use RESTEasy for their REST API, and run on a Tomcat server. The signature algorithm Ed25519 is used by the three entities to issue user's certificates (KGC) or to validate them (WebAL and Security Storage). The KGC Core uses AES-GCM to encrypt and decrypt its private keys, and OpenABE to issue users ABE keys. Although not currently implemented, WebAL would use FastORE to compare measurements with threshold values and alert formal caretakers if a patient has one or more measurements out of a defined range.

Between these three servers, only WebAL has a UI. It uses Swagger UI, which contains HTML, CSS, and JS code, to visually render documentation for an API defined with the OpenAPI Specification. Some JS code was added to perform cryptographic operations, such as keys encryption and decryption using ABE, data encryption and decryption using AES-GCM, and signatures generation with Ed25519. WebAL UI would also use FastORE to encrypt and decrypt measurements.

So, these are the main operations performed by each entity, related to the technologies and cryptographic algorithms used for the Database Encryption solution implementation.

---

## RESULTS AND DISCUSSION

---

In this section, the implemented scheme's practicality will be evaluated regarding its performance and database size. There is no use in performing comparative load tests regarding the proposed scheme since the increase in computational load occurs mainly on the client-side, which is not subject to a lot of simultaneous requests in a real environment. However, if server-side comparisons between encrypted measurements were performed on WebAL using FastORE, this type of analysis would make sense.

Also, because of this client-side processing, the performance analysis must be carried out on the client-side so all processing time is considered.

The obtained results will be compared with a simpler WebAL version, named PlainWebAL, which does not apply the presented scheme and stores all data in plaintext. It uses the same technologies as WebAL except for the cryptographic algorithms. Keycloak is used for authentication, but the relationships between users are stored in the PlainWebAL server. Also, there is no need to store signature or encryption keys since PlainWebAL won't have signed or encrypted data. So, the PlainWebAL server will only need the Keycloak server to be running, but not the Security Storage nor the KGC Core.

The performance evaluation will take into account the response time parameter and will be performed for this scheme's main operations, such as:

- User registration;
- Obtaining user data;
- User relationship registration;
- Patient health measurement creation;
- Obtaining patient measurements;
- ABE encryption;
- ABE decryption.

FastORE performance will also be evaluated but independently from the rest of the project. The obtained response times must be considered acceptable by the end-user, as presented in Table 4 (Nielsen, 2009).

Table 4: Response time limits for different user perceptions.

<b>Response Time Limit</b>	<b>User Perception</b>
0.1 s	System is reacting instantaneously
1 s	Uninterrupted flow of thought, even if the delay is noticed
10 s	User still keeps his attention focused on the ongoing interaction

According to this table, if an operation response time is at most 0.1 seconds, the users feel like their actions are directly (instantaneously) causing something to happen on the screen. This kind of response time is absolutely required in some cases, such as mouse movements or keypresses. Response times of 1 second at most make the users feel like the computer is causing the result to appear. They notice the short delay but stay focused on their current train of thought. If the response time is between 1 and 10 seconds, users get impatient and notice that they're waiting for a slow computer to respond. After about 10 seconds, the average attention span is maxed out. At that point, users will often leave the website.

In short, the main goals of this analysis are to understand whether the obtained response times are considered acceptable by the user, as well as how much the operations' performance has to decrease and the storage occupation increase to improve the security guarantees provided by the assisted living platform using the proposed scheme.

## **6.1 PERFORMANCE EVALUATION**




### **6.1.1 Setup**

With regards to the performance evaluation, a script written in JavaScript was developed so the measurement of the intended operations response time would include the client-side processing. This script measures the current time, accurate to a thousandth of a millisecond, before and after the intended operation, and uses the difference between them as the response time value. The intended operation and respective response time measurement are repeated  $N$  times.

The obtained results for each operation for both WebAL and PlainWebAL are then sent to the WebAL server, which stores them. A Python script uses this data to generate the intended graphics. This script discards the first response time values because of the JavaScript engine warm-up and uses the average of the remaining  $N - 2$  values as the operation response time presented in the graph, so the performance evaluation is more accurate.

This performance evaluation is carried out from both an old smartphone and a regular computer as clients, connected to the server through the same wireless network. This server hosts the Keycloak, KGC Core, Security Storage, WebAL, and PlainWebAL applications or services. The server and clients' specifications are presented in Table 5. The main goal of this setup is to evaluate if the response time values obtained using a smartphone with reduced resources are acceptable. This is important because a considerable part of the processing is carried out on the client-side. The response time values are measured on each device by the mentioned JS script, using  $N = 100$ .

Table 5: Devices used on the results experiment.

Characteristics	Server	Client Computer	Client Phone
Name	ASUS ZenBook UX430UN <sup>1</sup>	Lenovo YOGA 3 Pro <sup>2</sup>	HTC One <sup>3</sup>
Photo			
Operative System	Windows 10 Home	Windows 10 Home	Android 5.0.2
Release Date	2018, February	2014, October	2013, March
CPU	Intel® Core™ i7-8550U 1.8GHz Quad-core	Intel® Core™ M-5Y71 1.2GHz Dual-core	Quad-core 1.7GHz Krait 300
Graphics Card/GPU	NVIDIA GeForce MX150	Intel BDW GT2	Adreno 320
Internal Storage	512 GB SSD	512 GB SSD	32 GB
RAM Memory	16 GB	8 GB	2 GB

Taking into account that FastORE was not integrated with the rest of the project, its performance is evaluated on the server-side, in a standalone Java application. *Java Microbenchmark Harness (JMH)*

<sup>1</sup> <https://www.asus.com/Laptops/ASUS-ZenBook-UX430UN/>

<sup>2</sup> <https://support.lenovo.com/us/en/solutions/pd100359-product-overview-lenovo-yoga-3-pro-1370>

<sup>3</sup> [https://www.gsmarena.com/htc\\_one-5313.php](https://www.gsmarena.com/htc_one-5313.php)

## 6.1. Performance Evaluation

is used to build and run java benchmarks over the FastORE encryption, decryption, and comparison operations.

### 6.1.2 Evaluation

This section presents and analyzes the results of the performance evaluation. The response times obtained for the mentioned operations performed by the assisted living platform on both the computer and the phone described previously, for WebAL and PlainWebAL, are presented in Figure 26.

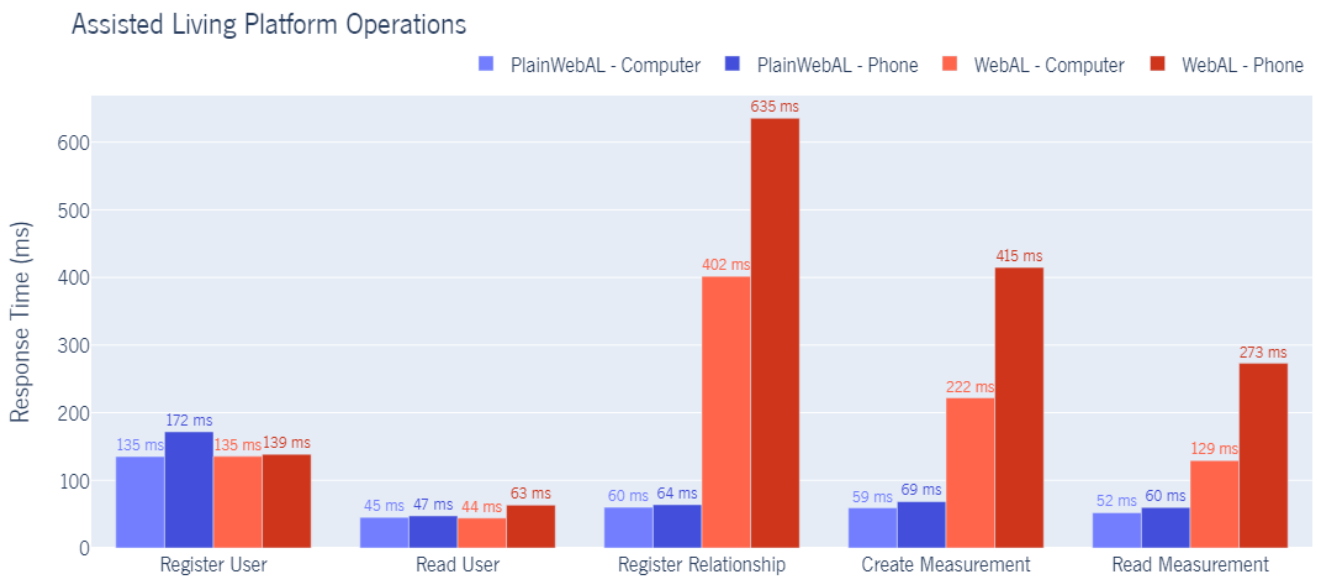


Figure 26: Response time of the assisted living platform operations.

Both the user registration and read operations perform similarly between WebAL and PlainWebAL, as well as between the client computer and smartphone. Regarding the slightly higher response time of the PlainWebAL in the smartphone, it may be attributed to variations in the network activity. The user registration operation includes additional requests in the case of WebAL, whose server communicates with the KGC Core and Security Storage servers. However, this communication occurs between servers that run on the same machine in this test scenario, so the related latency is reduced. This data is also stored in plaintext on WebAL, so there is no performance decrease on those operations because of cryptographic algorithms.

The WebAL relationship registration operation, on the other hand, involves signing data on the client-side and issuing a certificate by the KGC Core, as well as validating both. This and the need to store data on two servers (KGC and Security Storage) instead of just one (PlainWebAL) causes a significant decrease in WebAL's performance compared to PlainWebAL. On the smartphone, the WebAL response time increases considerably, which indicates that the signing algorithm performance is meaningfully worse on devices with



fewer resources. However, this specific operation is only executed by administrative entities in a work environment, usually on a computer, and it is not expected to be performed very often.

The measurement creation operation on WebAL involves multiple communications with Security Storage to obtain the needed encryption keys. Keys that do not yet exist are created, encrypted, and stored in Security Storage. Keys that already exist in Security Storage are obtained and decrypted. Keys that already exist and were previously loaded are obtained directly from *sessionStorage*. In addition to this process, there is also the encryption of one patient's health measurement per request. All this additional communication and client-side processing make the performance of WebAL considerably worse than PlainWebAL, which only stores the plaintext data about the new measurement. On the smartphone, the difference is even more striking due to the worse performance of cryptographic algorithms on mobile phone browsers.

The WebAL measurement read operation includes obtaining the patient's encrypted measurements and associated key version, as well as communicating with the Key Storage to obtain that encrypted key. This key is then decrypted with ABE and the result is used as a key to decipher the patient's measurement. Once again, this additional communication and client-side processing make the performance of WebAL considerably worse than PlainWebAL, especially on the smartphone.

The performance of the ABE encryption and decryption operations on the client-side was analyzed independently from the proposed scheme. A comparative analysis between a Java version running on Docker and a WebAssembly (Wasm) version generated by Emscripten was carried out, being the results presented in Figure 27.

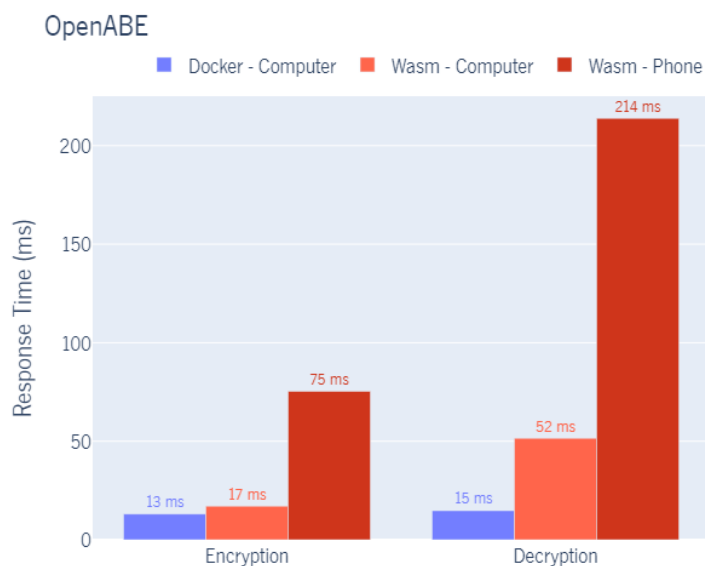


Figure 27: Performance evaluation of ABE encryption and decryption operations.

## 6.1. Performance Evaluation

The Docker version was only evaluated on the computer client since the docker engine is not currently supported on any version of Android. For testing this approach with a smartphone, an Android application providing ABE operations would have to be developed. The Docker version presents better performance than the WebAssembly version, being this difference more significant regarding the decryption algorithm, particularly when using the smartphone as a client.

Regarding the usage of ORE to encrypt measurements and compare their encrypted values with other encrypted values, an independent performance evaluation was carried out. This evaluation takes place on the server computer, and the performance results of using no encrypted data or FastORE are presented in Figure 28.

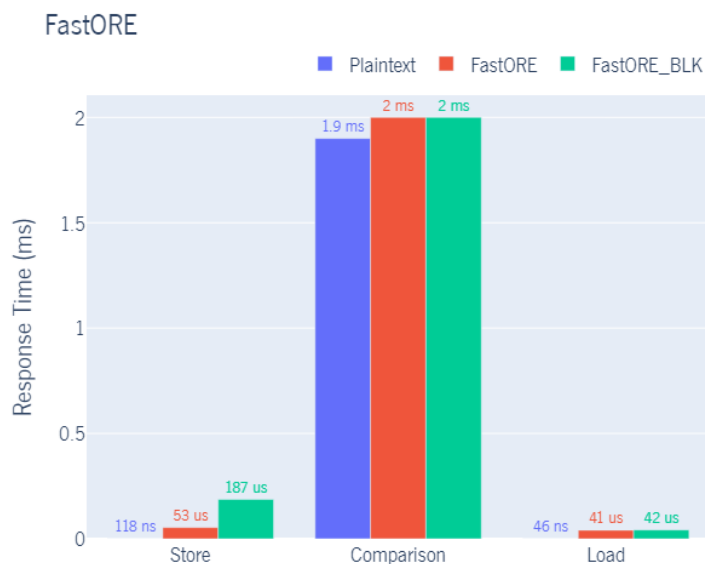


Figure 28: Performance evaluation of FastORE operations.

The store and load operations include the return of a plaintext value for the plaintext version, or the encryption and decryption of that value and return of the result, respectively, for the FastORE versions. The comparison operation includes the fetch of the needed threshold encrypted values from the database and the comparison between those and a specific measurement encrypted value. The obtained response times demonstrate that applying FastORE on encryption, comparison, or decryption of health measurements will not substantially impair the performance of the assisted living platform.

All these results allow concluding that, when the proposed scheme is applied, there is a considerable increase in the response time of operations that involve cryptography. However, even the operation with the worst performance has an average response time of less than one second, which is considered acceptable taking into account the data in Table 4.

Nonetheless, it should be noted that in this WebAL prototype, only one health measurement is encrypted per request. Therefore, the real implementation would have higher response times. However, the symmetric encryption scheme performs better than the ABE scheme, so it would be possible to encrypt multiple values with the same symmetric key without the respective operation response time exceeding one second. The same reasoning applies to operations such as creating users, whose involved data is not encrypted but on the real implementation should be, providing better guarantees of data privacy and security than this prototype.

Therefore, the obtained results are considered positive. Although the need to optimize cryptographic algorithms for devices with fewer resources stands out, it is already possible to apply the current algorithms to practical use cases, with acceptable response times for the user. In this scenario that deals with such sensitive data, the improvement of data security and privacy guarantees definitely compensates for the performance loss.

## **6.2 DATABASES SIZE**

### **6.2.1 Setup**

Regarding the database population, it is executed by the performance evaluation JS script. When the WebAL server receives the response time values measured by this script, it performs an SQL query to get each database's size and stores it together with the received response times.

Each database size is measured after the script runs with  $N = 5000$ , so the difference between PlainWebAL and WebAL (with KGC and Security Storage) databases is emphasized. This means that the databases will have 5000 users, 5000 relationships, and 5000 medical measurements. The script varies the inserted relationships, as well as each medical measurement creator and patient. So, the number of keys used to encrypt the WebAL data may vary, similarly to the real functioning of the assisted living platform.

### **6.2.2 Evaluation**

Data encryption is naturally associated with an increase in the occupied storage space. The proposed scheme triples the number of required entities (and databases) compared to the basic version of the assisted living platform. The space occupied by the databases of both versions, WebAL and PlainWebAL, is indicated in the graph shown in Figure 29.

## 6.2. Databases Size

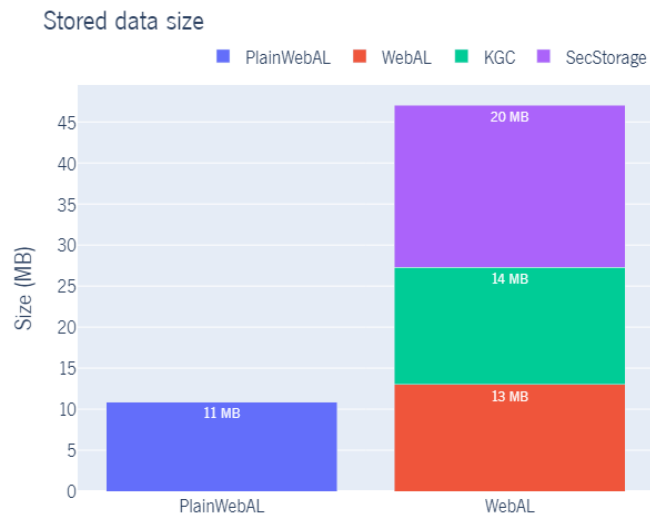


Figure 29: Size of the assisted living platform databases.

The set of databases related to the scheme that includes WebAL is approximately four times the PlainWebAL database's size. Although some of this difference in sizes is due to the existence of two more databases per se, the remainder is because of the additional stored information, such as ABE attributes, access structures, and keys, as well as the fact that the ciphertexts are larger than the respective plaintexts (an initialization vector and authentication tag must be stored for each ciphertext).

Regarding the occupied storage increase due to ABE and ORE usage, the comparison between some plaintext and its ciphertext size was carried out. So this analysis would match the real use case of these cryptographic schemes, the plaintext used for ABE was a symmetric key, while the one used for ORE was a health measurement like value. The sizes of the plaintexts and the obtained cryptograms, measured in characters, are presented in Figures 30 and 31.

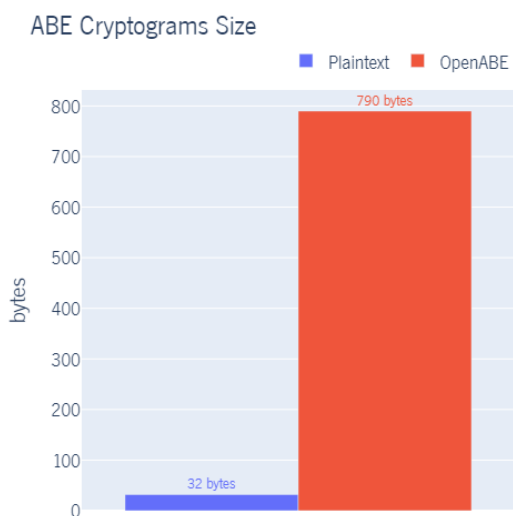


Figure 30: ABE ciphertexts' size.

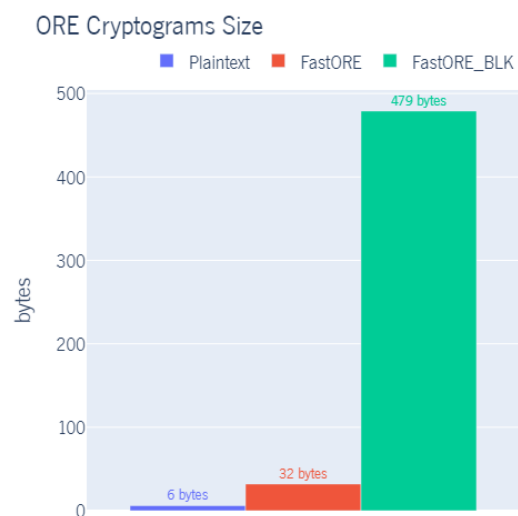


Figure 31: FastORE ciphertexts' size.

The results show a significant increase in the occupied space due to encryption, especially using the OpenABE and FastORE\_BLK schemes.

Whether the general increase in the occupied space is considered acceptable will have to be evaluated by the company that manages the platform. Considering the nature of the data managed by the assisted living platform, it should be justified to invest more in storage resources to offer users better privacy and security guarantees.

### **6.3 SUMMARY**

To evaluate the proposed scheme's practicality, the performance and database(s) size of two assisted living platform versions, PlainWebAL and WebAL, were considered. Both these versions provide identical functionality and the same kind of information to the user. However, while PlainWebAL deals only with plaintexts, WebAL improves privacy and security guarantees by applying the proposed scheme.

On WebAL, users' sensitive data is encrypted and decrypted on the client-side, signatures are validated on the server-side, and both of them sign messages (mostly the client-side). This means that the performance measurement must take into account both the client-side and server-side processing.

For the performance evaluation, the operations response time was measured for each version, both on the computer and smartphone clients. The main goal was to ensure that this scheme performance is considered acceptable for the user, even in a device with reduced resources, such as a 2013 smartphone. Despite the considerable increase in the response time of operations that involve cryptography, the main goal was achieved: all the operations had an average response time of less than one second. This performance loss is more evident when the smartphone client is used, which highlights the need to optimize cryptographic algorithms for devices with fewer resources.

Regarding the database(s) size, the usage of the proposed scheme implies a considerable increase in the occupied storage space, mainly due to the existence of more databases and stored data, such as ABE attributes, access structures, and keys.

Whether the increase in response time and occupied storage space are considered acceptable will have to be evaluated by the company that manages the platform. However, it is already possible to apply the current PETs to practical use cases, with acceptable response times for the user. In this scenario that deals with such sensitive data, the improvement of data security and privacy guarantees seems to compensate for the performance loss and the need for more storage resources.

---

## CONCLUSION

---

### 7.1 CONCLUSIONS

In the digital world, most industry revenue streams are based on the collection of massive amounts of personal data. It became a valuable asset with many applications and can offer advantages for both users and service providers. Users can benefit from sharing their data as this might unlock certain additional features of a service or contribute to a more personalized experience. Service providers can use personal data for analysis and targeted advertisement. Also, personal data can be shared or traded with other third parties. Finally, research can greatly benefit from databases filled with personal data. This is especially important in the health care sector, as a huge sample size leads to more precise results.

The current pandemic of COVID-19 has shown several usages of personal data (Becher et al., 2020), such as tracing chains of infection and analyzing medical records of treated COVID-19 patients, that might help to improve the treatment of future cases and could potentially speed up the process of producing a vaccine. Therefore, every data record is important, and the best possible results could be obtained if all nations would work together on a shared database.

However, the usage of personal data has several disadvantages from a data protection perspective. Combined with an increase in the frequency and prevalence of cybercrime, more of the public now faces the very real risk of privacy loss associated with the illegitimate use of private data. Privacy is a fundamental human right recognized in the United Nations (UN) Declaration of Human Rights, the International Covenant on Civil and Political Rights, and in many other international and regional treaties (Mendel et al., 2012). Data privacy laws such as the GDPR are designed to give users more control over their data while demanding businesses more protection for their users' personal data.

PETs are a means of implementing data protection by design on a technical level, thus helping to achieve GDPR compliance. They embody fundamental data protection principles by minimizing personal

data use, maximizing data security, and empowering individuals. However, PETs can be used for more than personal privacy, once data might be commercially sensitive or related to national security, for example.

Businesswise, the most obvious reason to invest in PETs would be to avoid the heavy fines imposed by GDPR, but that would be a narrow view of the problem. Recent data leaks have increased public perception of security. Through no fault of their own, their personal information can become available on the internet, which, at best, can be obnoxious and, at worst, life-ruining. There is a demand for accountability, and the consequences of ignoring fundamental security guidelines are clear. Privacy is no longer opt-in, it's a requirement.

As state-of-the-art research has shown, there is no shortage of good ideas for protecting individual privacy. A wide range of PETs have been proposed, some of which are addressed in this dissertation, but not all of them have made their way out of the research environment and into the marketplace or people's lives in any meaningful way.

There are some possible barriers to the implementation or adoption of PETs including, but not limited to, lack of awareness of the existence of these tools, their lack of usability, and a lack of incentive for organizations to offer or implement these tools. There are also some categories of PETs, such as data tracking, that do not seem to have attracted the same degree of research interest as others. It should also be noted that most existing PETs encryption schemes are based on conventional cryptographic hardness assumptions, which, in the future, will be vulnerable to adversaries equipped with quantum computers.

However, there are already some PETs that can be applied in practice to certain use cases. All PETs have their advantages and disadvantages, and when choosing which ones to use, we need to take into account the specific problem being addressed and strike a balance between security and usability.

In this work, a PETs based scheme was proposed to address the encryption of the data used by an assisted living platform. This way, sensitive data would only be successfully decrypted by authorized users, and not even the server would have access to that data in plaintext. However, the server would be able to perform some operations over the encrypted sensitive data. A prototype of this platform using the proposed scheme was implemented, and its cost in terms of usability, more specifically performance, space occupation, and key management, was considered acceptable. Given the very sensitive nature of the handled information, the increase in privacy and security, while maintaining the functionality, far outweighs the costs of using the proposed scheme.

So, despite the need for more research, some PETs already can (and should) be used to improve data privacy and security guarantees. Information is valuable, and users are starting to think twice before trusting a company with their data. This concern can be mitigated by the adoption of PETs, which in

turn will increase the company's credibility. By guaranteeing privacy-by-design products and services, a company's offer will certainly stand out from the crowd.

## **7.2 PROSPECT FOR FUTURE WORK**

This work opens several doors for further research or development. Regarding the PETs application scenarios, for example, it would be possible to identify and explore some more scenarios in the scope of the company Altice Labs, such as secure communication between IoT devices. It would also be interesting to develop prototypes related to more application scenarios and evaluate their performance, to understand whether the proposed solution could be applied in practice.

Regarding the proposed and implemented scheme, which encrypts the assisted living platform's data, further development of the prototype would be needed. Better alternatives could be investigated to implement certain features, such as the permissions validation using certificates, the size of which increases proportionally to the number of users related to the certificate holder. Nonetheless, the server operations addressed in section 4.4 could be implemented using the proposed approaches. Also, when more than one implementation is available, a comparative analysis of performance and storage occupation could be carried out between those PET implementations.

More generally, there is still the need to improve the effectiveness of the existing PETs and develop new ones, as well as improve cryptographic algorithms' performance on devices with fewer resources. Despite that, there are already some PETs that can be applied in practice.

There is a growing global recognition of the data value and the importance of prioritizing data privacy and security as critical cornerstones of business operations. This highlights the need to continue the research and integration of PETs in companies' products and services, as a way of guaranteeing data privacy and security even during data processing.



---

**BIBLIOGRAPHY**

---

- Acar, A., Aksu, H., Uluagac, A. S., and Conti, M. (2018). A survey on homomorphic encryption schemes: Theory and implementation. *ACM Comput. Surv.*, 51(4).
- Agrawal, S. and Chase, M. (2017). *FAME: Fast Attribute-based Message Encryption*. Accessed on 14-10-2020, available at <https://eprint.iacr.org/2017/807.pdf>.
- Ahn, H., Schuff, D., Zakai, A., Chen, G., and Lively, T. (2020). *Emscripten*. Accessed on 14-10-2020, available at <https://github.com/emscripten-core/emscripten>.
- Al-Maytami, B. A., Fan, P., Hussain, A. J., Baker, T., and Liatsis, P. (2020). An efficient queries processing model based on multi broadcast searchable keywords encryption (mbske). *Ad Hoc Networks*, 98:102028.
- Alabdulatif, A., Khalil, I., and Yi, X. (2020). Towards secure big data analytic for cloud-enabled applications with fully homomorphic encryption. *Journal of Parallel and Distributed Computing*, 137:192 – 204.
- Alloghani, M., M. Alani, M., Al-Jumeily, D., Baker, T., Mustafina, J., Hussain, A., and J. Aljaaf, A. (2019). A systematic review on the status and progress of homomorphic encryption technologies. *Journal of Information Security and Applications*, 48:102362.
- Altice (n.d.). *Privacy Policy*. Accessed on 13/09/2019, available at <https://www.telecom.pt/en-us/pages/politica-privacidade.aspx>.
- Anada, H. and Arita, S. (2017). Short cca-secure ciphertext-policy attribute-based encryption. In *2017 IEEE International Conference on Smart Computing – SMARTCOMP*, pages 1–6.
- Anonymous (2017). *Campanha contra o Nónio*. Accessed on 20/09/2019, available at <https://nonio.pt/>.
- Antipolis, S. (2018). *ETSI Releases Cryptographic Standards for Secure Access Control*. Accessed on 19/12/2020, available at <https://www.etsi.org/newsroom/press-releases/1328-2018-08-press-etsi-releases-cryptographic-standards-for-secure-access-control>.

## BIBLIOGRAPHY

- Archer, D. W., Calderón Trilla, J. M., Dagit, J., Malozemoff, A., Polyakov, Y., Rohloff, K., and Ryan, G. (2019). Ramparts: A programmer-friendly system for building homomorphic encryption applications. page 57–68.
- Arcieri, T. (2013). *What's wrong with in-browser cryptography?* Available at <https://tonyarcieri.com/whats-wrong-with-webcrypto>.
- Arias, D. (2019). *Hashing Passwords: One-Way Road to Security*. Accessed on 12/10/2020, available at <https://auth0.com/blog/hashing-passwords-one-way-road-to-security/>.
- Ariel, A., Bakker, B., Groot, M., and Grootheest, G. (2014). *Record Linkage in Health Data: a simulation study*. Accessed on 08/10/2019, available at <https://www.biolink-nl.eu/public/2014%20Record%20linkage%20simulation.pdf>.
- Attrapadung, N., Hanaoka, G., Matsumoto, T., Teruya, T., and Yamada, S. (2016). Attribute based encryption with direct efficiency tradeoff. In Manulis, M., Sadeghi, A.-R., and Schneider, S., editors, *Applied Cryptography and Network Security*, pages 249–266, Cham. Springer International Publishing.
- Auth0 (n.d.). *SAML vs. OpenID (OIDC)*. Accessed on 15/06/2020, available at <https://auth0.com/intro-to-iam/saml-vs-openid-connect-oidc/>.
- Bayatbabolghani, F. and Blanton, M. (2018). Secure multi-party computation. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18*, page 2157–2159, New York, NY, USA. Association for Computing Machinery.
- Becher, S., Gerl, A., Meier, B., and Bözl, F. (2020). *Big Picture on Privacy Enhancing Technologies in e-Health: A Holistic Personal Privacy Workflow*. Accessed on 01-11-2020, available at <https://www.mdpi.com/2078-2489/11/7/356/htm>.
- Biryukov, A., Dinu, D., Khovratovich, D., and Josefsson, S. (2020). *The memory-hard Argon2 password hash and proof-of-work function*. Accessed on 12/10/2020, available at [https://datatracker.ietf.org/doc/draft-irtf-cfrg-argon2/?include\\_text=1](https://datatracker.ietf.org/doc/draft-irtf-cfrg-argon2/?include_text=1).
- Bost, R. (2018). *Algorithmes de recherche sur bases de données chiffrées*. PhD thesis. Thèse de doctorat dirigée par Fouque, Pierre-Alain et Pointcheval, David Informatique Rennes 1 2018.
- Brakerski, Z. (2018). Quantum fhe (almost) as secure as classical. pages 67–95.

- Cavoukian, A. (2009). *Privacy by Design: The 7 Foundational Principles*. Accessed on 15/11/2020, available at <https://www.ipc.on.ca/wp-content/uploads/Resources/pbd-privacy-and-security-by-design-oracle.pdf>.
- Chakarov, D. and Papazov, Y. (2019). Evaluation of the complexity of fully homomorphic encryption schemes in implementations of programs. page 62–67.
- Chen, H., Dai, W., Kim, M., and Song, Y. (2019). Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference. page 395–412.
- Chenette, N., Lewi, K., Weis, S. A., and Wu, D. J. (2016). Practical order-revealing encryption with limited leakage. pages 474–493.
- Chung, S., Shieh, M., and Chiueh, T. (2019). Fetch: A cloud-native searchable encryption scheme enabling efficient pattern search on encrypted data within cloud services.
- Cigniti Technologies (2015). *Securing on-premise data through data masking*. Accessed on 01/10/2019, available at <https://www.cigniti.com/blog/securing-on-premise-data-through-data-masking-2/>.
- Cohen, R., Garay, J., and Zikas, V. (2020). Broadcast-optimal two-round mpc. In Canteaut, A. and Ishai, Y., editors, *Advances in Cryptology – EUROCRYPT 2020*, pages 828–858, Cham. Springer International Publishing.
- Curtis, B. R. and Player, R. (2019). On the feasibility and impact of standardising sparse-secret lwe parameter sets for homomorphic encryption. page 1–10.
- Danezis, G., Domingo-Ferrer, J., Hansen, M., Hoepman, J.-H., Métayer, D. L., Tirtea, R., and Schiffner, S. (2014). *Privacy and Data Protection by Design – from policy to engineering*.
- Data61, C. (2017). Anonlink private record linkage system. <https://github.com/data61/anonlink-entity-service>.
- de Muijnck-Hughes, J. (2016). *Predicate Based Encryption Library*. Accessed on 15-10-2020, available at <https://github.com/jfdm/pyPEBEL>.
- Dennedy, M. F., Fox, J., and Finneran, T. R. (2014). *The Privacy Engineer’s Manifesto: Getting from Policy to Code to QA to Value*. Apress, Berkeley, CA. Available at <https://doi.org/10.1007/978-1-4302-6356-2>.

## BIBLIOGRAPHY

- developers, O. (2020). *Java Microbenchmark Harness (JMH)*. Accessed on 22-10-2020, available at <https://github.com/openjdk/jmh>.
- Digital Bazaar, I. (2020). *Forge*. Accessed on 12-10-2020, available at <https://github.com/digitalbazaar/forge>.
- Dong, X., Zhang, Y., Wang, B., and Chen, J. (2020). Server-aided revocable attribute-based encryption from lattices. *Security and Communication Networks*, 2020(1460531).
- Drozdzowski, P., Buchmann, N., Rathgeb, C., Margraf, M., and Busch, C. (2019). On the application of homomorphic encryption to face identification. In *2019 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–5.
- Dwork, C. and Kohli, N. (2019). *Differential privacy in practice*. Accessed on 04/10/2019, available at <https://simons.berkeley.edu/events/differential-privacy-practice>.
- El-Yahyaoui, A. and Kettani, M. D. E.-C. E. (2019). A verifiable fully homomorphic encryption scheme for cloud computing security. *Technologies*, 7(1).
- Elizabeth, B. L. and Prakash, A. J. (2019). Verifiable top-k searchable encryption for cloud data. *Siddhan*, 45.
- Elmahdi, E., Yoo, S.-M., and Sharshembiev, K. (2020). Secure and reliable data forwarding using homomorphic encryption against blackhole attacks in mobile ad hoc networks. *Journal of Information Security and Applications*, 51:102425.
- Engelke, C. (2014). *Saving Cryptographic Keys in the Browser*. Accessed on 15-10-2020, available at <https://blog.engelke.com/2014/09/19/saving-cryptographic-keys-in-the-browser/>.
- ENISA – European Union Agency for Cybersecurity (2014). *Privacy and Data Protection by Design – from policy to engineering*. Available at [https://www.enisa.europa.eu/publications/privacy-and-data-protection-by-design/at\\_download/fullReport](https://www.enisa.europa.eu/publications/privacy-and-data-protection-by-design/at_download/fullReport).
- ENISA – European Union Agency for Cybersecurity (2015). *Privacy by design in big data: An overview of privacy enhancing technologies in the era of big data analytics*. Available at [https://www.enisa.europa.eu/publications/big-data-protection/at\\_download/fullReport](https://www.enisa.europa.eu/publications/big-data-protection/at_download/fullReport).

- ENISA – European Union Agency for Cybersecurity (2016a). *Big Data Threat Landscape and Good Practice Guide*. Accessed on 17/10/2019, available at <https://www.enisa.europa.eu/publications/bigdata-threat-landscape>.
- ENISA – European Union Agency for Cybersecurity (2016b). *Privacy enhancing technologies*. Accessed on 27/09/2019, available at <https://www.enisa.europa.eu/topics/data-protection/privacy-enhancing-technologies>.
- ENISA – European Union Agency for Cybersecurity (2018). *ENISA's PETs Maturity Assessment Repository*. Available at [https://www.enisa.europa.eu/publications/enisa2019s-pets-maturity-assessment-repository/at\\_download/fullReport](https://www.enisa.europa.eu/publications/enisa2019s-pets-maturity-assessment-repository/at_download/fullReport).
- EU GDPR Portal (2018). *GDPR – Encryption*. Accessed on 21/02/2020, available at <https://gdpr-info.eu/issues/encryption/>.
- EU GDPR Portal (n.d.). *The EU General Data Protection Regulation (GDPR) is the most important change in data privacy regulation in 20 years*. Accessed on 14/10/2019, available at <https://eugdpr.org/>.
- European Commission (2014). *Opinion 05/2014 on Anonymisation Techniques*. Accessed on 04/10/2019, available at [https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216\\_en.pdf](https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf).
- Evans, D., Kolesnikov, V., and Rosulek, M. (2018). *A Pragmatic Introduction to Secure Multi-Party Computation*. NOW Publishers.
- Feng, T. and He, W. (2018). Research on privacy preserving of searchable encryption. In *Proceedings of the 2018 2nd High Performance Computing and Cluster Technologies Conference, HPCCT 2018*, page 58–68, New York, NY, USA. Association for Computing Machinery.
- Franke, M., Gladbach, M., Sehili, Z., Rohde, F., and Rahm, E. (2019). *ScaDS Research on Scalable Privacy-preserving Record Linkage*.
- Fraunhofer AISEC (Applied and Integrated Security) (2020). *Rabe*. Accessed on 14-10-2020, available at <https://github.com/Fraunhofer-AISEC/rabe>.
- Friedman, A., Wolff, R., and Schuster, A. (2007). *Providing k-anonymity in data mining*.

## BIBLIOGRAPHY

- GDPR Report (2017). *Data masking: anonymization or pseudonymization?* Accessed on 30/09/2019, available at <https://gdpr.report/news/2017/09/28/data-masking-anonymization-pseudonymization/>.
- Google (2018). *Privacy & Terms*. Accessed on 06/09/2019, available at <https://policies.google.com/?hl=en>.
- Goyal, V., Pandey, O., Sahai, A., and Waters, B. (2006). *Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data*. Accessed on 14-10-2020, available at <https://eprint.iacr.org/2006/309.pdf>.
- Hoang, V., Lehtihet, E., and Ghamri-Doudane, Y. (2019). Forward-secure data outsourcing based on revocable attribute-based encryption. pages 1839–1846.
- Homomorphic Encryption Standardization (2017). *Homomorphic Encryption – Introduction*. Available at <https://homomorphicencryption.org/introduction/>.
- Homomorphic Encryption Standardization (2019). *Homomorphic Encryption*. Available at <https://homomorphicencryption.org/>.
- Hustinx, P. (2010). *Privacy by design: delivering the promises*. *Identity in the Information Society*, 3:253–255.
- Information Commissioner’s Officer (2018). *What is encryption?* Accessed on 11/10/2019, available at <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/encryption/what-is-encryption/>.
- intersoft consulting (2018). *General Data Protection Regulation*. Accessed on 14/10/2019, available at <https://gdpr-info.eu/>.
- Islam, M., Kuzu, M., and Kantarcioglu, M. (2012). *Access Pattern disclosure on Searchable Encryption: Ramification, Attack and Mitigation*. Accessed on 14/01/2021, available at <https://www.ndss-symposium.org/ndss2012/ndss-2012-programme/access-pattern-disclosure-searchable-encryption-ramification-attack-and-mitigation/>.
- Jiang, Z. L., Guo, N., Jin, Y., Lv, J., Wu, Y., Liu, Z., Fang, J., Yiu, S., and Wang, X. (2020). Efficient two-party privacy-preserving collaborative k-means clustering protocol supporting both storage and computation outsourcing. *Information Sciences*, 518:168 – 180.

- Kaiser, P., Langer, A., and Gaedke, M. (2019). Mppc: Generic secure multi-party computation in centralized cloud-based environments. In *Proceedings of the 2019 3rd International Conference on Big Data Research, ICBDR 2019*, page 60–66, New York, NY, USA. Association for Computing Machinery.
- Kalchev, I. (2018). *A lightweight Attribute-Based Encryption Scheme in C++*. Accessed on 14-10-2020, available at <https://github.com/ikalchev/kpabe-yct14-cpp>.
- Kalyani, G. and Chaudhari, S. (2019). Data privacy preservation in mac aware internet of things with optimized key generation. *Journal of King Saud University - Computer and Information Sciences*.
- Kamal, A. A. A. M. and Iwamura, K. (2019). Searchable encryption using secret-sharing scheme for multiple keyword search using conjunctive and disjunctive searching. In *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBD-Com/CyberSciTech)*, pages 149–156.
- Kamentsky, L. (2020). *The javabridge Python package*. Accessed on 15-10-2020, available at <https://github.com/LeeKamentsky/python-javabridge>.
- Kammerer, M. (2020). *Argon2 Binding for the JVM*. Accessed on 12-10-2020, available at <https://github.com/phxql/argon2-jvm>.
- Karaki, J. (2019). *Identity Management: SAML vs. OAuth2 vs. OpenID Connect*. Accessed on 15/06/2020, available at <https://medium.com/@jad.karaki/c9a06548b4c5>.
- Khader, D. (2014). Introduction to attribute based searchable encryption. In De Decker, B. and Zúquete, A., editors, *Communications and Multimedia Security*, pages 131–135, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Kumar, D. V. N. S. and Thilagam, P. S. (2019). Searchable encryption approaches: attacks and challenges. *Knowledge and Information Systems*, 61.
- Lai, J., Deng, R. H., Pang, H., and Weng, J. (2014). Verifiable computation on outsourced encrypted data. In Kutylowski, M. and Vaidya, J., editors, *Computer Security - ESORICS 2014*, pages 273–291, Cham. Springer International Publishing.
- Lewi, K. and Wu, D. J. (2016). Order-revealing encryption: New constructions, applications, and lower bounds. page 1167–1178.

## BIBLIOGRAPHY

- Lewko, A., Sahai, A., and Waters, B. (2010). *Revocation Systems with Very Small Private Keys*. pages 273–285.
- Li, D., Liao, X., Xiang, T., Wu, J., and Le, J. (2019a). *Privacy-preserving self-serviced medical diagnosis scheme based on secure multi-party computation*. Accessed on 04/02/2020, available at <https://doi.org/10.1016/j.cose.2019.101701>.
- Li, J., Huang, Y., Wei, Y., Lv, S., Liu, Z., Dong, C., and Lou, W. (2019b). Searchable symmetric encryption with forward search privacy. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1.
- Liao, T.-L., Lin, H.-R., Wan, P.-Y., and Yan, J.-J. (2019). Improved attribute-based encryption using chaos synchronization and its application to mqtt security. *Applied Sciences*, 9(4454).
- Liu, Z., Jiang, Z. L., Wang, X., and Yiu, S. (2018). Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating. *Journal of Network and Computer Applications*, 108:112–123.
- Lomas, N. (2018). *WTF is dark pattern design?* Accessed on 20/09/2019, available at <https://techcrunch.com/2018/07/01/wtf-is-dark-pattern-design/>.
- Lu, Y. and Li, J. (2019). Constructing certificateless encryption with keyword search against outside and inside keyword guessing attacks. *China Communications*, 16(7):156–173.
- Lu, Y., Li, J., and Zhang, Y. (2019). Scf-pepcks: Secure channel free public key encryption with privacy-conserving keyword search. *IEEE Access*, 7:40878–40892.
- López-Alt, A., Tromer, E., and Vaikuntanathan, V. (2017). Multikey fully homomorphic encryption and applications. *SIAM Journal on Computing*, 46(6):1827–1892.
- Ma, M., He, D., Fan, S., and Feng, D. (2020). Certificateless searchable public key encryption scheme secure against keyword guessing attacks for smart healthcare. *Journal of Information Security and Applications*, 50:102429.
- Ma, X., Liu, C., Cao, S., and Zhu, B. B. (2018). Jpeg decompression in the homomorphic encryption domain. page 905–913.
- Maddox, I. and Moschetto, K. (2019). *Modern password security for system designers*. Accessed on 12/10/2020, available at [https://datatracker.ietf.org/doc/draft-irtf-cfrg-argon2/?include\\_text=1](https://datatracker.ietf.org/doc/draft-irtf-cfrg-argon2/?include_text=1).



- Mainardi, N., Barengi, A., and Pelosi, G. (2019). Plaintext recovery attacks against linearly decryptable fully homomorphic encryption schemes. *Computers & Security*, 87:101587.
- Malvoni, K., Designer, S., and Knezovic, J. (2014). *Are Your Passwords Safe: Energy-Efficient Bcrypt Cracking with Low-Cost Parallel Hardware*. Accessed on 12/10/2020, available at <https://www.usenix.org/system/files/conference/woot14/woot14-malvoni.pdf>.
- Martins, C. (2019). *Site do Público esconde deliberadamente opção para recusar todos os cookies*. Accessed on 20/09/2019, available at <https://abertoatedemadrugada.com/2019/08/publico-esconde-recusar-cookies.html>.
- McCarthy, N. and Fourniol, F. (2020). The role of technology in governance: The example of privacy enhancing technologies. *Data & Policy*, 2:e8.
- Mendel, T., Puddephatt, A., Wagner, B., Hawtin, D., and Torres, N. (2012). *Global survey on internet privacy and freedom of expression*. United Nations Educational, Scientific and Cultural Organization - UNESCO.
- Mert, A. C., Öztürk, E., and Savaş, E. (2020). Design and implementation of encryption/decryption architectures for bfv homomorphic encryption scheme. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 28(2):353–362.
- Morland, W. (2017). *K-anonymity: An Introduction*. Accessed on 03/10/2019, available at <https://www.privitar.com/listing/k-anonymity-an-introduction>.
- Mudd, J., Allen, J., Baker, J., and Jenvey, P. (2020). *Jython: Python for the Java Platform*. Accessed on 15-10-2020, available at <https://github.com/jython/jython>.
- Nakov, S. (2019). *Digital Signatures*. Accessed on 12-10-2020, available at <https://cryptobook.nakov.com/digital-signatures>.
- Narayanan, A. and Shmatikov, V. (2008). Robust de-anonymization of large sparse datasets.
- Naveed, M., Kamara, S., and Wright, C. V. (2015). Inference attacks on property-preserving encrypted databases. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15*, page 644–655, New York, NY, USA. Association for Computing Machinery.
- Nguyen, A. (2019). *Understanding Differential Privacy*. Accessed on 04/10/2019, available at <https://towardsdatascience.com/understanding-differential-privacy-85ce191e198a>.

## BIBLIOGRAPHY

- Nguyen-Van, T., Nguyen-Anh, T., Le, T., Nguyen-Ho, M., Nguyen-Van, T., Le, N., and Nguyen-An, K. (2019). Scalable distributed random number generation based on homomorphic encryption. pages 572–579.
- Nielsen, J. (2009). *Powers of 10: Time Scales in User Experience*. Accessed on 22-10-2020, available at <https://www.nngroup.com/articles/powers-of-10-time-scales-in-ux/>.
- NIST (2020a). *Block Cipher Techniques – Current Modes*. Accessed on 13-10-2020, available at <https://csrc.nist.gov/projects/block-cipher-techniques/bcm/current-modes>.
- NIST (2020b). *Block Cipher Techniques – Project Overview*. Accessed on 13-10-2020, available at <https://csrc.nist.gov/projects/block-cipher-techniques>.
- Niu, S., Chen, L., Wang, J., and Yu, F. (2020). Electronic health record sharing scheme with searchable attribute-based encryption on blockchain. *IEEE Access*, 8:7195–7204.
- OpenDNSSEC (2009). *SoftHSM*. Accessed on 02/01/2021, available at <https://www.opendnssec.org/softhsm/>.
- Oracle (2014). *Java Native Interface*. Accessed on 14-10-2020, available at <https://docs.oracle.com/javase/8/docs/technotes/guides/jni/index.html>.
- Oracle (2017). *Java Cryptography Architecture (JCA) Reference Guide*. Accessed on 13-10-2020, available at <https://docs.oracle.com/javase/8/docs/technotes/guides/security/crypto/CryptoSpec.html>.
- Patient-Centered Outcomes Research Institute (2019). Building ways to link health information while maintaining patient privacy. Accessed on 08/11/2019, available at <https://www.pcori.org/research-results/2017/building-ways-link-health-information-while-maintaining-patient-privacy>.
- Peng, Z. (2019). *Danger of using fully homomorphic encryption: A look at Microsoft SEAL*. Accessed on 24/01/2021, available at <https://arxiv.org/abs/1906.07127v1>.
- Personal Data Protection Commission Singapore – PDPC (2018). *Guide to basic data anonymisation techniques*. Available at [https://www.pdpc.gov.sg/-/media/Files/PDPC/PDF-Files/Other-Guides/Guide-to-Anonymisation\\_v1-\(250118\).pdf](https://www.pdpc.gov.sg/-/media/Files/PDPC/PDF-Files/Other-Guides/Guide-to-Anonymisation_v1-(250118).pdf).
- Phillips, M., Knoppers, B. M., Baker, D., and Kaufmann, P. (2018). *Privacy-Preserving Record Linkage: Ethico-Legal Considerations*. Accessed on 08/10/2019, available at [https://www.irdirc.org/wp-content/uploads/2018/03/Rare-Genetic-Diseases-Workshop\\_final-version\\_public.pdf](https://www.irdirc.org/wp-content/uploads/2018/03/Rare-Genetic-Diseases-Workshop_final-version_public.pdf).

- Pomroy, S. (2017). *Static Versus Dynamic Data Masking*. Accessed on 01/10/2019, available at <https://www.imperva.com/blog/static-versus-dynamic-data-masking/>.
- Pournaghi, S., Bayat, M., and Farjami, Y. (2020). *MedSBA: a novel and secure scheme to share medical data based on blockchain technology and attribute-based encryption*. Accessed on 28/01/2020, available at <https://doi.org/10.1007/s12652-020-01710-y>.
- Prasser, F., Eicher, J., Spengler, H., Bild, R., and Kuhn, K. A. (2020). *Flexible data anonymization using ARX – Current status and challenges ahead*.
- Prezioso, M. (2019). *Password Hashing: Scrypt, Bcrypt and ARGON2*. Accessed on 12/10/2020, available at <https://medium.com/analytics-vidhya/password-hashing-pbkdf2-scrypt-bcrypt-and-argon2-e25aaf41598e>.
- Privacy Guy (2017). *What is Encryption & How Does It Work?* Accessed on 15/10/2019, available at <https://medium.com/searchencrypt/what-is-encryption-how-does-it-work-e8f20e340537>.
- Ptacek, T. (2011). *Javascript Cryptography Considered Harmful*. Available at <https://www.nccgroup.com/us/about-us/newsroom-and-events/blog/2011/august/javascript-cryptography-considered-harmful/>.
- Qaosar, M., Zaman, A., Siddique, M. A., Li, C., and Morimoto, Y. (2020). Secure k-skyband computation framework in distributed multi-party databases. *Information Sciences*, 515:388 – 403.
- Qin, B., Chen, Y., Huang, Q., Liu, X., and Zheng, D. (2020). Public-key authenticated encryption with keyword search revisited: Security model and constructions. *Information Sciences*, 516:515–528.
- QualityQuick Open Source Software (2019). *Transcrypt – Python in the browser, precompiled for speed*. Accessed on 15-10-2020, available at <https://github.com/qquick/Transcrypt>.
- Rajkumar, D. M. N., George, A., and Batley, B. (2014). An overview of multi-authority attribute based encryption techniques. *International Journal of Advanced Research in Computer and Communication Engineering*, 3.
- Ren, W., Tong, X., Du, J., Wang, N., Li, S. C., Min, G., Zhao, Z., and Bashir, A. K. (2021). Privacy-preserving using homomorphic encryption in mobile iot systems. *Computer Communications*, 165:105 – 111.
- Sarafianou, E. and Mogyorosi, M. (2019). *How Secure Are Encryption, Hashing, Encoding and Obfuscation?* Accessed on 15/10/2019, available at <https://auth0.com/blog/how-secure-are-encryption-hashing-encoding-and-obfuscation/>.

## BIBLIOGRAPHY

- Segers, T. (2020). *Simplified GDPR compliance using MPC cryptography, UN and EC studies explain*. Accessed on 24/01/2021, available at <https://medium.com/applied-mpc/simplified-gdpr-compliance-using-mpc-cryptography-un-and-ec-studies-explain-b2c21ecd0d7b>.
- Sehili, Z., Franke, M., Gladbach, M., Rohde, F., and Rahm, E. (2018). *Privacy Preserving Record Linkage (PPRL)*. Accessed on 07/10/2019, available at [https://dbs.uni-leipzig.de/research/projects/pper\\_big\\_data](https://dbs.uni-leipzig.de/research/projects/pper_big_data).
- Sethi, K., Pradhan, A., and Bera, P. (2020). Practical traceable multi-authority cp-abe with outsourcing decryption and access policy updation. *Journal of Information Security and Applications*, 51:102435.
- Sethi, K., Pradhan, A., Punith, R., and Bera, P. (2019). A scalable attribute based encryption for secure data storage and access in cloud. pages 1–8.
- Shen, C., Lu, Y., and Li, J. (2020). Expressive public-key encryption with keyword search: Generic construction from kp-abe and an efficient scheme over prime-order groups. *IEEE Access*, 8:93–103.
- Shinde, G. and Al-Rummana, G. (2018). Homomorphic encryption for big data security a survey. *International Journal of Computer Sciences and Engineering*, 6:503–511.
- Smirnof, P. and Turner, D. M. (2019). *Symmetric Key Encryption - why, where and how it's used in banking*. Accessed on 17/10/2019, available at <https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking>.
- Song, Y., Wang, H., Wei, X., and Wu, L. (2019). Efficient attribute-based encryption with privacy-preserving key generation and its application in industrial cloud. *Security and Communication Networks*, 2019.
- Soria-Comas, J. and Domingo-Ferrer, J. (2016). *Big Data Privacy: Challenges to Privacy Principles and Models*. *Data Science and Engineering*, 1:21–28.
- SpamLaws (2009). *Data Encryption Pros And Cons*. Accessed on 11/10/2019, available at [https://www.spamlaws.com/pros\\_cons\\_data\\_encryption.html](https://www.spamlaws.com/pros_cons_data_encryption.html).
- Tao, X., Lin, C., Zhou, Q., Wang, Y., Liang, K., and Li, Y. (2019). Secure and efficient access of personal health record: a group-oriented ciphertext-policy attribute-based encryption. *Journal of the Chinese Institute of Engineers*, 42(1):80–86.

- The Royal Society (2019). *Protecting privacy in practice: The current use, development and limits of Privacy Enhancing Technologies in data analysis*. Available at <https://royalsociety.org/-/media/policy/projects/privacy-enhancing-technologies/privacy-enhancing-technologies-report.pdf>.
- tutorialspoint (2015). *Cryptography Benefits & Drawbacks*. Accessed on 11/10/2019, available at [https://www.tutorialspoint.com/cryptography/benefits\\_and\\_drawbacks.htm](https://www.tutorialspoint.com/cryptography/benefits_and_drawbacks.htm).
- upb.crypto developers – Paderborn University (2020). *upb.crypto.craco*. Accessed on 15-10-2020, available at <https://github.com/upbcuk/upb.crypto.craco>.
- Varri, U., Pasupuleti, S., and Kadambari, K. V. (2019). A scoping review of searchable encryption schemes in cloud computing: taxonomy, methods, and recent developments. *The Journal of Supercomputing*.
- Vatsalan, D., Sehili, Z., Christen, P., and Rahm, E. (2017). Privacy-preserving record linkage for big data: Current approaches and research challenges. *Handbook of Big Data Technologies*.
- Venkata Rao, J., Krishna Reddy, V., and Pavan Kumar Hota, C. P. (2020). Enhanced ciphertext-policy attribute-based encryption (ecp-abe). pages 502–509.
- Volgushev, N., Schwarzkopf, M., Getchell, B., Varia, M., Lapets, A., and Bestavros, A. (2019). Conclave: Secure multi-party computation on big data. In *Proceedings of the Fourteenth EuroSys Conference 2019*, EuroSys '19, New York, NY, USA. Association for Computing Machinery.
- Wang, P., Xiang, T., Li, X., and Xiang, H. (2020). Public key encryption with conjunctive keyword search on lattice. *Journal of Information Security and Applications*, 51:102433.
- Wasmer Team (2020). *Wasienv*. Accessed on 15-10-2020, available at <https://github.com/wasienv/wasienv>.
- Watts, S. (2018). *Data Masking: An Introduction*. Accessed on 01/10/2019, available at <https://www.bmc.com/blogs/data-masking/>.
- Wikipedia, the free encyclopedia (2017a). *Encryption*. Accessed on 11/10/2019, available at <https://en.wikipedia.org/wiki/Encryption>.
- Wikipedia, the free encyclopedia (2017b). *Key generation*. Accessed on 15/10/2019, available at [https://en.wikipedia.org/wiki/Key\\_generation](https://en.wikipedia.org/wiki/Key_generation).

## BIBLIOGRAPHY

- Williams, A., Håkansson, J., and Rust Wasm Working Group Core Team (2020). *wasm-pack – Your favorite Rust to Wasm workflow tool!* Accessed on 14-10-2020, available at <https://github.com/rustwasm/wasm-pack>.
- Winton and UC Berkeley (2018). *Using Differential Privacy to Protect Personal Data*. Accessed on 04/10/2019, available at <https://www.winton.com/research/using-differential-privacy-to-protect-personal-data>.
- Witkowski, D. (2020). *Argon2 in browser*. Accessed on 12-10-2020, available at <https://github.com/antelle/argon2-browser>.
- Wu, C.-F., Ti, Y.-W., Kuo, S.-Y., and Yu, C.-M. (2019). Benchmarking dynamic searchable symmetric encryption with search pattern hiding. In *2019 International Conference on Intelligent Computing and its Emerging Applications (ICEA)*, pages 65–69.
- Wu, D. and Lewi, K. (2016). *FastORE*. Accessed on 15-10-2020, available at <https://github.com/kevinlewi/fastore>.
- Xu, L., Yuan, X., Steinfeld, R., Wang, C., and Xu, C. (2019). Multi-writer searchable encryption: An Iwe-based realization and implementation. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security, Asia CCS '19*, page 122–133, New York, NY, USA. Association for Computing Machinery.
- Yadav, G. (2017). *Cryptographic Key Storage Options & Best Practices*. Accessed on 15-10-2020, available at <https://www.globalsign.com/en/blog/cryptographic-key-management-and-storage-best-practice>.
- Yamada, Y., Rohloff, K., and Oguchi, M. (2019). Homomorphic encryption for privacy-preserving genome sequences search. pages 7–12.
- Yang, K., Jia, X., and Ren, K. (2015). Secure and verifiable policy update outsourcing for big data access control in the cloud. *IEEE Transactions on Parallel and Distributed Systems*, 26(12):3461–3470.
- Yao, X., Chen, Z., and Tian, Y. (2015). A lightweight attribute-based encryption scheme for the internet of things. *Future Generation Computer Systems*, 49:104 – 112.
- Yin, H., Qin, Z., Zhang, J., Deng, H., Li, F., and Li, K. (2020). A fine-grained authorized keyword secure search scheme with efficient search permission update in cloud computing. *Journal of Parallel and Distributed Computing*, 135:56 – 69.

- Yin, H., Xiong, Y., Zhang, J., Ou, L., Liao, S., and Qin, Z. (2019). A key-policy searchable attribute-based encryption scheme for efficient keyword search and fine-grained access control over encrypted data.
- Zeutro, LLC (2020). *OpenABE*. Accessed on 14-10-2020, available at <https://github.com/zeutro/openabe>.
- Zhang, X., Wu, F., Yao, W., Wang, Z., and Wang, W. (2018). Multi-authority attribute-based encryption scheme with constant-size ciphertexts and user revocation.
- Zhang, X., Wu, F., Yao, W., Wang, Z., and Wang, W. (2019). Multi-authority attribute-based encryption scheme with constant-size ciphertexts and user revocation. *Concurrency and Computation: Practice and Experience*, 31(21).
- Zhang, X., Xu, C., Wang, H., Zhang, Y., and Wang, S. (2019a). Fs-peks: Lattice-based forward secure public-key encryption with keyword search for cloud-assisted industrial internet of things. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1.
- Zhang, Y., Xu, C., Ni, J., Li, H., and Shen, X. S. (2019b). Blockchain-assisted public-key encryption with keyword search against keyword guessing attacks for cloud storage. *IEEE Transactions on Cloud Computing*, pages 1–1.
- Zhou, X., Ye, H., and Ye, J. (2019). Attribute-based encryption without abuse of private keys. In Abawajy, J. H., Choo, K.-K. R., Islam, R., Xu, Z., and Atiqzaman, M., editors, *International Conference on Applications and Techniques in Cyber Intelligence ATCI 2019*, pages 1475–1480, Cham. Springer International Publishing.
- Zhu, R., Cassel, D., Sabry, A., and Huang, Y. (2018). Nanopi: Extreme-scale actively-secure multi-party computation. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18*, page 862–879, New York, NY, USA. Association for Computing Machinery.



---

## USERS' PERSPECTIVE ON THE EASE OF EXERCISING THEIR RIGHTS

---

Even if the companies claim to respect all the regulations for data protection, some show disrespect for users by resorting to unethical UI or business strategies, as specified by Lomas.

The **dark pattern design** is a UI strategy that allows the companies to intentionally deceive users, making them to give up more than they realize, or to agree to things they probably wouldn't if they genuinely understood the decisions they were being pushed to make.

One key element of dark pattern design is **manipulative timing**. In other words, when the user sees a notification can determine how he responds to it, or even if he notices it. For example, it is often used dark patterns to obtain consent to collect users' personal data, by combining an unwelcome interruption with a built-in escape route, like a brightly colored "agree and continue" button, that will allow the user to proceed to what he is trying to do. Furthermore, the dark pattern design will ensure that the opt-out button, if there is one, will be nearly invisible.

**Friction** is another key tool of this dark pattern: designs that require lots of clicks/taps and interactions if the user wants to opt-out are used to influence the user not to do it. A real example of this usage of dark pattern design is the "Público" website, which presents a long list of companies with whom the data is shared on the cookies selection screen. This website always had a button that allows the user to select all companies, but there was a period when it didn't have one to deselect all of them. Moreover, it was discovered at the time that the option to deselect all was present on the website, but it was purposely hidden with CSS (Martins, 2019).

Deceptive designs can also make it appear that opting out is not even possible. One way to do it is to opt-in by default, and require the users to locate a hard-to-spot alternative click if they try to find a way to opt-out. E-commerce sites sometimes suggestively present an optional (priced) add-on in a way that makes it appear like an obligatory part of the transaction. Airlines have also been caught using deceptive design to upsell pricier options, such as by obscuring cheaper flights and/or masking prices so it's harder to figure



out what the most cost-effective choice actually is. Facebook also used that technique (**hard-to-spot opt-out**) to link WhatsApp users' accounts with Facebook accounts in 2016.

Another often used deceptive design that aims to manipulate online consent flows, works against users by presenting a few selectively **biased examples**. These examples give the illusion of helpful context around a decision to the user, but they are an attempt to manipulate him by presenting a self-centered skewed view that is in no way a full and balanced picture of the consequences of the consent. Facebook, for example, used this technique to encourage European users to switch on its face recognition technology.

**Emotional manipulation** is another method to persuade users to opt-in. Facebook also used it in the previous example, by playing on people's fears (claiming its tech will "help protect you from a stranger"), and people's guilt or sense of goodwill (claiming user's consent will be helpful to people with visual impairment).

Platforms certainly remain firmly in control of everything, until a court tells them otherwise: they control not just the buttons and levers, but their positions, sizes, colors, and ultimately their presence.

What changed is that there are attempts to legally challenge digital dishonesty, especially around privacy and consent. Europe's GDPR has tightened the requirements around consent and is creating the possibility of compensation via penalties worth the enforcement. It has already caused some data-dealing businesses to pull the plug entirely or exit Europe.

A concrete example of privacy and GDPR violation is the Nónio platform, as emphasized by the page *Campanha contra o Nónio*. Nónio intended to be a unique login platform developed with the alliance of some of the biggest Portugal media groups, to offer users more customized content with security and quality (according to them). In the registration, the platform would have access to an enormous data quantity. If Nónio had been successful and all media had been forced to adhere to the system, the users would have to make their register on the platform if they wanted to read articles. This platform violates users' privacy because it collects through cookies and other fingerprinting techniques data as the Internet Protocol (IP) address of each session, date and time of access to the article, browser version, operating system, screen resolution, location data, Wi-Fi access points, and canvas signatures. The platform collects and stores all users' article history of the signed websites and that data can be accessed by third-parties. Analyzing these data allows deducting, for example, political affiliation, religion, buying ability, emotional state, and sleep patterns. Even if the user wants to remove all his data, it is only deleted or anonymized one year after the deactivation of the user register, which is a violation of the right to be forgotten of the GDPR.

So, what should exist in the future is the “**light pattern design**”, which is a user-centric way, at least when it comes to privacy and consent issues. This means genuinely asking for permission, using no privacy-hostile defaults (opt-ins, no opt-outs), where the consent is freely given because it can be revoked at will and it's based on genuine information, not self-serving deception.

---

## LEGAL PERSPECTIVE ON CUSTOMER RIGHTS – GOOGLE VS MEO

---

In this annex, a summary of the information given by Google and Altice/MEO to their users/customers regarding personal data processing is presented. To collect the mentioned information, Google and Altice privacy policies were analyzed, as well as an MEO contract model. To facilitate the comparative analysis, that information is divided into main subjects and presented in a tabular format.

### Collected Information

Table 6: Collected information.

Type of Information	Google	MEO
<b>Personal Information</b>	<ul style="list-style-type: none"> <li>• Name</li> <li>• Phone number</li> <li>• Email address</li> <li>• ...</li> </ul>	<ul style="list-style-type: none"> <li>• Name</li> <li>• Address</li> <li>• Phone number</li> <li>• Email address</li> <li>• Identification Card / Citizen's Card / Passport number</li> <li>• NIF</li> <li>• Birthdate (optional)</li> <li>• Signature</li> <li>• Personal code to access the online extract of the mobile voice service</li> <li>• ...</li> </ul>

---

Table 6: Collected information (continuation).

Type of Information	Google	MEO
<b>Payment information</b>	If the user expressly gives that information or Google participates in the transaction.	If user expressly gives that information, by: <ul style="list-style-type: none"> <li>• Using MEO Wallet.</li> <li>• Authorizing SEPA (Single Euro Payments Area) direct debit payment.</li> </ul>
<b>Content created, saved or received</b>	<ul style="list-style-type: none"> <li>• Documents</li> <li>• Files</li> <li>• Comments</li> <li>• Emails</li> <li>• ...</li> </ul>	On MEO Cloud, for example, any information that may be hosted, made available or transmitted through the service, such as data, text, software, music, graphics, photos, images, sounds, videos, messages and any other of a similar nature.
<b>Apps, browsers or device information</b>	<ul style="list-style-type: none"> <li>• Unique identifiers</li> <li>• Browser type and settings</li> <li>• Device type and settings</li> <li>• Operating system</li> <li>• Mobile network information</li> <li>• Application version number</li> <li>• IP address</li> <li>• Crash reports</li> <li>• System activity</li> <li>• Date, time, and referrer URL of the request</li> <li>• Installed apps</li> </ul>	—

Table 6: Collected information (continuation).

Type of Information	Google	MEO
<p><b>Users activity and interaction with the services</b></p>	<ul style="list-style-type: none"> <li>• Searched terms</li> <li>• Watched videos</li> <li>• Views and interactions with content and ads</li> <li>• Voice and audio information when audio features are used</li> <li>• Purchase activity</li> <li>• People with whom the user communicates or shares content</li> <li>• Activity on third-party sites and apps that use Google services</li> <li>• Chrome browsing history, when synced with the Google Account</li> </ul> <p>Google Hangouts, Google Voice, and Google Fi:</p> <ul style="list-style-type: none"> <li>• Calling-party and receiving-party number</li> <li>• Forwarding numbers</li> <li>• Time and date of calls and messages</li> <li>• Duration of calls</li> <li>• Routing information</li> <li>• Types of calls</li> <li>• Voicemail greeting(s)</li> <li>• Voicemail messages</li> <li>• Short Message Service (SMS) messages</li> <li>• Recorded conversations</li> </ul>	<p>Consumption or profile data – Data resulting from the adhesion and use of the services, through the features that integrate them.</p> <p>Traffic data – Any data processed to send information through an electronic communications network or for billing purposes, such as:</p> <ul style="list-style-type: none"> <li>• Subscriber number or ID, address and type of device (fixed/mobile)</li> <li>• Total number of units to charge for the counting period</li> <li>• Communications type, start date/time and duration</li> <li>• Volume of data transmitted</li> <li>• Receiving-party service and number</li> <li>• Other payment information</li> </ul> <p>MEO may record communications made to the Customer Support Service.</p> <p>Data Service – Mobile and WI-FI Internet access:</p> <ul style="list-style-type: none"> <li>• Personal Identification Code (“Username”)</li> <li>• Internet Network Entry Code (“Access Code”/“Password”)</li> <li>• Date and time of Mobile Internet Service communications.</li> </ul>

Table 6: Collected information (continuation).

Type of Information	Google	MEO
<b>Location</b>	<p>Data used to obtain user's location:</p> <ul style="list-style-type: none"> <li>• GPS (precise location)</li> <li>• IP address (general location)</li> <li>• Sensor data from the user's device (for example, an accelerometer can be used to determine the user's speed, and a gyroscope to figure out the user's direction of travel)</li> <li>• Information about things near the user's device (such as Wi-Fi access points, cell towers, and Bluetooth-enabled devices) – Google Location Services (GLS)</li> </ul>	<p>Location data – Any data processed on an electronic communications network or within an electronic communications service that indicates the geographical position of the terminal equipment of a user of a publicly available electronic communications service.</p>

**Privacy Controls**

Table 7: Privacy controls.

Type of Controls	Google	MEO
<b>User activity</b>	<p>Activity controls (signed-in) – Allows deciding what types of activity the user would like saved in his account (e.g., Location History or YouTube Watch History).</p>	–

Table 7: Privacy controls (continuation).

Type of Controls	Google	MEO
<b>Payments</b>	—	SEPA Direct Debit Payment Authorization – Allows the user’s bank to debit his account under MEO instructions. The authorization has no immediate effect.
<b>Ads</b>	Ad settings (signed-in/out) – Allows the user to manage his preferences about the ads shown on Google and other sites, as well as apps that partner with Google to show ads. Even if the user opts out of Ads Personalization, he may still see ads based on factors such as the general location derived from his IP address, his browser type, his current search terms, and the topic of the website or app he’s looking at, but not on user Google Account Activity, interests, search history, browsing history, or locations history. However, user information can still be used for other purposes, such as to measure the effectiveness of advertising, and protect against fraud and abuse.	Processing and communication of personal data – Allows the user to authorize or not (at any time) the processing of their data (customer’s traffic data, geographical location, profile, or consumption) for MEO marketing communications, as well as sharing them with Altice Portugal Group companies for that same purpose.
<b>Search personalization</b>	Search personalization (signed-out) – The user can choose whether his search activity is used to offer him more relevant results and recommendations.	—
<b>Videos history</b>	YouTube settings (signed-out) – Allows pausing and deleting the user’s YouTube Search History and YouTube Watch History.	—

Table 7: Privacy controls (continuation).

Type of Controls	Google	MEO
<b>User personal information</b>	<p>About the user (signed-in) – The user can control what others see about him across Google services.</p> <p>Shared endorsements (signed-in) – Allows choosing whether the user name and photo appear next to his activity, like reviews and recommendations that appear in ads.</p>	<p>Customers are guaranteed at any time the right to limit their personal data. For example, the client can authorize or not:</p> <ul style="list-style-type: none"> <li>• MEO to provide the Simplified Information Sheet, as well as other information regarding the provision of the contracted service, in the Client Area available at <i>meo.pt</i>.</li> <li>• Information services: <ul style="list-style-type: none"> <li>– Disclosure of user data to the 1820 Information Service.</li> <li>– Disclosure of user data when the search is not based on his name, but rather on the telephone number or address.</li> <li>– Transmission of user data to third parties for publication in information services.</li> </ul> </li> </ul>
<b>People to share information with</b>	<p>Information you share (signed-in) – Allows the user to control whom he shares information with, through his account on Google+.</p>	<p>MEO Kanal – Allows the user to create a channel and to share it with a restricted group of users or all the users of the service.</p> <p>MEO Cloud – Allows the user to access the platform, store and/or access files, and make them accessible on all his terminal equipment, while also allowing to share them with third parties.</p>



## Goals of Data Collection

Table 8: Goals of Data Collection.

Google	MEO
<p>Provide, maintain, and improve Google services, as well as help on new services development.</p>	<p>Manage the contractual relationship and provide the contracted services.</p>
<p>Provide personalized services, including recommendations, content, search results, and ads.</p> <ul style="list-style-type: none"> <li>• Personalized ads based on sensitive categories (such as race, religion, sexual orientation, or health) are not shown.</li> <li>• Information that personally identifies the customer is not shared with advertisers unless the client requests it.</li> </ul>	<p>Adapt the services to the needs and interests of the user:</p> <ul style="list-style-type: none"> <li>• Access to service-specific features.</li> <li>• Suggested content.</li> <li>• Proximity information services (in particular on service pharmacies, proximity services, etc.).</li> <li>• Information and marketing actions.</li> <li>• Inclusion of clients in subscriber lists or directories.</li> </ul> <p>Disseminate the Group's institutional information and/or publicizing campaigns, promotions, advertising, and news about Altice Portugal products and/or services.</p>
<p>Measure performance (of Google services or third-party ad campaigns).</p>	<p>Conduct market research or evaluation surveys.</p>
<p>Protect Google, its users, and the public. For example:</p> <ul style="list-style-type: none"> <li>• Collect and analyze IP addresses and cookie data to protect against automated abuse attacks, such as sending spam emails, stealing money from advertisers by fraudulently clicking on their ads, Distributed Denial of Service (DDoS) attacks, and more.</li> <li>• Gmail's "last account activity" feature lets you find out if, when and where (IP address) someone accessed your email without your knowledge.</li> </ul>	<p>—</p>

Table 8: Goals of Data Collection (continuation).

Google	MEO
<p data-bbox="432 405 459 427">—</p> <p data-bbox="135 748 523 781">Communicate with the customer:</p> <ul data-bbox="140 801 762 1010" style="list-style-type: none"> <li data-bbox="140 801 762 891">• Google &gt; User: suspicious login, upcoming changes or improvements to services, etc.</li> <li data-bbox="140 913 762 1010">• User &gt; Google: a record of the user’s request is kept to help solving any issues he might be facing.</li> </ul>	<p data-bbox="831 398 1145 432">Social intervention actions.</p> <p data-bbox="831 477 1222 510">Communicate with the customer:</p> <ul data-bbox="836 530 1461 1021" style="list-style-type: none"> <li data-bbox="836 530 1461 846">• All communications from MEO to the customer may be made by any means of contact provided by the customer, such as an address, e-mail address, automatic voice message broadcasting system, SMS, and, when applicable, through the display on equipment used by the customer.</li> <li data-bbox="836 869 1461 1021">• If the customers wish to contact MEO, they may do so by using the contacts made available in their customer area on <i>meo.pt</i> (if they are registered users).</li> </ul> <p data-bbox="831 1081 1281 1115">Court communications and summons:</p> <ul data-bbox="836 1135 1461 1339" style="list-style-type: none"> <li data-bbox="836 1135 1461 1339">• The client’s summons and judicial notifications will be made to the address agreed for this purpose, the change of which must be notified in writing by the client to MEO.</li> </ul>

## Data Processing

Table 9: Data Processing – Legal grounds.

Google	MEO
<p>Information only is processed in the following situations:</p> <ul style="list-style-type: none"><li>• With user consent.</li><li>• For Google's and third parties' legitimate interests while applying appropriate safeguards that protect users' privacy. This means that the information can be processed for things like:<ul style="list-style-type: none"><li>– Provide advertising to make many of Google services freely available for users.</li><li>– Detect, prevent, or otherwise address fraud, abuse, security, or technical issues with Google services.</li><li>– Protect against harm to the rights, property or safety of Google, its users, or the public as required or permitted by law, including disclosing information to government authorities.</li><li>– Perform research that improves Google services for users and benefits the public.</li></ul></li><li>• To provide a service that has been requested by the user, under a contract.</li><li>• To comply with legal obligations.</li></ul>	<p>Information only is processed in the following situations:</p> <ul style="list-style-type: none"><li>• With permission of the data subject:<ul style="list-style-type: none"><li>– Personal, traffic, geographic location, profile, and/or consumption data are processed for marketing or disclosure purposes of Altice Portugal's goods or services.</li><li>– Personal data is further processed (and possibly transmitted to third parties) for the dissemination of information services, within the scope of the universal service.</li></ul></li></ul> <p>If there is prior consent of the user, it may be withdrawn at any time, without the legality of the treatment based on the prior consent given being compromised.</p> <ul style="list-style-type: none"><li>• To provide services and/or products, as well as to process personal data at the social intervention level according to more detailed information made available to the respective holders.</li><li>• To comply with legal standards regarding the retention and transmission of data for investigation, detection, and prosecution of serious crimes, among other legally binding treatments.</li><li>• Location information can be recorded and transmitted to organizations with legal competence to receive emergency calls.</li></ul>

Table 10: Data Processing – Types of treatment.

Google	MEO
<ul style="list-style-type: none"> <li>• Automated systems to analyze contents, to provide customized search results, personalized ads, detect spam, malware, and illegal content, etc.</li> <li>• Algorithms to recognize patterns in data:               <ul style="list-style-type: none"> <li>– Make sense of images: For example, face detection.</li> <li>– Voice search: For each voice query made, data such as the language, the country, and Google system’s guess of what was said is stored.</li> </ul> </li> </ul>	<p>The collected personal data may be processed by a computer, in an automated or non-automated manner, guaranteeing in all cases strict compliance with the personal data protection legislation. The collected data is stored in specific databases created for this purpose and under no circumstances it will be used for a purpose other than the ones for which it was collected or the data subject has given consent to.</p>

## Data Sharing

Table 11: Data Sharing – Legal Grounds.

Google	MEO
<p>Data protection laws vary among countries, with some providing more protection than others. Regardless of where the user’s information is processed, Google applies the same protections and complies with certain legal frameworks regarding data transfer, such as the EU-US and Swiss-US Privacy Shield Frameworks.</p>	<p>The provision of certain services by Altice Portugal may involve data transfer outside Portugal, including outside the European Union or to International Organizations. In such case, Altice Portugal will strictly comply with the applicable legal provisions, in particular, the determination of the suitability of the country(ies) of destination concerning the protection of personal data and the requirements applicable to such transfers, including, when applicable, the establishment of appropriate contractual instruments guaranteeing and respecting the legal requirements in force.</p>

Table 11: Data Sharing – Legal Grounds (continuation).

<b>Google</b>	<b>MEO</b>
<p>Google doesn't share user's personal information with companies, organizations, or individuals outside of Google, except in the following cases:</p> <ul style="list-style-type: none"> <li>• With user consent.</li> <li>• With domain administrators.</li> <li>• For external processing (in compliance with Google Privacy Policy and any other appropriate confidentiality and security measures).</li> <li>• For legal reasons.</li> <li>• Only non-PII is involved, such as when: <ul style="list-style-type: none"> <li>– Publicly sharing information to show trends about the general use of Google services.</li> <li>– Google-specific partners collect information from user's browser or device for advertising and measurement purposes, using their cookies or similar technologies.</li> </ul> </li> </ul>	<p>Altice Portugal may communicate the personal data of the client, to fulfill legal obligations, namely to police, judicial, tax, and regulatory entities.</p> <p>Altice Portugal, as part of its activity, may use third parties to provide certain services. Sometimes the provision of these services implies access by these entities to the personal data of customers/users. When this happens, Altice Portugal takes appropriate measures to ensure that entities that have access to the data are reputable and offer the highest guarantees at this level, which is duly established and contractually agreed between Altice Portugal and the third-party entity(s).</p> <p>Thus, any entity subcontracted by Altice Portugal will treat the user's personal data on behalf of Altice Portugal and take the necessary technical and organizational measures to protect personal data against accidental or unlawful destruction, accidental loss, alteration, diffusion or unauthorized access, and against any other form of unlawful treatment.</p> <p>In any case, Altice Portugal remains responsible for the personal data made available.</p>
<p>If Google is involved in a merger, acquisition, or sale of assets, it will continue to ensure the confidentiality of user's personal information and give affected users notice before personal information is transferred or becomes subject to a different privacy policy.</p>	<p style="text-align: center;">—</p>

## Data Review/Update

Table 12: Data Review/Update.

Google	MEO
<p>If EU data protection law applies to the processing of user's information, Google provides some controls so the user can exercise his right to request access to, update, remove, and restrict the processing of his information. He also has the right to object to the processing of his information.</p>	<p>As holders of personal data, Customers/Users are guaranteed at any time the right to access, rectify and update their personal data.</p>
<ul style="list-style-type: none"><li>• My Activity – Allows to review and control data that are created when Google services are used. It can be browsed by date and by topic and deleted part or all of the user activity.</li><li>• Google Dashboard – Allows managing information associated with specific products.</li><li>• Personal information – Allows the user to manage contact information, such as his name, email, and phone number.</li></ul>	

## Data Exportation

Table 13: Data exportation.

Google	MEO
<p>The user can export a copy of the content in his Google Account if he wants to back it up or use it with a service outside of Google.</p>	<p>The customer has the right to data portability.</p>

## Data Removal/Elimination

Table 14: Data removal – General information.

Google	MEO
<p>Some data the user can delete whenever he likes, some data is deleted automatically, some data Google retains for longer periods when necessary.</p> <p>The user may delete his call history, voicemail greeting(s), voicemail messages (both audio and/or transcriptions), SMS messages, and recorded conversations through his Google Voice account, although his call history for billable calls may remain visible on his account.</p>	<p>The period for which personal data is stored and retained varies according to the purpose for which the information is processed.</p> <p>When there is no specific legal requirement, data will be stored and retained only for the minimum period necessary to achieve the purposes for which it was collected or further processed, as defined by law.</p> <p>As holders of personal data, customers are guaranteed at any time the right to delete their personal data, except for data that are indispensable for service provisioning by Altice Portugal or the fulfillment of legal obligations to which the responsible for the treatment is subject.</p>
<p>Sometimes business and legal requirements oblige Google to retain certain information, for specific purposes, for an extended period:</p> <ul style="list-style-type: none"> <li>• Security, fraud and abuse prevention.</li> <li>• Financial record-keeping.</li> <li>• Complying with legal or regulatory requirements.</li> <li>• Ensuring the continuity of Google services.</li> <li>• Direct communications with Google.</li> </ul> <p>Some data is kept until Google Account is erased if it is useful to help understand how users interact with Google functionalities and how Google services can be better.</p>	<p>MEO can retain certain information for an extended period, such as in cases like the following:</p> <ul style="list-style-type: none"> <li>• Specific legal requirements.</li> <li>• For communication and billing purposes, traffic data will be processed for a maximum of 6 months from the date of registration.</li> <li>• The customer's geographic location, profile and/or consumption data will be processed as far as they are essential for the provision of the contracted services and the duration of the provision.</li> <li>• MEO may record communications made to the Customer Support Service for quality of service monitoring, the recording of which will be maintained for the legally provided period.</li> </ul>

Table 15: Data removal – Details specified only by Google.

<b>Google</b>	
<b>Data that expires after a specific period of time</b>	<p>For each type of data, Google sets retention time-frames based on the reason for its collection and anonymizes certain data within set time periods. For example:</p> <ul style="list-style-type: none"> <li>• To ensure that the services display properly on many different types of devices, Google may retain browser width and height for up to 9 months.</li> <li>• Google anonymizes advertising data in server logs by removing part of the IP address after 9 months and cookie information after 18 months.</li> </ul>
<b>Safe and complete deletion</b>	<p>There may be delays between when the user deletes something and when copies are deleted from our active and backup systems, once Google tries to ensure that their services protect information from accidental or malicious deletion.</p> <p>Data is completely and securely removed from servers (or is only retained anonymously) accordingly with the following steps:</p> <ol style="list-style-type: none"> <li>1. Immediately removes the deleted data from the view and it may no longer be used to personalize user's Google experience.</li> <li>2. Safely and completely deletes the data from Google storage systems, which are used to protect users and customers from accidental data loss. Each Google storage system from which data gets deleted has its own detailed process for safe and complete deletion. This might involve repeated passes through the system to confirm all data has been deleted or brief delays to allow recovery from mistakes. The complete data removal from servers generally takes around 2 months from the time of deletion, but additional time may be required to completely and securely delete the data. This often includes up to a month-long recovery period in case the data was removed unintentionally.</li> </ol> <p>Google services also use encrypted backup storage as another layer of protection to help recover from potential disasters. Data can remain on these systems for up to 6 months.</p>

## Data Security



Table 16: Data Security.

Google	MEO
<p>Google tries to protect the user and itself from unauthorized access, alteration, disclosure, or destruction of information that Google hold:</p>	<p>Altice Portugal has adopted several security measures, both technical and organizational, to protect the personal data that is made available to the user against its unauthorized dissemination, loss, misuse, alteration, processing, or access, as well as against any other form of illicit treatment:</p>
<ul style="list-style-type: none"> <li>• Usage of encryption to keep user data private while in transit.</li> <li>• Offer a range of security features, like Safe Browsing, Security Checkup, and 2 Step Verification to help the user to protect his account.</li> <li>• Review of Google information collection, storage, and processing practices, including physical security measures, to prevent unauthorized access to its systems.</li> <li>• Restrict access to personal information to Google employees, contractors, and agents who need that information to process it. Anyone with this access is subject to strict contractual confidentiality obligations and may be disciplined or terminated if they fail to meet these obligations.</li> <li>• The numbers of credit and debit cards provided to Google are encrypted and stored on secure servers in an also secure location.</li> </ul>	<ul style="list-style-type: none"> <li>• On all Altice Portugal websites, personal data collection forms require encrypted browser sessions. All personal data provided is securely stored in Altice Portugal's systems, which in turn are located in an Altice Portugal Datacenter, covered by all physical and logical security measures that Altice Portugal deemed essential for the protection of users' personal data.</li> <li>• MEO guarantees that the electronic communications networks used to provide the services meet the necessary and appropriate requirements for the security of the same and the network itself, but cannot guarantee its inviolability by unauthorized third parties.</li> </ul>
<p>To maintain its commitment to user privacy, Google uses anonymization, strict controls on user data access, policies to control and limit the joining of data sets that may identify users, and the centralized review of anonymization and data governance strategies to ensure a consistent level of protection across all of Google.</p>	<p>MEO will not be responsible for improper access by third parties or data deletion, destruction, damage, modification, or loss, as well for any loss or damage caused by misuse of the personal code of access to the online extract that is not attributable to the company. It is considered performed by the customer, the use of the service by third parties using the codes provided by MEO unless proven otherwise.</p>

## Cookies

Table 17: Cookies – Concept and utility examples.

Google	MEO
<p>Google offers services that let website operators target their ads to people who visited their pages. For this to work, Google either reads a cookie that's already in the user's browser or places a cookie in it when the user visits the Website (assuming that the user's browser lets this happen).</p> <p>To publish ads on services where cookie technology is not available, such as mobile applications, Google may use technologies that perform similar functions to cookies. Google sometimes links the identifier used for mobile app advertising to an advertising cookie on the same device to coordinate ad serving between mobile apps and browsers. This can happen, for example, when the user sees an ad in an app that opens a web-page in the mobile browser. This also helps improve the reporting available to advertisers on the effectiveness of their campaigns.</p>	<p>Cookies are used to help determine the usefulness, interest, and number of uses of websites, enabling faster and more efficient browsing and eliminating the need to repeatedly enter the same information.</p> <p>There are two groups of cookies according to the period in which they are stored:</p> <ul style="list-style-type: none"> <li>• Permanent cookies – These are cookies that are stored at the browser level on access devices (PC, mobile, and tablet) and are used whenever the user revisits one of the Altice Portugal websites. They are generally used to direct navigation to the user's interests, enabling them to provide a more personalized service.</li> <li>• Session cookies – These are temporary cookies that remain in the browser cookie file until you leave the website, log out of the session, or clear cookies. The information obtained by these cookies serves to analyze web traffic patterns, allowing them to identify problems and provide a better browsing experience.</li> </ul>

Table 18: Cookies – Types of cookies, according to their features.

Google	MEO
<p>Preferences – Allow Google websites to remember information that changes the way the site behaves or looks, such as user's preferred language, text size, etc.</p>	<p>Functionality cookies – Store user preferences regarding the use of the site so that the user does not have to reconfigure the site each time he visits it.</p>

Table 18: Cookies – Types of cookies, according to their features (continuation).

Google	MEO
<p>Security – These cookies are used to authenticate users, prevent fraudulent use of login credentials, and protect user data from unauthorized parties. For example, Google uses the cookies ‘SID’ and ‘HSID’, which contain digitally signed and encrypted records of a user’s Google account ID and most recent sign-in time. The combination of these two cookies allows blocking many types of attacks, such as attempts to steal the content of forms that the user completes on web pages.</p>	<p>Strictly Required Cookies – Allows website navigation and application usage, as well as access to secure areas of the website. Without these cookies, the required services cannot be provided.</p>
<p>Processes – Help to make the website work and deliver services that the visitor expects, like navigating around web pages or accessing secure areas of the website. Without these cookies, the website cannot function properly.</p>	
<p>Advertising – Allows making advertising more engaging to users, and more valuable to publishers and advertisers. Some common applications of cookies are to select advertising based on what’s relevant to a user, to improve reporting on campaign performance, and to avoid showing ads that the user has already seen.</p>	<p>Advertising cookies – They target advertising based on the interests of each user, to target advertising campaigns to users’ tastes, and limit the number of times they see the ad, helping to measure the effectiveness of advertising and the success of the website organization.</p>
<p>Session State – These cookies are used to collect information about how users interact with a website. This may include the pages users visit most often, whether users get error messages from certain pages, or anonymously measure the effectiveness of Pay Per Click (PPC) and affiliate advertising.</p>	<p>Third-party cookies – Measure the success of applications and the effectiveness of third-party advertising. They can also be used to customize a widget with user data.</p>

Table 18: Cookies – Types of cookies, according to their features (continuation).

Google	MEO
<p>Analytics – Google Analytics may use cookies to collect information and report website usage statistics without personally identifying individual visitors to Google. This tool can also be used, together with some advertising cookies, to help show more relevant ads and to measure interactions with the ads Google shows.</p>	<p>Analytical Cookies – These are used anonymously to create and analyze statistics to improve the functioning of the website.</p>

Table 19: Cookies – Management.

Google	MEO
<p>The user can manage the Google ads he sees and to opt out of Ads Personalization, as well as manage cookies in general through the web browser. He can also manage many companies' cookies used for online advertising via the consumer choice tools created under self-regulation programs in many countries.</p>	<p>The user can accept the use of cookies through the “Accept” button in the informational phrase on Al-tice Portugal sites, or make his choices and management on the informational page through the available browsers and tools. However, disabling cookies may prevent some web services from functioning properly.</p>

## Complaints

Table 20: Complaints.

Google	MEO
<p>The user can send formal written complaints to Google, to which Google responds by contacting the person who made the complaint or, when the problem cannot be solved directly with the user, by working with the appropriate regulatory authorities, including local data protection authorities.</p>	<p>Despite being able to submit complaints directly to Al-tice Portugal, through the contacts made available for this purpose, the client/user may complain directly to the Control Authority, which is the National Data Protection Commission (CNPD), using the contacts provided by this entity for that purpose.</p>

## Data Anonymization

Table 21: Data anonymization.

---

### Google

---

Anonymization is a data processing technique that removes or modifies personally identifiable information. It results in anonymized data that cannot be associated with any one individual.

By analyzing anonymized data, Google can build safe and valuable products and features, like autocompletion of an entered search query, and better detect security threats, like phishing and malware sites, all while protecting user identities. Google can also safely share anonymized data externally, making it useful for others without putting the privacy of the users at risk.

- Generalizing the data – Certain data elements are more easily connected to certain individuals. To protect those individuals, Google uses generalization to remove a portion of the data or replace some part of it with a common value. For example, it can be used generalization to replace segments of all area codes or phone numbers with the same sequence of numbers.

Generalization allows us to achieve  $k$ -anonymity, a term used to describe a technique for hiding the identity of individuals in a group of similar persons. In  $k$ -anonymity, the  $k$  is a number that represents the size of a group. If for any individual in the data set, there are at least  $k-1$  individuals who have the same properties, then we have achieved  $k$ -anonymity for the data set.

If all individuals in a data set share the same value of a sensitive attribute, sensitive information may be revealed simply by knowing these individuals are part of the data set in question. To mitigate this risk, it may be leveraged  $l$ -diversity, a term used to describe some level of diversity in the sensitive values.

- Adding noise to data – Differential privacy describes a technique for adding mathematical noise to data. With differential privacy, it's difficult to ascertain whether an individual is part of a data set because the output of a given algorithm will essentially appear the same, regardless of whether an individual's information is included or omitted. It's also important to note that adding noise to a data set may render it less useful.
-

## Shared database

Table 22: Shared database.

---

### MEO

---

In the event of non-compliance with the obligation to pay invoices for the provision of services, the user's personal data may be included in a shared database created under the terms of the law, which makes it possible to identify customers who have not fulfilled this obligation.

If desired, the user can remedy the breach of contract by paying the amount owed through the means provided by MEO or demonstrating the unenforceability or non-existence of the debt, requiring MEO to notify the customer of this possibility at least 5 working days from the date of inclusion in the shared database.

In the event of inclusion of the data in the shared database, such inclusion will be communicated to the client within 5 working days of its realization. MEO guarantees to the customer the right of access, rectification, and updating of their data, as well as their immediate deletion from that database, after payment of the debts in question.

---

### Final Considerations

After a detailed analysis of the information provided by both companies to users or customers, it is clear that much of it is common to both entities, with an effort on both parts to comply with the GDPR. This concern with ensuring user's privacy emphasizes the need to use privacy and security mechanisms in data processing. If the user chooses to trust the company with his/her data, it is that entity's responsibility to ensure data protection.

It is important to note, however, that even if the companies provide all this information, including the general purposes for data collection, they do not specify how the corresponding treatment will be carried out. Disclosing the mechanisms used to ensure data security, for example, would be quite useful for this work, but it could also cause some security issues with the associated services.

---

## ADDITIONAL INFORMATION ABOUT PETS

---

### C.1 ANONYMIZATION AND PSEUDONYMIZATION

**BIG DATA CONSIDERATIONS** There are some aspects of anonymization (and pseudonymization) specific to big data (ENISA – European Union Agency for Cybersecurity, 2015), such as:

- **Controlled linkability** – Big data anonymization should be compatible with linking data from several (anonymized) sources, while fulfilling the other two usual goals of anonymization (prevent re-identification and attribute disclosure).
- **Composability** – A privacy model is composable if its privacy guarantees hold for a dataset constructed by linking together several datasets, for each of which the privacy guarantee of the model holds.
- **Anonymization of dynamic/streaming data** – Most of the anonymization literature refers to static datasets. However, in big data, continuous data streams are very frequent.
- **Computability for large data volumes** – In big data, computational efficiency may be a critical issue when choosing a privacy model or an anonymization method.
- **Decentralized anonymization** – This paradigm reduces the need for data subjects to trust the controller, which is positive in a typical big data scenario.

**TRADE-OFFS** Perfect anonymization/pseudonymization is difficult in practice without compromising the utility of the dataset. In big data, this problem increases due to the amount and variety of data. So, the challenge is to protect privacy, keeping an acceptable disclosure risk, with minimum loss of accuracy.

**APPROACHES** Regarding anonymization for big data, there are different approaches that can be used according to where it takes place (ENISA – European Union Agency for Cybersecurity, 2015):

- Centralized – The anonymization is performed by a data controller who has access to the entire original dataset. On one hand, with this approach, the data controller is in the best position to optimize the trade-off between data utility and extant disclosure risk. On the other hand, this requires all parties that provide original data to trust the data controller, which may not happen in a big data scenario. Additionally, anonymization can be a computational burden too heavy for a single controller, especially in the case of big data.

Using this approach, the mainly known anonymization strategies (Cigniti Technologies, 2015; Pomroy, 2017; Watts, 2018) are:

- Static Data Masking – The important data is masked in the original database environment. This process is usually performed on the golden copy of the database, but it can also be applied to values in other sources, including files.

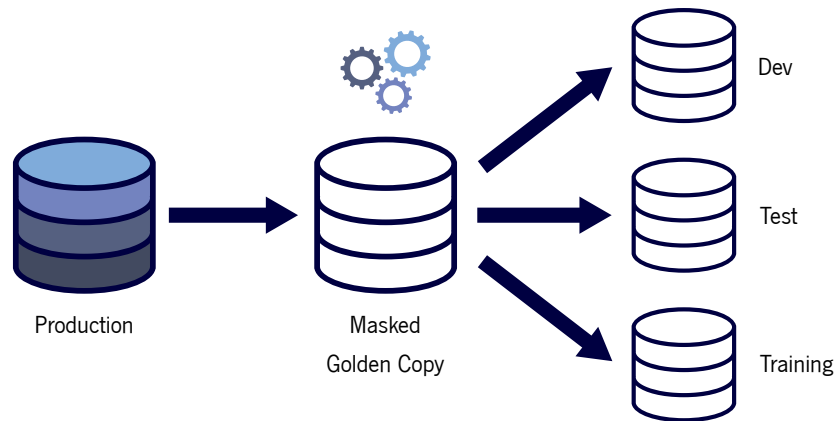


Figure 32: High-level static data masking architecture.

- On-the-Fly Data Masking – The important data is masked on demand, in the process of transferring it from environment to environment. It is masked within the memory of a given database application, which is particularly useful for agile companies focused on continuous delivery.

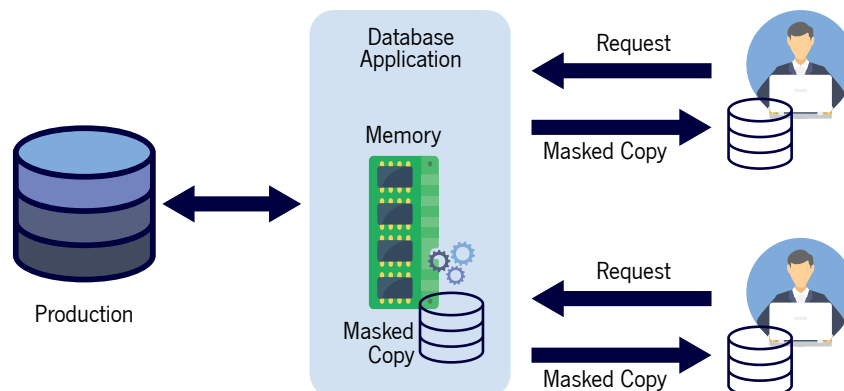


Figure 33: High-level on-the-fly data masking architecture.



- Dynamic Data Masking – This is the attribute-based and policy-driven process of masking production data when the data request is actually made (on demand). There are two types of dynamic masking:
  - \* View Based Masking – Maintains the production version and the masked version of the data in the same database. Users who are not approved to view production data or who trigger the security filter in any way are shown masked data. The decision to show masked or production data is made in real-time based on pre-programmed rules.
  - \* Proxy-Based Masking – Introduces a proxy layer between the user and the database. The user query goes through the proxy which substitutes the result of the query with masked values. This provides data protection without the need to alter the database. Another technique is query substitution, which intercepts and redirects the query to retrieve data from masked columns. Such queries are very flexible and can pick masked data from a view or a file, or even link to another database.

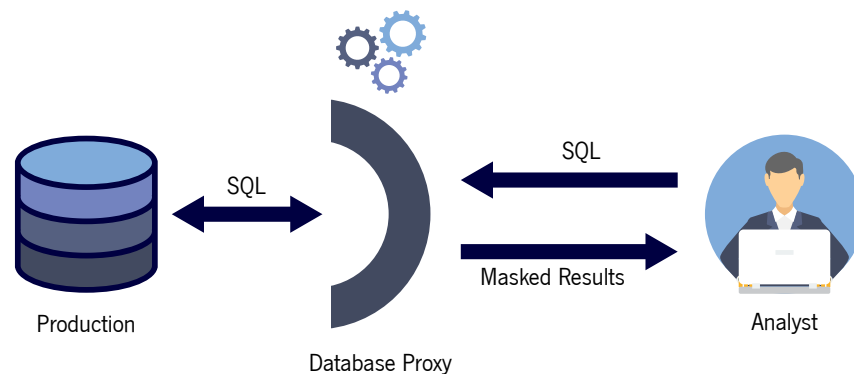


Figure 34: High-level dynamic (proxy-based) data masking architecture.

Overall, the selection of an anonymization strategy must take into account the size of the organization, as well as the location (cloud or on-premises) and complexity of the data to protect.

- Local (decentralized) – Each subject anonymizes his own data before handing them to the data controller. It is suitable for scenarios where the individuals do not trust (or trust only partially) the data controller. In comparison to centralized anonymization, local anonymization usually results in greater information loss. The reason is that each individual needs to protect his data without seeing the other individuals' data, which makes it difficult for him to find a good trade-off between the disclosure risk limitation and the information loss.
- Co-utile collaborative (decentralized) – Allows to generate, in a collaborative and distributed manner, an anonymized dataset that satisfies the following conditions:

### C.1. Anonymization and Pseudonymization

1. It incurs no more information loss than the dataset that would be obtained with the centralized approach for the same privacy level;
2. Neither the data subjects nor the data controller gain more knowledge about the confidential attributes of any other specific data subject than the knowledge contained in the final anonymized dataset.

Collaborative anonymization is feasible because it leverages the co-utility principle, that is, mutual utility. Note that if one data subject from a set of  $n$  subjects is re-identified, the remaining  $n - 1$  become easier to re-identify (as the crowd in which they are hiding gets thinner). Hence, data subjects are rationally interested in collaborating for anonymity.

**PRIVACY BREACH** In the context of anonymization (and pseudonymization), privacy can be compromised by means of two types of disclosure:

- Identity Disclosure or Entity Disclosure – Occurs when a record in the anonymized dataset can be linked with a particular individual.
- Attribute Disclosure – Occurs when the value of a confidential attribute of an individual can be determined more accurately with access to the released data than without.

Depending on their disclosure potential, attributes of a dataset can be classified as several non-disjoint types:

- Identifiers – Attributes that unambiguously identify the data subject to whom a record corresponds. Examples are passport number, social security number, and full name. Identifiers are removed as a precondition towards obtaining an anonymized dataset.
- Quasi-Identifiers (QIs) or key attributes – They identify the data subject with some ambiguity. In combination, they can be linked with external information to re-identify (some of) the subjects to whom (some of) the records in the original dataset refer. Examples are job, age, gender, city of residence, and phone number. Unlike identifiers, QIs cannot be removed as part of the anonymization process, because they often have high analytical value. Furthermore, any attribute is potentially a QI.
- Confidential outcome attributes – They contain sensitive information on the data subject. Examples are salary, religion, and health condition.
- Non-confidential outcome attributes – Other attributes which contain non-sensitive data subject information.

Disclosure risk comes from QIs, so anonymization procedures must deal with them.

GDPR In the context of GDPR, the differences between anonymization and pseudonymization are reinforced. Because pseudonymous data still allows for some form of re-identification, it is still considered personal data, unlike anonymous data. The principles of data protection should therefore not apply to anonymous information.

So, under the GDPR, there are different obligations to each type of de-identification, as summarized by Table 23.

Table 23: Different obligations under the GDPR regarding identified, pseudonymized and anonymized data.

<b>GDPR Obligation</b>	<b>Identified</b>	<b>Pseudonymized</b>	<b>Anonymized</b>
1. Provide notice to data subject	Required	Required	Not Required
2. Obtain consent or have another legal basis	Required	Potentially helps	Not Required
3. Give right to erasure / right to be forgotten	Required	Depends on strength	Not Required
4. Other data subject rights (access, portability...)	Required	Depends on strength	Not Required
5. Basis for cross-border transfers	Required	Required	Not Required
6. Data protection by design	Required	Partially met	Not Required
7. Data security	Required	Partially met	Not Required
8. Data breach notification	Likely	Less likely	Not Required
9. Data retention limitations	Required	Required	Not Required
10. Documentation / recordkeeping obligations	Required	Required	Not Required
11. Vendor / sub-processor management	Required	Required	Not Required

## C.2 K-ANONYMITY AND ITS EXTENSIONS

PROPERTIES The main *k*-anonymity model properties (European Commission, 2014; Soria-Comas and Domingo-Ferrer, 2016; Personal Data Protection Commission Singapore – PDPC, 2018) are:

- Guideline – The *k*-anonymity model is used as a guideline before and after anonymization techniques have been applied, to ensure any record’s direct and/or indirect identifiers are shared by at least  $k - 1$  other records.
- Composability – Despite *k*-anonymous data releases being in general not composable, in some restricted scenarios composability may be satisfied. If it can be guaranteed that there are no overlapping subjects or confidential attributes in the *k*-anonymous data releases, *k*-anonymity is preserved. However, it seems difficult to guarantee such conditions in a big data environment.
- Singling out – Because the same attributes are shared by *k* records, creating an equivalence class with *k* users, it should be no longer possible to single out an individual within a group of *k* users.
- Linkability – While linkability is limited, it remains possible to link records by groups of *k* users. Furthermore, if some of the confidential attributes are shared between the datasets to be linked, the accuracy of the linkage improves. It could even be possible to accurately link individual records.
- Inference – The main flaw of the *k*-anonymity model is that it does not prevent any type of inference attack. Indeed, if all *k* individuals are within a same group and it is known which group an individual belongs to, it is trivial to retrieve the value of this property.
- Computational Cost – Finding the minimal generalization or the optimal microaggregation has been shown to be an NP-hard problem, but several approximation heuristics have been proposed that reduce its cost.
- Assess Protection – An anonymized dataset may have different *k*-anonymity levels for different sets of indirect identifiers, but to assess the protection against linking, the lowest *k* is used for comparison against the threshold.

PROCESS First, it is necessary to decide on a value for *k*, which should be the lowest value to be achieved among all equivalence classes (Personal Data Protection Commission Singapore – PDPC, 2018). After the anonymization techniques are applied, it is verified if each record has at least  $k - 1$  other records with the same attributes addressed by the *k*-anonymization. Records in equivalence classes with less than *k* records should be considered for suppression. Alternatively, more anonymization can be performed.

**TRADE-OFFS** Generally, the higher the value of  $k$ , the greater the protection once it is harder to identify data subjects. However, higher values negatively impact the utility of the data as it implies more stringent applications of generalization and suppression techniques (Personal Data Protection Commission Singapore – PDPC, 2018). Anonymization techniques can sometimes be used in combination, but the exact chosen way can affect data utility.

**VARIATIONS/EXTENSIONS**  $k$ -Anonymization is able to prevent identity disclosure, which means that a record in a  $k$ -anonymized dataset cannot be mapped back to the corresponding record in the original dataset. However, it may fail to protect against attribute disclosure when the value of a confidential attribute is the same or very similar in all  $k$  records sharing the same combination of QI values. A number of  $k$ -anonymity extensions exist to address this basic vulnerability to inference attacks (ENISA – European Union Agency for Cybersecurity, 2015):

- $p$ -Sensitive  $k$ -anonymity – A dataset is said to satisfy  $p$ -sensitive  $k$ -anonymity for  $k > 1$  and  $p \leq k$  if it satisfies  $k$ -anonymity and, for each group of records with the same combination of QI attribute values, the number of distinct values for each confidential attribute within the group is at least  $p$ .
- $l$ -Diversity – A dataset is said to satisfy  $l$ -diversity if, for each group of records sharing a combination of QI attribute values, there are at least  $l$  “well-represented” values for each confidential attribute. This means that  $l$  is the minimal size of a group in the context of homogenous groups. There are several definitions of “well-represented”, such as:
  - a) the  $l$  values are merely distinct (in this case  $l$ -diversity is equivalent to  $l$ -sensitive  $k$ -anonymity);
  - b) Shannon’s entropy of the confidential attribute values within each group is at least  $\log_2(l)$ ;
  - c) recursive  $l$ -diversity, which requires that the most frequent values do not appear too frequently and the least frequent values do not appear too rarely.
- $t$ -Closeness – A dataset is said to satisfy  $t$ -closeness if, for each group of records sharing a combination of QI attribute values, the distance between the distribution of the confidential attribute in the group and the distribution of the attribute in the whole dataset is no more than a threshold  $t$ .
- $(n,t)$ -Closeness – A relaxation of  $t$ -closeness. In this case, for each group of records sharing a combination of QI attribute values, the distance between the distribution of the confidential attribute in the group and the distribution in a superset of the group with at least  $n$  records should be no more than a threshold  $t$ .

COMPARISONS The main differences between *k*-anonymity and its extensions (ENISA – European Union Agency for Cybersecurity, 2015; Personal Data Protection Commission Singapore – PDPC, 2018) are highlighted below:

- The presented *k*-anonymity extensions do not improve *k*-anonymity regarding unlinkability. The issue is the same as with any cluster: the probability that the same entries belong to the same data subject is higher than  $1/N$  (where  $N$  is the number of data subjects in the database).
- The main improvement of the *k*-anonymity extensions over *k*-anonymity is that setting up inference attacks with a 100% confidence is no longer possible. *p*-Sensitive *k*-anonymity and *l*-diversity make sure that every attribute has at least *p* different or *l* “well-represented” values in each equivalence class. However, these extensions cannot prevent the leakage of information if the attributes within a partition are unevenly distributed or belong to a small range of values or semantic meanings, because these techniques do not take into account the semantics of the confidential attribute values. So, *p*-Sensitive *k*-anonymity and *l*-diversity are vulnerable to probabilistic inference attacks. Such attacks are better countered by *t*-closeness.

#### APPLICATION EXAMPLE

##### Google’s Password Checkup

Google released the new Password Checkup Chrome extension. Whenever a user signs into a site, Password Checkup will trigger a warning if the used username and password pair is one of over 4 billion credentials that Google knows to be unsafe.

Password Checkup was designed jointly with cryptography experts at Stanford University to ensure that Google never learns the user’s username or password, and that any data breach stays safe from wider exposure. Also, all statistics reported by the extension are anonymous. These metrics include the number of lookups that surface an unsafe credential, whether an alert leads to a password change, and the web domain involved for improving site compatibility. Since Password Checkup is an early experiment, Google shared the technical details behind its privacy preserving protocol to be transparent about how it keeps user’s data secure.

At a high level, Password Checkup needs to query Google about the breach status of a username and password without revealing the information queried. At the same time, Google needs to ensure that no information about other unsafe usernames or passwords leaks in the process, and that brute force guessing is not an option. Password Checkup addresses all of these requirements by using multiple rounds of hashing, *k*-anonymity, and private set intersection with blinding.

Font: <https://security.googleblog.com/2019/02/protect-your-accounts-from-data.html>

### C.3 DIFFERENTIAL PRIVACY AND RELATED MODELS

PROPERTIES The main properties of the differential privacy model (Soria-Comas and Domingo-Ferrer, 2016; Nguyen, 2019) are:

- **Guideline** – Privacy loss is a measure in any differential privacy mechanism or algorithm. It allows comparisons among different techniques. Privacy loss is controllable and ensures a trade-off between it and the accuracy of general information.
- **Composability** – The privacy loss quantification allows the analysis and control of the cumulative loss over multiple data releases. By accumulating differentially private data about a set of individuals, differential privacy is not broken, but the level of privacy decreases.
- **Singling out** – Differential privacy follows the principle that adding any one person's data to a dataset should not materially change the result of queries run on that data. So, it should not be possible to single out people's data using specific queries. Furthermore, differential privacy allows the analysis and control of privacy loss incurred by groups, such as families.
- **Linkability** – In partially synthetic data, the subjects in the original dataset are also present in the synthetic dataset. Thus, if data about a subject are collected and anonymized independently, it will be possible to link the records. In fully synthetic data, a new sample from the underlying population is used to generate the new dataset. Thus, even if data about the same subject are collected by different sources, there is no guarantee that the subject will be present in the synthetic datasets.

For the case of a dataset generated by masking the values of the original records, the discussion is similar. If the values used in the linkage have not been masked, then it can be expected perfect linkage accuracy. However, it must be kept in mind that the values of the masked attributes are no longer the original values. If the values used in the linkage have been masked, the differentially private datasets are not linkable.

- **Computational Cost** – For the case of a synthetically generated differentially private dataset, the computational cost is highly dependent on the model used to approximate the distribution of the population. The alternative approach consists of computing a differentially private histogram, whose cost is proportional to the number of histogram bins. The number of bins for the joint histogram grows exponentially with the number of attributes.

For the case of aggregation plus noise, the computational cost and the amount of noise needed to attain differential privacy are determined by the type of aggregation used. There are microag-

gregation techniques proposed with a quasilinear cost on the number of records  $n$  of the dataset:  $\mathcal{O}(n \ln(n))$ .

- Closure Under Post-Processing – Differential privacy is immune to post-processing, which means that a data analyst without additional knowledge about the private database cannot compute a function of the output of a differentially private algorithm and make it less differentially private.

**PROCESS** The mechanism used by differential privacy is to add deliberate errors to data so that even if it were possible to recover data about an individual, there would be no way to know whether that information was meaningful or nonsensical. One useful feature of this approach is that the errors deliberately introduced into the data roughly cancel each other out when the data is aggregated (Winton and UC Berkeley, 2018).

There are two main approaches to generate differentially private datasets (Soria-Comas and Domingo-Ferrer, 2016):

- (i) Create a synthetic dataset from a differentially private model for the data (usually from a differentially private histogram);
- (ii) Add noise to mask the values of the original records (probably in combination with some prior aggregation function to reduce the amount of required noise).

**TRADE-OFFS** A key issue in differential privacy is that adding noise can harm utility (The Royal Society, 2019). Intuitively, for population-level statistics, the more individuals included in a dataset, the harder it might be to identify that a specific individual was included. Therefore, less noise needs to be added in order to protect privacy.

In other words, with differential privacy there is a better trade-off of utility and privacy with larger datasets, where noise will have less of an impact on the output. In machine learning, achieving differential privacy also goes hand in hand with preventing overfitting to particular examples.

There is a security parameter called “privacy budget”, which is a limit after which a user is not allowed to perform any more queries. Setting its overall value requires careful consideration of the statistical inferences that might happen after the release of results and how, for example, outsiders might be able to link data with side information.

**VARIATIONS/EXTENSIONS**  $\epsilon$ -Differential privacy and some of its variants are described next (ENISA – European Union Agency for Cybersecurity, 2015):



- $\epsilon$ -Differential Privacy – The parameter  $\epsilon$ , which is the information leaked about a specific entity, increases linearly with the number of queries and can be set as the limit after which a user is not allowed to perform any more queries (“privacy budget”). This  $\epsilon$  parameter allows one to reason about the level of privacy protection desired. A randomized function  $\kappa$  that returns the query answer plus some noise satisfies  $\epsilon$ -differential privacy if, for all datasets  $D1$  and  $D2$  that differ in one record (neighbor datasets) and all  $S \subset \text{Range}(\kappa)$ , it holds that  $\Pr(\kappa(D1) \in S) \leq \exp(\epsilon) \times \Pr(\kappa(D2) \in S)$ . This means that the presence of any original record needs to be unnoticeable within  $\exp(\epsilon)$  in the anonymized dataset. So, it is hard to preserve any significant utility unless  $\epsilon$  is large (often much larger than 1), in which case the privacy guarantee is no longer that strong.
- Crowd-Blending Privacy – Can be viewed as a relaxation of differential privacy with a view to improve utility. A dataset satisfies  $k$ -crowd blending privacy if each individual record  $i$  in the dataset “blends” with  $k$  other records  $j$  in the dataset. This means that the output of the randomized query function  $\kappa$  is “indistinguishable” if  $i$  is replaced by any of the records  $j$ . Thus, differential privacy is relaxed into crowd-blending privacy by transforming a requirement involving the entire dataset to a requirement involving only a group of records containing a specific record.
- Blowfish – Can be regarded as a relaxation or a generalization of differential privacy: it uses the same condition, but it changes the definition of neighbor datasets. Whereas in differential privacy neighbor datasets  $D1$  and  $D2$  are defined as those differing in any single record, in Blowfish one can use any definition of neighborhood. If, as a result, the number of neighbors is a strict subset of those in standard differential privacy, then we have a relaxation.

COMPARISONS The main differences between differential privacy and  $k$ -anonymity (ENISA – European Union Agency for Cybersecurity, 2015) are highlighted below:

- It is possible to reach differential privacy from  $t$ -closeness (with a suitable distance) and vice versa. This does not mean that both models are equivalent, but it illustrates that they can provide similar privacy guarantees if the proper parameter choices are made.
- In general,  $k$ -anonymity and its extensions provide less strict privacy guarantees than  $\epsilon$ -differential privacy. However  $k$ -anonymity-like models usually entail less utility loss than  $\epsilon$ -differential privacy.
- $k$ -anonymity and its variants are focusing on anonymizing a dataset before its release for further analysis. Differential privacy, on the other hand, can be used to run queries on the data, following a predefined type of analysis, in a way that the answers do not violate individuals’ privacy.

Although it has been argued that the differential privacy's query-based approach is superior to the "release and forget" approach of  $k$ -anonymity, its practical implementation (taking into account the utility-privacy trade-off) is not possible in every data analytics scenario. Thus,  $k$ -anonymity still has a dominant role, especially on data releases (i.e. when the query-based model is not applicable).

- For both privacy models, computational complexity is very dependent on the anonymization method used to satisfy them.

Differential privacy has some similarities but also important differences in the definition of "anonymization" in data protection law. For instance, in differential privacy, the concern is if the relative difference in the result reveals whether a specific individual or entity is included or excluded in the input. In contrast, "anonymization" in data protection law is concerned about removing any attributes associated with an individual that could make them identifiable in a dataset. The two concepts can be brought together by setting the value of  $\epsilon$  so that the relative difference in the result is so small that it is unlikely that anyone could infer with confidence anything about a specific individual or entity in the input.

#### APPLICATION EXAMPLE

##### US Census

In 2017, the US Census Bureau announced that it would be using differential privacy as the privacy protection mechanism for the 2020 decennial census – this is after having implemented differential privacy for other services (OnTheMap, 2008 – an online application developed in partnership with 50 US states, for creating workforce related maps, demographic profiles, and reports). By incorporating formal privacy protection techniques, the Census Bureau will be able to publish a higher number of tables of statistics with more granular information than previously. By fixing a privacy budget for that given set of released publications, the institution can reason mathematically about the risk of disclosure of information regarding a specific individual. In contrast, the Census Bureau says that such a risk is much less controlled in the case of traditional approaches to statistical disclosure control.

In the light of these benefits, and despite encountering a number of hurdles, the Census Bureau is continuing to pursue its decision to implement differential privacy. The institution is not only implementing differential privacy in its statistical analyses but integrating it into its organizational structure.

Setting the value of the privacy budget  $\epsilon$  has not been trivial. In practice, the value of  $\epsilon$  chosen was far higher than those envisioned by the creators of differential privacy. More efficient mechanisms and proofs are needed to achieve lower amounts of noise for the same level of privacy loss and to make efficient use of the privacy-loss budget for iterative releases of edited and corrected statistics.

*Font: The Royal Society (2019)*

## C.4 PRIVACY-PRESERVING RECORD LINKAGE

PROPERTIES Some PPRL properties are explained next (Franke et al., 2019):

- Scalability – PPRL must achieve high efficiency with scalability to large data volumes, potentially with many data sources. Scalability to large datasets can be addressed by several blocking and filtering techniques that alleviate the inherent quadratic complexity (every record has to be compared with every other record). There are already parallel linkage implementations based on Apache Flink, supporting scalability to millions of records.
- Privacy – A high degree of privacy has to be ensured by PPRL to protect the represented entities' data. For this purpose, secure protocols and encoding (encryption) techniques are used to minimize the risk of disclosing sensitive information. Several masking techniques have been developed, using two different types of general approaches: MPC and data perturbation. These techniques should not only prevent persons' re-identification, but they also must preserve the (dis)similarity between records to allow the correspondence between the encoded records and their plaintext counterparts. In other words, PPRL should ensure that the identity of each person cannot be discovered, but their linkage is still possible. Furthermore, all parties that are involved in the linkage process should have no access to data relevant for analysis, e.g., health-related information.
- Linkage Quality – Despite the use of encoded attribute values, PPRL must achieve a high linkage quality by avoiding false or missing matches. A high precision (avoidance of false matches) is especially important, e.g., to avoid that information about different patients is combined. Linkage quality is largely affected by the quality of the input data, which often contain typographical errors and heterogeneity in structure. To weaken such problems, a preprocessing step to clean and standardize the input data is usually conducted. Besides data quality, many factors significantly influence the final linkage quality, such as the encoding method, the blocking or filtering approach, the respective parametrization, and the selected similarity threshold (which is used to decide whether a record pair represents a match or a nonmatch).

PROCESS The basic idea of the PPRL techniques is to mask (encode) the databases at their sources and to conduct the linkage using only the masked data (Vatsalan et al., 2017). This means no sensitive data are ever exchanged between the organizations involved in a PPRL protocol or revealed to any other party. At the end of the linkage process, the database owners only learn which of their own records match with high similarity records from the other database(s). The next step would be exchanging certain attributes

values of the matched records between the database owners or sending the selected attribute values to a third party, such as a researcher who requires the linked data for their project. Recent research outcomes and experiments conducted in real health data linkage validate that PPRL can achieve linkage quality with only a small loss compared to traditional record linkage using unencoded QIs.

**BIG DATA CONSIDERATIONS** Big data implies enormous data volume, as well as massive flows (velocity) of data, leading to scalability challenges even with advanced computing technology. Also, the variety and veracity aspects of big data require biases, noise, variations, and abnormalities in data to be considered, which makes the linkage process more challenging. Besides that, with big data containing massive amounts of personal data, linking and mining it may breach the privacy of those represented by the data. A practical PPRL solution to be used in real-world applications should, therefore, address these challenges of scalability, linkage quality, and data privacy (Vatsalan et al., 2017).

**TRADE-OFFS** The main trade-off that needs to be considered in PPRL is between linkage quality and performance.

**APPROACHES** There are two basic approaches used in the linkage step (Vatsalan et al., 2017):

- Two-party protocols – Only the two database owners who wish to link their data are involved;

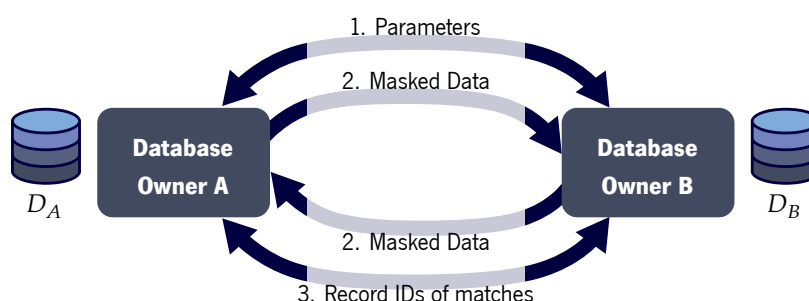


Figure 35: Outline of PPRL two-party protocols.

- Three-party protocols – They use a (trusted) third party to conduct the linkage. This party will never see any unencoded values, but collusion is possible.

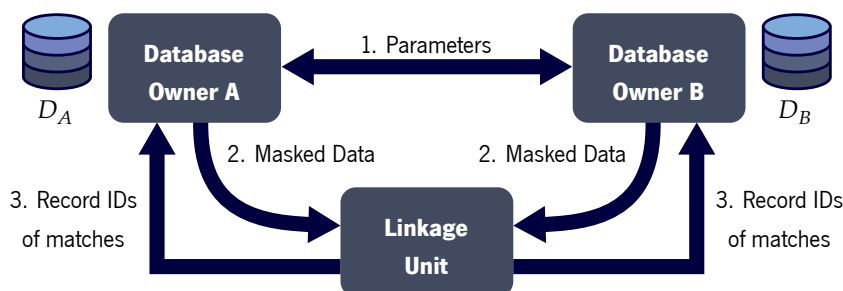


Figure 36: Outline of PPRL three-party protocols.

Besides those basic approaches, there are the multi-party protocols, which focus on linking records from more than two databases, with or without a linkage unit. Some challenges of this approach are matching records that are only in subsets of the sources, exponential complexity with the number of sources, and increased privacy risk of collusion.

An example of the linkage process is illustrated in Figure 37, which presents a three-party protocol commonly deployed, with the usage of a trusted third party.

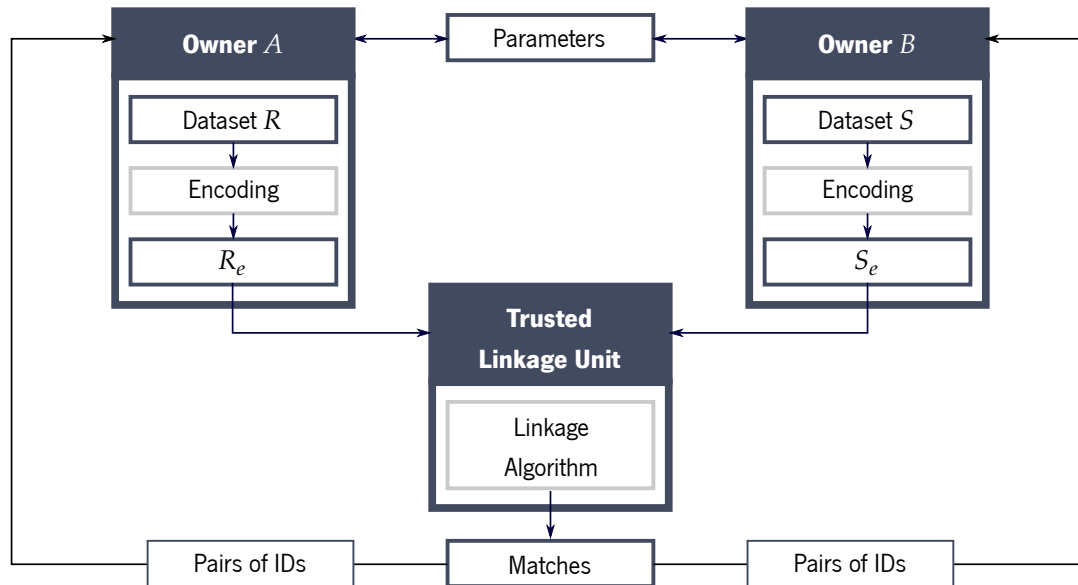


Figure 37: Example of PPRL protocol.

In this example, two data holders  $A$  and  $B$  own the datasets  $R$  and  $S$  respectively. Those data holders and a trusted third party, the linkage unit ( $LU$ ), carry out the linkage step. Three-party protocols inherently support scalability, since the  $LU$  can apply efficient blocking and filter techniques, as well as use large-scale computing resources. Furthermore, this approach can be extended to support matching for more than two data owners. The overall PPRL process consists of the following steps:

1. Preprocessing – The data holders exchange some parameters about the attributes to use in the linkage, and how to clean and standardize them. For example, they exchange the encoding parameters. Then, each data holder conducts a data cleaning and standardization step. This step is essential to achieve a high linkage quality since real-world data is considered to be dirty. After that, data holders  $A$  and  $B$  encode (mask) their records and send the set of encoded records,  $R_e$  and  $S_e$  respectively, to the linkage unit.
2. Blocking/Filtering – To avoid the need for every record of the first source to be compared with every record of the second source, blocking or filtering techniques are used to exclude obviously dissimilar record pairs from detailed comparisons. Although these two methods have the same

#### C.4. Privacy-Preserving Record Linkage

goal and can be used in combination, they operate differently. On one side, blocking is used to group records in blocks such that only records within the same block are compared with each other. On the other side, filter techniques exploit the properties of the similarity functions and the match similarity threshold to filter out dissimilar record pairs that cannot reach or exceed the threshold.

3. Similarity calculation and match classification – Several similarity functions can be used to compare pairs of records and derive the pairs of matching records. To identify matching records, PPRL methods mostly follow a simple threshold-based approach such that two records are considered to represent the same real-world entity if their similarity meets or exceeds a threshold.
4. Matches return – The *LU* returns the pairs of the matching records identifiers to the data holders, which are then able to link their data of interest, such as medical data.

VARIATIONS/EXTENSIONS In PPRL, the linkage has to be conducted on a masked or encoded version of the QIs to preserve entities' privacy. Several approaches have been used to protect data subjects privacy (Vatsalan et al., 2017), such as:

- Pseudo-Random Function (PRF) – Deterministic secure function that, when given an  $n$ -bit seed  $k$  and an  $n$ -bit argument  $x$ , returns an  $n$ -bit string  $f_k(x)$  such that it is infeasible to distinguish  $f_k(x)$  for different random  $k$  from a truly random function. In PPRL, PRFs have been used to generate random secret values to be shared by a group of parties.
- Reference values – Uses as reference randomly generated values or values extracted from a publicly available source in the same domain (e.g., telephone directory). Then, it requires the calculation of similarities between private values using the similarities of each private value with the reference value (triangular inequality).
- Noise addition – Extra (fake) records are included in the datasets to perturb data. This technique overcomes frequency attacks (improves privacy) at the cost of more comparisons and loss in linkage quality (due to false matches).
- Differential privacy – It has emerged as an alternative to random noise addition. In this case, the probability of holding any property on the perturbed database is approximately the same whether or not an individual value is present in the database. The magnitude of noise depends on the privacy parameter and data sensitivity.
- Phonetic encoding – Groups values that have a similar pronunciation together. The main advantage of using a phonetic encoding for PPRL is that it inherently provides privacy, reduces the number

of comparisons (increasing scalability), and supports approximate matching. Two drawbacks of phonetic encodings are that they are language dependent and are vulnerable to frequency attacks.

- Generalization techniques – Overcome the problem of frequency attacks by generalizing records in a way that re-identification of individual records from the masked data is not feasible. Some generalization techniques include  $k$ -anonymity, generalization hierarchies, top-down specialization, and data binning.
- Secure hash-encoding – It is a basic building block of many PPRL protocols. The idea is to use a one-way hash function (like SHA) to encode values and then compare hash-codes. Only having access to hash-codes will make it nearly impossible with current computing technology to infer their original input values. However, a major problem with this masking technique is that only exact matches can be found. Indeed, a single character difference between the two input values results in completely different hash codes. With this technique, dictionary and frequency attacks are possible.
- Statistical Linkage Key (SLK) – It is a derived variable generated from components of QIs. For example, the SLK-581 uses the second and third letters of the first name, the second, third, and fifth letters of the surname, full date of birth, and sex to link records from the Home and Community Care datasets. However, it has already been shown that these SLK-based masking methods provide limited privacy protection and poor sensitivity.
- Embedding space – Embeds QI values into a multi-dimensional metric space while preserving the distances between them, using a set of pivot values that span the multidimensional space. A drawback of this approach is that it is often difficult to determine a good dimension of the metric space and select suitable pivot values.
- Encryption schemes – They are used in PPRL techniques to allow MPC in a way that, at the end of the computation, no party knows anything except its own input and the final results of the computation. Examples of encryption schemes that are used in PPRL are commutative and homomorphic encryption. A drawback of this approach is that it is computationally expensive.
- Bloom filter encoding – It has been used as an efficient masking technique in a variety of PPRL approaches, providing a means of privacy assurance. A Bloom filter  $b_i$  is a bit vector data structure of length  $l$  bits where all bits are initially set to 0.  $k$  independent hash functions,  $h_1, h_2, \dots, h_k$ , each with range  $1, \dots, l$ , are used to map each of the elements  $s$  in a set  $S$  into the Bloom filter, by setting the bit positions  $h_1(s), h_2(s), \dots, h_k(s)$  to 1. So, the bloom filter is a bit vector data

structure into which values are mapped by using a set of hash functions. It is possible to distribute the similarity calculation across all parties (multi-party bloom filter).

- Count-min sketches – Probabilistic data structures (similar to Bloom filters) that can be used to hash-map values along with their frequencies in a sub-linear space. Count-min sketches have been used in PPRL where the frequency of occurrences of a matching pair/set also needs to be identified. However, these approaches only support exact matching of categorical values.

Other privacy aspects in a PPRL project are the secure generation and exchange of public/private key pairs, employee confidentiality agreements to reduce internal threats, encrypted communication, secure connections, and secure servers to reduce external threats.

In order to improve efficiency, the number of comparisons between records can be reduced by adopting blocking or filtering approaches. The matching step can also be performed in parallel on multiple processing nodes. Multiple approaches have been studied to solve the scalability challenge (Vatsalan et al., 2017), such as:

- Blocking Techniques – Most blocking strategies group records into disjoint or overlapping blocks. Blocking for PPRL is based on the known approaches for regular record linkage but aims at improving privacy. A general approach with a linkage unit is for each data owner to apply a previously agreed blocking strategy on the original records. Then, all records within the different blocks are masked (encoded) and sent to the linkage unit. This entity then compares the masked records block-wise with each other. Some examples of blocking strategies for PPRL are:
  - Standard blocking – Uses the values of a so-called blocking key to divide all records into disjoint blocks. The blocking key values may be the values of a selected attribute or the result of a function on one or several attribute values.
  - Phonetic Blocking – The basic idea is to encode the values of a blocking key attribute (e.g., last name) with a phonetic function. All records with the same phonetic code, i.e., with similar pronunciation, are then assigned to the same block.
  - Locality-Sensitive Hashing (LSH) – It has been proposed to solve the problem of the nearest neighbor search in high-dimensional spaces. The basic idea of LSH is to apply a set of hash functions on the objects of interest, i.e., bit vectors. These hash functions are drawn from a family of functions sensitive to a certain distance measure. For each hash function in that family, objects with a small distance (high similarity) must have a much higher probability of a collision, i.e., to obtain the same output value for two different input values, than objects with greater distance (low similarity). It is possible to improve the scalability of PPRL by



parallelizing the PPRL process using the LSH-based blocking approach with the Hamming distance: Hamming Locality-Sensitive Hashing (HLSH).

- Filtering Methods – Filtering strategies use properties of the similarity function, as well as the chosen similarity threshold, to identify pairs that cannot satisfy the match criteria, reducing the search space and thereby speeding up the linkage process.
  - PPJoin – It is a fast implementation for similarity joins, applying several filters to exclude record pairs from the similarity computation, in particular the length, prefix, and position filters. The length filter, for example, excludes all pairs from further consideration if their lengths are very different. There is an adaptation for Bloom filters, called P4Join.
  - The Metric Space Approach – A metric space  $\mathcal{M}(\mathcal{U}, d)$  consists of a set  $\mathcal{U}$  of data objects and a metric distance function  $d$  to compute the distances between those objects. The distance function  $d$  must satisfy several properties, in particular the triangle inequality:  $\forall r, s, p \in \mathcal{U} : d(r, s) \leq d(r, p) + d(p, s)$ , which can be used for filtering pairs of records to find matches. For example, pivot-based is a filtering technique for metric spaces that uses the triangle inequality to reduce the search space.

Regarding the final linkage quality, only moderate quality can be obtained for some datasets, even if the records are only slightly corrupted and all parameters are carefully (empirically) selected (Franke et al., 2019). The reason behind this phenomenon is the existence of datasets where many non-matching record pairs have a high similarity score. For example, datasets containing many families or households tend to have many non-matches with high similarity since many persons share their last name and address. The situation becomes even worse if the dataset also is large or of poor quality. Furthermore, most PPRL approaches only apply a simple threshold-based classification, using a single threshold value. This match criterion often leads to situations where a record has more than one record in the other source for which the similarity threshold is reached. For duplicate-free source databases, this would be incorrect and should thus be avoided.

To weaken this effect, a post-processing step is often applied after the classification to determine the most likely matches. This step ensures that the final matches represent a 1:1-mapping between the two datasets, where one record is linked to at maximum one record of the other source. Different post-processing strategies can be used, such as the heuristic symmetric, also known as Max-Both. Max-Both accepts a candidate pair  $(r, s)$  as a match, only if  $s$  has the maximal similarity with  $r$  among all records in the second source and  $r$  has the maximal similarity with  $s$  in the first source.

COMPARISONS A comparison between some of the mentioned PPRL variations is presented next:

- An extensive evaluation has shown that LSH-based blocking can achieve high quality and high scalability up to millions of records. In contrast, phonetic blocking approaches are susceptible to cryptanalysis and are more sensitive to data quality problems. Moreover, phonetic blocking based on a single attribute is not feasible for large datasets due to the low number of blocks and the data skew impact introduced by frequent names (Sehili et al., 2018).
- Tests have shown that the P4Join filtering method achieves only modest improvements (less than a factor of 2) compared to a naive nested loop scheme (Vatsalan et al., 2017). By contrast, the pivot-based metric space approach achieves order of magnitude improvements.
- Regarding linkage quality, the heuristic symmetric approach is very efficient and effective (high precision) and outperforms other approaches such as the appliance of the Gale-Shapley or the Hungarian algorithms (Franke et al., 2019).

PRIVACY BREACH Different adversary models are assumed in PPRL techniques (Vatsalan et al., 2017):

- Honest-but-curious (HBC) or semi-honest parties – They try to find out as much as possible about another party's input to a protocol while following the protocol steps. If all parties involved in a PPRL protocol have no new knowledge at the end of it, then the protocol is considered to be secure in the HBC model. However, it is important to note that the HBC model does not prevent parties from colluding with each other to learn about another party's sensitive information. Most of the existing PPRL solutions assume the HBC adversary model.
- Malicious parties – They can behave arbitrarily in terms of refusing to participate in a protocol, not following it in the specified way, choosing arbitrary values for their data input, or aborting the protocol at any time. Limited work has been done in PPRL for the malicious adversary model. Evaluating privacy under this model is very difficult because there are potentially unpredictable ways for malicious parties to deviate from the protocol that cannot be identified by an observer.
- Covert and accountable computing models – They are advanced adversary models developed to overcome the problems associated with the HBC and malicious models. On one hand, the covert model guarantees that the honest parties can detect the misbehavior of an adversary with high probability. On the other hand, the accountable computing model provides accountability for privacy compromises by the adversaries without the excessive complexity and cost that incur with the malicious model. Research is required towards transforming existing HBC or malicious PPRL protocols into these models, as well as proving the privacy of solutions under these models.

The main privacy attacks or vulnerabilities to which a PPRL technique is susceptible are:

- Dictionary attack – An adversary masks a large list of known values using multiple existing encoding functions until a matching masked value is identified. A keyed encoding approach, such as HMAC, can be used to prevent dictionary attacks.
- Frequency attack – Even with a keyed masking approach, when the frequency distribution of a set of masked/encoded values is matched with the distribution of known unmasked values, it is still possible to infer the original values of the masked ones.
- Cryptanalysis attack – It is a special category of frequency attack that is applicable to Bloom filter-based data masking techniques.
- Composition attack – It can be successful by combining knowledge from more than one independent masked datasets to learn sensitive values of certain records.
- Collusion – Vulnerability associated with multi-party or three-party PPRL techniques, where some of the parties involved in the protocol work together to find another database owner's data.

GDPR Several aspects must be taken into account to develop a PPRL system in compliance with the GDPR (Phillips et al., 2018), such as:

- When outsourcing personal data to other data owners or the linkage unit, the respective records should be coded, having their direct and Quasi-Identifiers removed and replaced with a re-identification code that is generated independently of the values of identity attributes. So, this outsourced information must be pseudonymized, as defined by the GDPR.
- PPRL systems should exercise caution about including biometrics in the immutable personal data that is fed to the hash function, complying with the GDPR restrictions regarding the processing of biometric data to uniquely identify a natural person.
- The PPRL system should have build-in robust support for the irrevocable right of participants to withdraw their consent any time after giving it.
- There is a risk that data aggregation through linkage may transform data that was not reasonably predictably identifiable into potentially identifiable data. The linkage should thus either require that continued non-identifiability is proven, using measures like  $k$ -anonymity or  $l$ -diversity, or that sufficient alternative safeguards are put in place.
- Under EU data protection law, the linkage entity is likely to be considered a joint controller along with the other entities in the system, once the linkage entity will not process its data solely on the

#### C.4. Privacy-Preserving Record Linkage

instructions of any other single entity. However, the GDPR relieves a controller of duties that would otherwise apply, including fulfillment of the right of access, in cases where the controller is unable to identify the data it holds. This is the case for the linkage unit, even if the data remains personal.

- From the GDPR's perspective, disclosure and transfer must also meet distinct regulatory conditions to be legal.

**APPLICATION EXAMPLE** In medical research, data of several sources (e.g., hospitals) has to be matched to investigate possible correlations between some diseases of the same patients without revealing their identity. An example of investigation in this field is specified next.

#### Patient-Centered Outcomes Research Institute (PCORI)

People get health care from many places, such as doctor's offices, hospitals, and pharmacies. Each place creates and stores patient health records. These records include identifiers or pieces of information that reveal a patient's identity, such as name, birth date, or address. When sharing or using patient health records for research, organizations usually remove identifiers to protect patients' privacy.

Having more complete patient records that use data from more than one source could help improve patient care, public health reporting, and medical research. Researchers need ways to link patient data from different sources while ensuring patient privacy. But linking patient records from different sources can be hard without identifiers. Also, too much health information in one record creates a privacy risk, even if there are no identifiers. People with uncommon conditions might be recognized by their health records.

Currently, there is a research project in progress, named "Building Ways to Link Health Information while Maintaining Patient Privacy", that is funded by Patient-Centered Outcomes Research Institute (PCORI). In this study, the research team is building and using tools to link patient records from multiple sources without identifiers.

*Font: Patient-Centered Outcomes Research Institute (2019)*

## C.5 TRADITIONAL ENCRYPTION

**PROPERTIES** The main encryption properties (Sarafianou and Mogyorosi, 2019) are:

- Confidentiality – It is about protecting information from being accessed by unauthorized parties or, in other words, making sure that only those who are authorized have access to restricted data.
- Unpredictable key – The key's value should be extremely difficult to guess in order to preserve confidentiality.
- Unique key – The key should be used in a single context. Key re-use carries the security risk that if its confidentiality is circumvented, the impact is higher because it “unlocks” more sensitive data.

**PROCESS** Encryption uses algorithms and a key to scramble and then store or transmit information. When the data needs to be recovered from the scrambled information, the latter is decoded using the same or another specific key. The message contained in the encrypted data is referred to as plaintext. In its encrypted and unreadable form, it is referred to as ciphertext. There are many types of algorithms, all of which involve different ways of encrypting and then decrypting information (Privacy Guy, 2017).

Regarding the keys generation, they can be derived deterministically, using a passphrase and a key derivation function, or randomly, using a Random Number Generator (RNG) or Pseudo-Random Number Generator (PRNG) (Wikipedia, the free encyclopedia, 2017b). A PRNG is a computer algorithm that produces data that appears random under analysis. PRNGs that use system entropy to seed data generally produce better results, since this makes the initial conditions of the PRNG much more difficult for an attacker to guess. For example, user mouse movements are commonly used to generate unique seeds. Another way to generate randomness is to use information outside the system. Modern systems generate a fresh key for every session to ensure forward secrecy, thus adding another layer of security.

**BIG DATA CONSIDERATIONS** Encryption key management can be challenging in big data (ENISA – European Union Agency for Cybersecurity, 2016a). This is due to more consumer-provider relationships, as well as greater demands and diversity of infrastructures on which both the key management system and protected resources are located. As a good practice, NIST Big Data Working Group publications suggest that encryption keys should be managed by Chief Security Officers (CSO) only and that separate key pairs should be issued for customers and internal users.

Another challenge related to the use of cryptography in big data is how to protect sensitive information while maintaining scalability and performance.

### C.5. Traditional Encryption

**TRADE-OFFS** In modern cryptosystems, key length is measured in bits, and each bit of key increases the difficulty of a brute-force attack exponentially. It is important to note that in addition to adding more security, each bit slows down the cryptosystem as well. Because of this, the key length is a trade-off between performance and security. It is important to note that the adequate key length also depends on the type of cryptosystem used once there are other kinds of possible attacks.

**APPROACHES** There are the following types of encryption schemes (Smirnoff and Turner, 2019):

- Symmetric – The same key can be used to encrypt and decrypt the ciphertext. Symmetric encryption schemes are generally categorized as being one of the following:
  - Block ciphers – Take a number of bits and encrypt them as a single unit (block), padding the plaintext so that its size is a multiple of the block size. As the data is being encrypted, the system holds the data in its memory while waiting for complete blocks.
  - Stream ciphers – Data is encrypted as it streams instead of being retained in the system's memory.
- Public-key (or asymmetric) – It uses the concepts of public and private keys. The public-key can be used by anyone to encrypt data, but the private key is required to decrypt the ciphertext.
- Hybrid – It can be build using two separate cryptosystems: a public-key encryption scheme as key encapsulation scheme and a symmetric one as data encapsulation scheme.

**VARIATIONS/EXTENSIONS** Some examples of symmetric encryption algorithms include (Smirnoff and Turner, 2019):

- Rivest Cipher 4 (RC4) – Stream cipher also known as ARC4 or ARCFOUR. Despite being remarkable for its simplicity and speed in software, multiple vulnerabilities have been discovered in RC4, rendering it insecure. It is especially vulnerable when the beginning of the output keystream is not discarded, as well as when nonrandom or related keys are used.
- Salsa20 and ChaCha – They are very similar stream ciphers. Salsa20, the original cipher, was designed in 2005 and submitted later to eSTREAM by Daniel J. Bernstein. ChaCha is a modification of Salsa20 published by Bernstein in 2008. It uses a new round function that increases the performance on some architectures, as well as the diffusion.
- Data Encryption Standard (DES) – It was the first standardized (block) cipher for securing electronic communications in modern computing. It has some variations, such as the Triple Data Encryption

Standard (3DES), which applies the DES cipher algorithm three times to each data block. DES is not used anymore as it is considered too weak due to the processing power of modern computers. 3DES has also been deprecated by National Institute of Standards and Technology (NIST) in 2017.

- Advanced Encryption Standard (AES) – The most commonly used symmetric algorithm is AES, which is a block cipher and was originally known as Rijndael. This is the standard set by the U.S. NIST in 2001 for the encryption of electronic data. This standard supersedes DES, which had been in use since 1977. Under NIST, the AES cipher has a fixed block size of 128 bits, but different key lengths can be used: AES-128, AES-192, or AES-256.

An asymmetric encryption algorithm's example is Rivest-Shamir-Adleman (RSA). It is one of the first public-key cryptosystems and is widely used for secure data transmission. In RSA, the asymmetry between private and public-key is based on the practical difficulty of the factoring problem: the factorization of the product of two large prime numbers. RSA is a relatively slow algorithm, so it is not commonly used to directly encrypt user data. More often, RSA is used to encrypt shared keys for symmetric cryptography, which in turn can perform bulk encryption/decryption operations at a much higher speed (hybrid scheme).

Another approach for hybrid schemes is to use the Diffie-Hellman (DH) algorithm as a key agreement protocol. The DH key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel. In order to decipher the shared key, one of the private keys of the involved parties must be known or the DH problem solved. The DH problem is currently considered difficult for proper parameters. The shared key can then be used to encrypt subsequent communications using a symmetric cipher.

COMPARISONS The presented approaches differ in the following aspects (Smirnov and Turner, 2019):

- While symmetric encryption is an older method of encryption, it is faster and more efficient than asymmetric encryption, which produces bigger ciphertexts and requires heavy CPU usage.
- With symmetric encryption, there are some concerns related to secure and scalable key management, as well as the possibility to perform certain functionalities without disclosing the secret key. These concerns disappear with the public-key schemes.
- Due to the better performance and faster speed of symmetric encryption, symmetric cryptography is typically used for bulk encryption. The public-key schemes are mainly used in hybrid schemes for distributing secret keys, as well as in digital signatures.

### C.5. Traditional Encryption

- Hybrid schemes are predominant because they combine the advantages of public-key encryption in scalability and key management with the speed and space advantages of symmetric encryption.

**PRIVACY BREACH** A data breach occurs if an attacker can figure out the plaintext that originated a certain ciphertext. One way to achieve this is to obtain the symmetric or private key that allows the ciphertext decryption.

Symmetric algorithms usually have a strictly defined security claim. It is typically equal to the key size of the cipher, which is equivalent to the complexity of a brute-force attack.

Regarding asymmetric algorithms (public-key cryptography), most of their designs rely on neat mathematical problems that are efficient to compute in one direction but inefficient to reverse by the attacker. However, attacks against current public-key systems are always faster than a brute-force search of the keyspace. Their security level isn't set at design time but represents a computational hardness assumption, which is adjusted to match the best currently known attack.

There are two security models commonly used to prove the security of encryption algorithms: Indistinguishability under Chosen-Plaintext Attack (IND-CPA) and Indistinguishability under Chosen-Ciphertext Attack (IND-CCA). They are modeled using a hypothetical game between a challenger that holds the secret key information and the attacker who wants to break the encryption algorithm. The idea is to mathematically show that an attacker has no chance at winning such a game, unless he can also break some underlying hard mathematical problem. In both models, the attacker may choose 2 plaintexts. As a challenge, one of them will be encrypted. The attacker's job is to figure out (distinguish) which one was encrypted.

In IND-CPA, the attacker can obtain the encryption of any message of his choosing. This model implies some sort of randomized ciphertext. Otherwise, the attacker would just encrypt the two plaintexts of his choice by himself, then compare those against the challenge he receives.

In IND-CCA, the attacker can obtain the decryption of any ciphertext of his choosing, except the challenge one. It models the case where tricking an enemy into decrypting a lot of ciphertexts will not help to break any others. One consequence of IND-CCA security is that the ciphertext should be tamperproof. Otherwise, the attacker can slightly alter the challenge ciphertext in a predictable way, to try getting the decryption of this altered challenge.

**GDPR** Data protection has become a very important question for many companies dealing with massive amounts of sensitive data. The first reason for that is the development of dedicated legislation, such as the



Health Insurance Portability and Accountability Act (HIPAA) and the GDPR, obliging companies to secure their data, in particular personal data.

Accordingly to the GDPR (EU GDPR Portal, 2018), for example, “Companies can reduce the probability of a data breach and thus reduce the risk of fines in the future, if they chose to use encryption of personal data. [...] Encryption is the best way to protect data during transfer and one way to secure stored personal data. It also reduces the risk of abuse within a company, as access is limited only to authorised people with the right key.”

So, these laws intend to prevent any disclosure or access, either accidental, unlawful, or unauthorized, to sensitive data. In most cases, this implies the use of encryption for data at rest to ensure their confidentiality, but also the use of cryptographic techniques, such as user authentication and data at rest integrity verification, to prevent (undetected) data tampering. Also, with some of these regulations, outsourcing the storage of financial data or the processing of healthcare data to the cloud without relying on cryptographic access control mechanisms is simply illegal (Bost, 2018).

New regulations are not the only origin of the development of encrypted databases. Many internet users, both companies and individuals, got convinced of the necessity of encrypting their communications and their data to protect their privacy and/or their business. In particular, service providers are no longer trusted, as they can get subpoenaed to reveal encryption keys.

In this setting, only protecting the communications between the user and the server, using Transport Layer Security (TLS) to establish a secure channel, for example, is not enough as the server will still see the client’s data in plaintext. Indeed, modern instant messaging services use end-to-end encrypted protocols to ensure that only the communicating parties can read the messages they exchange. For databases, a system preventing the server to infer information about the outsourced dataset and/or the accesses of the client is similarly needed.

It is very important that the security of the scheme can be proven using standard cryptographic assumptions, as its users must be confident in its claimed security. In addition to being secure, this system needs to be usable in practice. This means that it has to be easy to use (i.e., easy to deploy and support a set of features similar to its insecure counterpart), that its performance must be competitive with an unencrypted database, and that it has to be scalable and support large datasets. Otherwise, the secure system will never be chosen in preference to non-encrypted systems: Signal and WhatsApp are far more successful than Pretty Good Privacy (PGP) because these mobile applications are trivial to use, the opposite of PGP-encrypted emails.

### C.5. Traditional Encryption

However, traditional encryption fails regarding usability in some scenarios, since it does not allow, for example, to perform queries or other operations over the encrypted data without decrypting it.

#### APPLICATION EXAMPLE

##### HTTPS

HyperText Transfer Protocol Secure (HTTPS) is an extension of the HTTP. It is used for secure communication over a computer network and is widely used on the Internet. In HTTPS, the communication protocol is encrypted using TLS or, formerly, Secure Sockets Layer (SSL).

The main motivation for HTTPS is to ensure authentication, confidentiality, and integrity while the data is in transit. The bidirectional encryption of communications between a client and a server protects them against communication's eavesdropping and tampering.

According to the 2019 Internet Trends report, online communications are becoming more secure and individual users are making decisions that helps them to be more secure. At the start of 2019, 87% of web traffic was encrypted, compared to just 53% in 2016.

*Font:* <https://en.wikipedia.org/wiki/HTTPS> AND  
<https://duo.com/decipher/encryption-privacy-in-the-internet-trends-report>

## C.6 ATTRIBUTE-BASED ENCRYPTION

PROPERTIES Some possible ABE properties are presented next (Liu et al., 2018; Sethi et al., 2019; Sethi et al., 2020):

- Outsourcing computing – It aims to move the heavy computation workloads (commonly the decryption operation) to powerful computation resources. As a result, the computational cost imposed on resource-constrained devices can be effectively reduced.
- Policy updating – The data owner might require to modify the access policy of the ciphertexts based on various requirements. Thus, an efficient policy updating algorithm with lower computation and communication costs is often required.
- User revocability – When there is the need to remove a user from the whole system, it must be assured that the revoked user cannot reuse his/her outdated secret key to access or compromise any data.
- Attribute revocability – When an attribute is revoked from a user, it should be guaranteed that the user loses the associated privileges and cannot reuse his/her outdated secret key to access or compromise currently unauthorized data.
- Unrestricted Attributes – In some ABE schemes, every current and future attribute must be enumerated at system initialization. On other schemes, however, attributes can be represented by any string and can be used an unlimited number of times in a policy.
- Traceability – In ABE schemes, different users may share the same decryption privileges if they are associated with the same set of attributes. This may potentially lead to the exposure of decryption keys by some malicious users. Therefore, it is important to enforce traceability in ABE schemes to find such malicious users. Traceability can be divided into two main types:
  - White-box – Identification of malicious users who are involved in exposing decryption keys.
  - Black-box – The tracing algorithm treats the decryption black-box as an oracle and finds out the identity of the malicious users whose decryption keys are utilized to construct decryption equipment.
- Collusion resistance – If the users are unable to decrypt the ciphertext individually with their own keys, then they should not be able to decrypt it even if they combine their key components. In other words, users cannot combine their attributes to decipher the encrypted data.

## C.6. Attribute-Based Encryption

**PROCESS** Attribute-Based Encryption (ABE) is a kind of public-key encryption that applies user attributes as a public key. User identification is as a specific attribute, so ABE can implicitly include Identity-Based Encryption (IBE) as well.

This encryption algorithm includes the involvement of 3 entities: the data owner, the user that receives the data, and a trusted third party, called Key Generation Center (KGC). The role of the KGC is to generate and distribute the users' secret keys, which depend on their certified representative attribute sets. The data owner encrypts the data with the KGC's public key and some attributes. Any user that receives the data and possesses a certified attribute set satisfying the defined ciphertext's attributes can decrypt the data using its own private key, without any additional interactions with the data owner (Hoang et al., 2019).

**BIG DATA CONSIDERATIONS** ABE is a promising technique to ensure end-to-end security of big data, e.g., in the cloud. However, the policy updating has always been a challenging issue when ABE is used to construct access control schemes. A trivial implementation is to let data owners retrieve the data and re-encrypt it under the new access policy, and then send it back to the cloud. This method, however, incurs a high communication overhead and heavy computation burden on data owners. There are already some scheme proposals that enable efficient access control with dynamic policy updating for big data in the cloud (Yang et al., 2015).

Also, most ABE schemes do not consider privacy protection issues during the key generation phase. So, the KGC knows both attributes and corresponding keys of each user. This problem is especially serious in big data scenarios because it can cause great damage to a lot of individuals or the business secrets of industrial enterprises. There are already schemes that protect user's privacy during key issuing. One way to do so is to separate the functionalities of attribute auditing and key generation (Song et al., 2019).

**TRADE-OFFS** In Attribute-Based Encryption, all trade-offs depend on the chosen scheme.

A commonly considered trade-off is related to indirect revocation, which is presented in more detail below. In indirect revocation, the KGC periodically runs a key update algorithm and distributes new private keys to unrevoked users (Hoang et al., 2019). Despite its simplicity in design, this approach encounters a tricky trade-off between security and system performance. Indeed, if the time interval is set too large, the revoked users are still able to use their private keys to decrypt data until a predetermined key expiration date. On the other hand, if the time interval is too small, it will cause a heavy key distribution workload to the KGC.

There are also other schemes in which trade-offs between the key size and the ciphertext size, or the encryption and decryption time can be established (Attrapadung et al., 2016).

APPROACHES With regards to the KGC trusted entity, the following approaches can be considered (Zhang et al., 2018):

- Single authority – The trusted authority holds each user’s attribute keys and has the power to decrypt every ciphertext (key escrow problem).
- Multi-authority – Many different authorities operate simultaneously, each handing out secret keys for a different set of attributes so that none of the authorities can figure out the encrypted message by itself.

Regarding user or attribute revocability, two approaches can be considered (Hoang et al., 2019):

- Indirect revocation – It requires the KGC to periodically distribute a key update to all unrevoked users. Even if a user is revoked, its private key is still valid until the end of the current time interval.
- Direct revocation – It allows to revoke users as soon as necessary. Direct revocation fits into high-security requirement systems.

Once a user is revoked, he should be no longer capable of decrypting data shared in the past. To achieve forward secrecy, the following approaches are possible:

- Unrevoked users re-encrypt all the shared data, which is not computationally efficient.
- The data is re-encrypted by the storage service using its own secret key. The service takes full control of the revocation, which removes the original user-centric characteristic of ABE.
- Delegate to a semi-trusted proxy the burden of blindly translate a ciphertext associated with certain attributes to one related to other attributes, using a re-encryption key. This key is computed by the data owner and sent to the proxy. Hence, proxy re-encryption techniques can be used to achieve forward secrecy and keep the scheme user-centric.

VARIATIONS/EXTENSIONS ABE can be divided into two main variants (Pournaghi et al., 2020):

- Ciphertext-Policy Attribute-Based Encryption (CP-ABE) – The user’s private key is dependent on arbitrary attributes, and the encoder encrypts a message adopting a specific access policy. A user in this method can decrypt a ciphertext if and only if the attributes of the user meet the policy determined by the ciphertext. Therefore, the ciphertext in CP-ABE is associated with the access structure to control which user can decrypt it. A representation of how CP-ABE encryption and decryption operations work is presented in Figure 38.

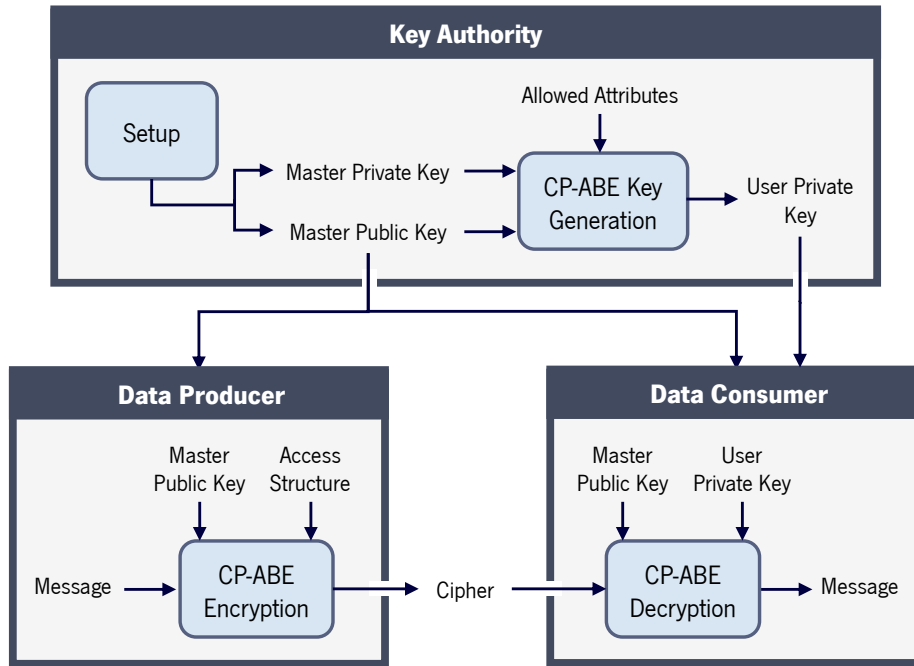


Figure 38: CP-ABE encryption and decryption representation.

- Key-Policy Attribute-Based Encryption (KP-ABE) – The encryption depends on a set of attributes and the user private key is dependent on an access structure. In this method, the user decrypts the ciphertext only when the ciphertext’s attribute set satisfies the user’s access structure. Hence, the user’s private key is associated with the access structure to control what ciphertexts the user can decrypt. A representation of how KP-ABE encryption and decryption operations work is presented in Figure 39.

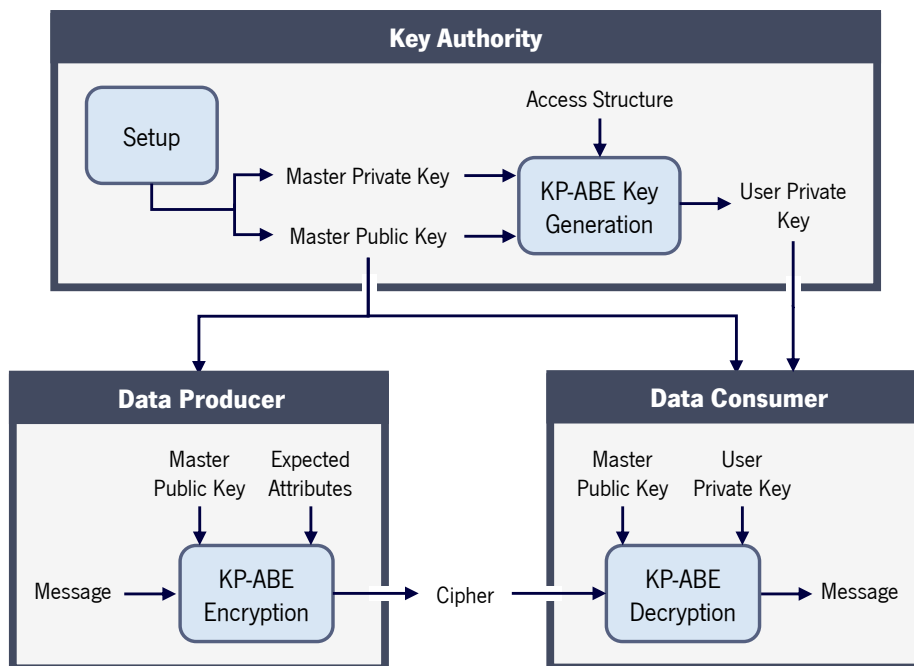


Figure 39: KP-ABE encryption and decryption representation.

## COMPARISONS

- In KP-ABE, the ciphertext is labeled and if the user key access structure matches the labels then he can decrypt the ciphertext. But in CP-ABE, the user's private key is labeled and if that set of labels satisfies the ciphertext access structure, then the user can access the ciphertext. In short, in KP-ABE the KGC determines what messages a user can access, while in CP-ABE the data owner indicates which users can read the ciphertext (Pournaghi et al., 2020).
- In a single authority scheme, all system attributes are managed by that single authority. So, failure or corruption of the authority affects the whole system. Another drawback of a single authority scheme is the mentioned key escrow problem. Multi-authority ABE schemes are proposed to overcome these drawbacks (Rajkumar et al., 2014). The proposed multi-authority CP-ABE schemes are generally more expressive than the multi-authority KP-ABE schemes, but they also entail higher implementation complexity.
- Direct revocation requires unrevoked users to locally keep a long and up-to-date list of revoked users, which will be embedded in the ciphertexts. Indirect revocation requires the key authority to release a key update material periodically so that only unrevoked users can update their keys. Hence, revoked users' keys are implicitly rendered useless. An advantage of the indirect method over the direct one is that it does not require senders to know the revocation list. Also, the direct revocation generally makes the computational costs of encryption and decryption algorithms increase linearly with the number of revoked users in the list (Hoang et al., 2019). As the system grows rapidly over time, users may incur a huge computational overhead. In contrast, an advantage of the direct method over the indirect one is that it does not involve a key update phase for all unrevoked users, thus not requiring to interact with the key authority (Rajkumar et al., 2014).

**PRIVACY BREACH** A data breach occurs if an attacker or an unauthorized user can figure out the plaintext that originated a certain ciphertext. Most ABE schemes in academic literature only satisfy IND-CPA. However, there are already some that satisfy IND-CCA security (Zeutro, LLC, 2020; Anada and Arita, 2017).

**GDPR** Attribute-Based Encryption facilitates personal data protection in a connected world. Because ABE enforces access control at a cryptographic (mathematical) level, it provides better security assurance than software-based solutions. So, ABE has been identified by European Telecommunication Standards Institute (ETSI) as a key enabler technology for access control in highly distributed systems, such as 5G and IoT.

In 2018, the ETSI Technical Committee on Cybersecurity released two specifications on ABE that describe how to protect personal data with fine-grained access controls (Antipolis, 2018). Both specifications enable compliance with the GDPR by allowing the secure exchange of personal data among data controllers and data processors. A standard using ABE has several advantages for the industry. It provides an efficient and secure-by-default access control mechanism for data protection that avoids binding access to a person's name, but instead to pseudonymous or anonymous attributes.

#### APPLICATION EXAMPLE

##### MQTT Security

In recent years, Internet of Things (IoT) has developed rapidly and has been widely used in industry, agriculture, e-health, smart cities, and families. As the total amount of data transmission will increase dramatically, security will become a very important issue in data communication in IoT. There are many communication protocols for Device to Device (D2D) or Machine to Machine (M2M) in IoT. One of them is Message Queuing Telemetry Transport (MQTT), which is quite prevalent and easy to use.

MQTT is designed for resource-constrained devices, so its security is not as strong as other communication protocols. To enhance MQTT security, it needs an additional function to overcome its weakness. However, considering the limited computational abilities of resource-constrained devices, they cannot use too powerful or complex cryptographic algorithms. Attribute-Based Encryption can be applied in such a scenario.

*Font: Liao et al. (2019)*

##### Electronic Health Records Security

With the digitization of traditional medical records, medical institutions encounter difficult problems, such as electronic health record storage and sharing. Patients and doctors spend considerable time querying the required data when accessing electronic health records, but the obtained data are not necessarily correct, and access is sometimes restricted. A medical data sharing scheme can be designed using ABE, for example, to ensure data confidentiality and access control of medical data.

*Font: Niu et al. (2020)*



## C.7 SEARCHABLE ENCRYPTION

PROPERTIES Some SE schemes' properties are specified below (Varri et al., 2019; Al-Maytami et al., 2020; Li et al., 2019b; Elizabeth and Prakash, 2019):

- Search functionality – SE schemes allow users to search over encrypted data using keywords or ranges, for example, without compromise the data subject's privacy.
- Privacy-preserving – The scheme should not disclose information such as the stored data or the contents of the query trapdoors to the server or unauthorized parties.
- Dynamic updates – Some SE schemes have provisions for dynamic updates. This means that index-based schemes, for example, should allow both addition and deletion of documents without reconstructing the index.
- Forward privacy – It can be regarded as forward update privacy, once it requires that the update (addition and deletion) operations cannot be linked to previous search queries. This means that the server should not learn whether newly updated data match a previously searched query or not. There are also more stringent security notions that can be considered. For example, if an SE scheme has forward update privacy and its search operation over newly updated data does not leak the past query information, it achieves a new security notion called forward search privacy.
- Backward privacy – An SE scheme satisfies backward privacy if after deleting some data matching a certain query, the server cannot reveal the deleted data entry from the subsequent searches of that query.
- Verifiability – Some schemes support the verifiability of the integrity or completeness of the results returned from the server.
- Access control – Access control prevents unauthorized data access. Some SE schemes allow the user to share specific data with a group of selected users.

PROCESS Generally, SE schemes comprise three types of entities: data owner, data user, and a server (Wang et al., 2020). The basic idea is that the data owner encrypts the data before uploading it to the server. Then, the data user sends a query trapdoor to the server, which could be generated by the data user, the data owner, or a trusted authority, depending on the applied scheme. Using the query trapdoor, the server can perform the search over the encrypted data and return the result to the data user, which then can decrypt it.

**BIG DATA CONSIDERATIONS** Most of the traditional SE schemes do not capture the relevance of the retrieved data, which brings a few drawbacks when SE is applied to big data (Elizabeth and Prakash, 2019). First, without foreknowledge of the encrypted data stored at the server, the authorized data users have to go through all data matching some restricted search queries to find what meets their interest. Second, downloading all that data may incur unnecessary traffic and cost.

**TRADE-OFFS** Although in principle the concepts of search and encryption might seem contradictory, SE methods allow doing this kind of search over encrypted data, achieving different trade-offs between privacy, efficiency, and query expressiveness (ENISA – European Union Agency for Cybersecurity, 2015).

**APPROACHES** There are various approaches to search over encrypted data, including Searchable Encryption, Fully Homomorphic Encryption (FHE), Oblivious RAM (ORAM), Functional Encryption, and Property-Preserving Encryption (PPE). All these approaches achieve different trade-offs between security, query expressiveness, and efficiency (Kumar and Thilagam, 2019).

The SE approaches can be classified into two categories based on their ability to guarantee precision (Kumar and Thilagam, 2019):

- Deterministic encryption – The same plaintext is always mapped to the same ciphertext. These schemes guarantee high precision and are efficient, but there is a lot of scope for the possibility of security attacks.
- Non-deterministic or probabilistic encryption – The same plaintext is encrypted to different ciphertexts each time, using the same key. These schemes are secured, but they are not as efficient nor guarantee as high precision as deterministic encryption schemes.

In the conventional SE schemes, the main searching functions are performed using one search query or multiple search queries (Kamal and Iwamura, 2019). The search method using one search query can be further divided into the total matching search, partial matching search, range search, and similarity search. The search method using multiple search queries can be further divided into logical conjunctive search (*AND* operator), logical disjunctive search (*OR* operation), and their combination.

Other cryptographic approaches can be combined with SE to prevent both the search pattern and access pattern leakage information, such as:

- Private Information Retrieval (PIR) – For each query, a set of random queries are also sent to the server. Similarly, a set of random documents are stored at the server along with the actual documents. Due to the random queries and random documents, the server does not come to know which query is real and which original documents are returned to the users for their queries.

- Blind storage – The encrypted documents are stored in blocks of memory so that the server only knows which blocks are uploaded and retrieved. It does not come to know which blocks constitute the same file.
- Oblivious RAM (ORAM) – The documents are kept re-encrypted and the locations of the stored documents in memory keep changing. So, the server does not come to know which documents the user is frequently accessing.

For the application of Searchable Encryption in practice, it can be divided into four application models (Feng and He, 2018):

- Single User Model – The owner of the key generates both the searchable ciphertext and the trapdoor for the query. Generally speaking, the key can only be owned by one user, who is the data owner and the data user.
- Single Data Owner/Multiple Query Users Model – The data is encrypted and uploaded to the server by a single data owner. Multiple data users send query trapdoors to the server to retrieve the encrypted data. The trapdoors generation and ciphertexts decryption can be performed by the data users, the data owner, or a trusted authority, depending on the applied scheme.
- Multiple Data Owners/Single Query User Model – The searchable ciphertext is generated by multiple users, usually resorting to a public key, and uploaded to the server. The data user then resorts to the private key to generate a query trapdoor that allows retrieving the ciphertext and decrypts it.
- Multi-Users Model – Multiple users are the data owners and data users. There are different schemes under this model, generally involving a trusted third party, such as a group manager or a key management server.

VARIATIONS/EXTENSIONS Different types of SE schemes can be considered (Varri et al., 2019; Lu et al., 2019; Xu et al., 2019):

- Symmetric Searchable Encryption (SSE) – Symmetric Searchable Encryption (SSE) schemes deal only with one private key and allow the user who holds it to generate ciphertexts and query trapdoors. In particular, a data structure that supports fast search on the dataset is built, such as an index, and is encrypted using the SSE scheme. The dataset is also encrypted using a standard symmetric encryption scheme, such as AES. So, to query the data, it is only necessary to query the encrypted data structure. These schemes are appropriate when the entity that searches over the data is also the one that generates it.

### C.7. Searchable encryption

- Asymmetric/Public Searchable Encryption (PSE) – It deals with a public key and a private key, allowing multiple data owners to use the public key of the data user to encrypt the data, which later can be searched and decrypted by the same data user's private key. It is appropriate in any setting where the party searching over the data is different from the one that generates it. There are many PSE schemes and variants, each one adequate for different application scenarios.
- Attribute-Based Searchable Encryption (ABSE) – In ABSE, a trusted third-party authority generates each user's private key accordingly to his/her attributes, such as name, department, or gender. The data owners encrypt the documents under defined access policies to control user access. A data user is only able to decrypt the documents when their attributes match the access policy.

COMPARISONS The existent SE schemes are different from each other in terms of security, efficiency, and precision (Kumar and Thilagam, 2019). Hence, they should be selectively chosen while encrypting data as per the precision and security requirements of data owners. If the concern of data owners is more about precision, they can use deterministic encryption schemes. If their concern is more about privacy and security, they can use non-deterministic encryption schemes.

Regarding the SE variants, SSE schemes are computationally efficient, but they only allow the secret key of the data owners to perform the search operation (Kumar and Thilagam, 2019). So, these schemes must deal with key management and distribution problems.

On the contrary, PSE schemes enable the flexibility of sharing encrypted data with multiple users, allowing for advanced query expressiveness. However, PSE schemes are computationally expensive and the security of most of them relies on classic assumptions, which have been proven to be unable to withstand the attack of quantum computers (Xu et al., 2019).

ABSE is better suited for scenarios where each user owns a set of attributes and the data owner decides on an attribute-based access policy that determines who, among the users of the system, is eligible to search and decrypt the ciphertext (Khader, 2014).

PRIVACY BREACH For some SE schemes to meet certain precision requirements, the data owners must provide some extra information in data structures, which may cause information leakages during the search or update operations (Kumar and Thilagam, 2019; Wu et al., 2019), such as:

- Data structure information – Data structure (e.g., index) information may be leaked from the encrypted data structure or the ciphertext.

- Search pattern – Search pattern information allows the server to derive whether two search queries include the same content, for example.
- Access pattern – Access pattern information can be derived by the server from the search results.

So, a secure SE scheme must keep all this information private (Feng and He, 2018). In some scenarios, user anonymity (identity privacy) can also be a requirement.

Several security attacks exploit these leakages to infer the plaintext information from the trapdoors, data structures, or the actual content of the outsourced encrypted data. If these leakages are not prevented, they undermine the purpose of SE. The most common attacks that exploit these leakages are (Kumar and Thilagam, 2019):

- Inference Attacks – The objective of this type of attack is to derive the plaintext keyword, for example, of the encrypted data structures, such as indexes. Frequency analysis attacks, sorting attacks, and counting attacks come under the category of inference attacks.
- Scale Analysis Attack – This attack exploits the leakage of the search pattern, which is used to infer the plaintext information of the trapdoors. Search pattern conveys whether the earlier issued trapdoors and current trapdoor are generated from the same keyword set or not, for example.
- IKK attack (Islam et al., 2012) – This attack is based on the leakage of the access pattern, which is exploited to infer the plaintext information of the trapdoor. The access pattern conveys whether the returned data for the given two or more trapdoors are the same or not. If they are the same, the server assumes that the data user is accessing those documents frequently, so he/she issues the same trapdoor.
- Keyword Guessing Attack (KGA) – This attack aims at keyword-based SE schemes and is more prone to occur in PSE. In SSE, inferring plaintext information from the encrypted data is possible only through brute-force attacks, which are computationally expensive to carry out. In PSE, the data owners use the public-key-based SE schemes to encrypt the data structures and the documents. These attacks are based on the principle of guessing some candidate keywords and encrypting those keywords with the same public-key approach. It is known that the number of commonly used keywords is not so big. These guessed keywords can then be verified in an offline manner by comparing them with the actual data owners' encrypted keywords. If these guesses are matched, the adversaries can confirm themselves that the guessed keywords are present in the data owners' encrypted information. So, KGAs allow the adversaries to infer plaintext keywords of trapdoors and data structures. KGAs can be performed by the server that stores the data or by outsiders.

The proposed SE schemes must prove that they can preserve the privacy of the user's data and prevent information leakage. There are different models that these proposals can take into account to prove their security (Varri et al., 2019; Wu et al., 2019), such as Indistinguishability under Chosen-Plaintext Attack (IND-CPA) or Indistinguishability under Chosen Keyword Attack (IND-CKA).

Additionally, some schemes ensure the security properties forward privacy and backward privacy. The former means that if a certain query was previously searched, the server cannot know whether a new file matches that same query. The latter means that queries cannot be executed over deleted files.

GDPR In Annex C.5, traditional encryption advantages regarding GDPR and disadvantages concerning usability were presented. Indeed, achieving both security and efficiency for simple functionalities is delicate. For example, some encrypted databases may need to support search operations, which is not efficiently possible using traditional encryption. Searchable Encryption schemes can be used for this purpose.

#### APPLICATION EXAMPLE

##### IloT Security

Internet of Things (IoT) deployment is composed of various types of sensors, actuators, and other intelligent terminal equipment connected to the Internet. It provides identification, computation, and mutual information exchange among the connected devices. As a special type of IoT, the Industrial Internet of Things (IIoT) has been increasingly prevalent. It relies on various kinds of mobile intelligent terminal devices and wireless communication technologies to realize remote monitoring and management in modern industrial sectors. With the sharp increase of the massive industrial data, cloud computing technologies have been integrated into IIoT to store and process this information. Thus, apart from upgrading to intelligent industries, the cloud-assisted IIoT also brings a vast improvement in industrial manufacturing efficiency and lowers the production cost.

Despite the great benefits regarding managing industrial data brought by cloud-assisted IIoT, security and privacy concerns in data outsourcing have been raised, of which data confidentiality is one of the most important aspects. From the perspective of cloud users, the contents of the outsourced industrial critical data are very sensitive and should be kept confidential for privacy preservation. Therefore, sensitive industrial data should be encrypted before being outsourced to the cloud.

Along with data confidentiality, data sharing is also indispensable. Thus, Public Searchable Encryption is a good candidate in cloud-assisted IIoT to achieve data retrieval without leaking privacy.

*Font: Zhang et al. (2019a)*

## C.8 HOMOMORPHIC ENCRYPTION

### PROPERTIES

- Homomorphism – HE schemes allow computations to be performed directly on the ciphertext without decrypting it (Mert et al., 2020). An encryption function  $E$  is homomorphic if  $E(a) \oplus E(b) = E(a \otimes b)$ , for some operators  $\oplus$  and  $\otimes$ . There are different types of homomorphisms (Elmahdi et al., 2020; Li et al., 2019a), such as:
  - Additive Homomorphism – Performing decryption on some operation of two ciphertexts is the same as the addition of the two plaintexts:  $E(a + b) = E(a) \oplus E(b)$ .
  - Multiplicative Homomorphism – Performing decryption on some operation of two ciphertexts is the same as the multiplication of the two plaintexts:  $E(a \times b) = E(a) \oplus E(b)$ .
  - Mixed Multiplicative Homomorphism – Performing decryption on some operation of one ciphertext and one plaintext is the same as the multiplication of the two plaintexts:  $E(a \times b) = E(a) \oplus b$ .
- Privacy-preserving – HE schemes allow conducting operations over encrypted data to protect the privacy of the underlying data (Ren et al., 2021).
- Verifiability – Usually, Homomorphic Encryption provides no verifiability that the computation performed over the encrypted data by a third-party server, for example, is correct. However, there are already some schemes that take this property into account, enabling verifiable computation on outsourced encrypted data (Lai et al., 2014; El-Yahyaoui and Kettani, 2019).

**PROCESS** Generally, HE schemes comprise three types of entities: the data owner, the data user, and the server. First, the data owner encrypts the data before uploading it to the server. Then, the server performs some computation directly on the encrypted data without requiring access to a secret key (Homomorphic Encryption Standardization, 2017). HE includes multiple types of encryption schemes that can perform different classes of computations over encrypted data. The result of such a computation remains in its encrypted form and can be revealed at a later point by the data user, who owns the secret key. In some schemes, the data owner is also the data user.

**BIG DATA CONSIDERATIONS** With the emergence of big data and the continued growth in cloud computing applications, serious security and privacy concerns emerged. Considering security, data are generally encrypted before being stored in a cloud database (Shinde and Al-Rummana, 2018). Thus, it is quite

difficult for unauthorized users to understand the encrypted data and get the corresponding plaintext back. However, the value of data is also crucial for big data environments. It refers to statistical inferences, events within the data sets, correlations among attributes, and hypotheses used in the analysis. Encrypting data in a traditional way makes it impossible to leverage its value because the data needs to perform the decryption first whenever it is to be processed.

Consequently, several researchers and cybersecurity experts have embarked on a quest to extend data encryption to big data systems and cloud computing applications (Alloghani et al., 2019). HE emerged as a possible solution where the client's data is encrypted on the cloud in a way that allows some search and manipulation operations without proper decryption. It could be a solution to the growing demands of big data and the absence of security and privacy mechanisms therein. It has the potential to ensure security and privacy, preserving the CIA goals in the context of big data and cloud computing.

**TRADE-OFFS** HE intends to eliminate the common tradeoff between data usability and data privacy. In FHE, all features may be used in an analysis, without compromising privacy. However, the flexibility, performance, and computational workload of each scheme vary (Drozdowski et al., 2019). There is still the need for better resources or guides to explain the tradeoffs between the different schemes.

**APPROACHES** There are two different approaches regarding coping with the noise in SHE or FHE ciphertexts, named noisy and noise-free schemes (Mainardi et al., 2019). Both these approaches were previously introduced in the HE disadvantages. Noisy schemes frequently require the usage of noise management techniques, such as bootstrapping, modulus switching, and scale-invariant schemes. In noise-free schemes, the ciphertext has no noise, thus an unbounded number of homomorphic operations can be performed without any of those costly techniques being involved. Nevertheless, while common noisy SHE/FHE schemes are based on well-known and scrutinized mathematical problems, noise-free schemes usually rely on less common algebraic trapdoors.

Regarding the encryption keys, HE generally requires the ciphertexts used during processing to be encrypted with the same key. However, some enhanced HE systems, named Multi-key Homomorphic Encryption, are capable of operating on inputs encrypted under multiple unrelated keys. A ciphertext resulting from a multi-key homomorphic evaluation can be jointly decrypted using the secret keys of all the users involved in the computation (López-Alt et al., 2017).

**VARIATIONS/EXTENSIONS** Different HE types allow for different levels of computations (Ma et al., 2018; Acar et al., 2018):



- Partially Homomorphic Encryption (PHE) – PHE allows for one type of operation that works an unlimited amount of times. It supports homomorphic additions or multiplications, but not both. For example, the Paillier, Goldwasser-Micali, Benaloh, Okamoto-Uchiyama, and Kawachi cryptosystems are only homomorphic over addition, while RSA and El-Gamal are homomorphic over multiplication.
- Somewhat Homomorphic Encryption (SHE) – SHE allows for several types of operations that work a limited amount of times. It supports a limited number of both additions and multiplications on encrypted data. Some of the major SHE schemes are:
  - Boneh-Goh-Nissim scheme – Allows for an unlimited number of additions but only one multiplication.
  - Polly Cracker scheme – The size of the encrypted message grows exponentially with the homomorphic operations.
  - Sander, Young, and Yung scheme – The size of the encrypted message grows by a constant with each homomorphic operation.

When the size of the encrypted message grows, it becomes more computationally expensive up to a point where it is no longer realistic to compute. These SHE schemes are a step on the way to figure out FHE schemes.

- Fully Homomorphic Encryption (FHE) – FHE allows for unlimited types of operations that work an unlimited amount of times. It supports to compute any polynomial function on the data, meaning an unlimited number of additions and multiplications. Several different approaches have been used to try to create a fully homomorphic encryption scheme, including lattice-based, bootstrapping, and LWE, all of which come with downsides mostly relating to their computationally expensive nature. These processes make use of linear algebra, number theory, and abstract algebra along with additional mathematical concepts. FHE is at the point of research since it is still inefficient in practice.

**PRIVACY BREACH** While the homomorphic capabilities of a cryptosystem do not weaken the security guarantees per se, they may increase the adversarial power if combined with other vulnerabilities (Mainardi et al., 2019). In fact, despite most HE schemes meet the IND-CPA security, all the ones that currently can be used in practice don't meet the IND-CCA security. There is some research to address this problem in the theoretical world, but the performance is completely impractical. Therefore, all current solutions using HE cannot guarantee security in the IND-CCA scenario (Peng, 2019).

GDPR In Annex C.5, traditional encryption advantages regarding GDPR and disadvantages concerning usability were presented. Indeed, achieving both security and efficiency for simple functionalities is delicate. Some servers may need, for example, to perform operations over the encrypted data without compromise the data subject privacy, which is not possible in an efficient or practical way using traditional encryption. Homomorphic Encryption schemes can be used for this purpose.

#### APPLICATION EXAMPLE

##### Privitar-NHS De-identification project

The National Health Service (NHS) holds a wealth of patient-level data that it makes available to recipients with legitimate permissions, in de-identified form. Currently, the NHS has some tools which are used to de-identify the data. The De-identification project was set up to replace these disparate tools with a single NHS-wide solution. This means that when the right legal basis, controls, and safeguards are in place, data can be linked across different care settings and geographic boundaries. The aim is to help to improve health and care services through research and planning, and ultimately lead to better individual care.

The De-identification methodology applies a variety of actions to deliver de-identification:

- Redaction – An identifier, such as name, is deleted.
- Derivation – The granularity of a data item is reduced (e.g., date of birth to the year of birth). This is an irreversible activity.
- Tokenization – Consistently obscuring the data item (e.g., NHS number), by replacement with a token. This is a reversible activity.

De-identification of data is a balance between risk and utility. As the granularity of the information in a dataset is reduced, the risk of unauthorized re-identification is reduced, but so is the utility of that dataset.

One of the major benefits of using a single de-identification tool across the NHS is the ability to link data. However, for security reasons, data is de-identified in different pseudonymization domains for each different part of an organization. Within one domain, all data with the same base value will be replaced with the same token. Across domains, the same base value will receive different tokens.

Making data available and linkable for specific recipients may require transforming data between domains. When doing this using standard encryption or tokenization techniques there is a requirement to remove the encryption for the first domain and replace it with the second domain encryption. This reveals the base value – in this case, the NHS number, which is an identifiable attribute – an action that cannot be allowed. Using consistent tokenization and PHE by Privitar Publisher, it is possible to transform data items between any two domains without revealing the base value, even if they have been de-identified by two instances of the de-identification service using different encryption keys.

This methodology allows the De-identification toolset to be deployed to multiple locations across the NHS and to make data de-identified by any of these tools potentially linkable between them. This gives the greatest potential utility to the data held by the NHS.

*Font: The Royal Society (2019)*

## C.9 SECURE MULTI-PARTY COMPUTATION

**PROPERTIES** Some secure MPC properties are presented next (Volgushev et al., 2019; Evans et al., 2018):

- **Privacy-preserving** – MPC guarantees the privacy of the computation’s input and intermediate data. Specifically, no information about the private data held by the parties can be inferred from the messages sent during the execution of the protocol. The only information that can be inferred about the private data is whatever could be inferred from the final output of the computation.
- **Correctness** – Any proper subset of adversarial colluding parties willing to share information or deviate from the instructions during the protocol execution should not be able to force honest parties to output an incorrect result. This correctness goal comes in two flavors: either the honest parties are guaranteed to compute the correct output (a “robust” protocol), or they abort if they find an error (an MPC protocol “with abort”).

**PROCESS** The computation process significantly varies among the existing cryptographic techniques that can be used to perform secure MPC. These techniques will be further discussed below.

**BIG DATA CONSIDERATIONS** Data is indispensable in today’s business and technology world. Various organizations, dealing with several types of business and services, are producing a massive amount of data every day. Analyzing this data for gaining insights has been drawn much attention from many organizations and academia since it can help in making strategic decisions.

In the current business trend, multiple organizations dealing with similar kinds of applications want to perform computation on the union of their datasets, i.e., Multi-Party Computation (Qaosar et al., 2020). This computation may involve data mining/analysis, queries over the joint datasets, classification, statistical analysis, among others. Since the business applications contain sensitive information, such as personal, financial, or health-related data, sharing such data can potentially violate individual privacy and lead to significant financial loss to the company. So, the organizations do not want to disclose their dataset to others. However, when multiple organizations wish to conduct a data mining operation jointly, they are willing to get the mining results from the union of their dataset without disclosing the sensitive data to other parties.

Secure MPC can help to achieve this goal, but the current algorithms need to be improved regarding scalability with the data size (Volgushev et al., 2019).

**TRADE-OFFS** Existing actively-secure MPC protocols require either linear rounds or linear space (Zhu et al., 2018). Due to this fundamental space-round dilemma, no existing MPC protocol can run large-scale computations without significantly sacrificing performance. In short, there is a trade-off between performance, space, and achievable security level in MPC protocols (Cohen et al., 2020).

**APPROACHES** Existing secure computation solutions can be divided into two main categories based on the number of participants they support (Bayatbabolghani and Blanton, 2018):

- **Secure Two-Party Computation** – The secure computation is carried out by two parties. Each party provides private input, both jointly evaluate the function, and one or both of them learn the output.
- **Secure Multi-Party Computation** – The secure computation is carried out by a number of computational parties. The input can come from any number of participants and the output can be delivered to a desired set of participants. This setting supports the ability to outsource the computation by one or multiple input owners to external parties who securely carry out the computations without learning private data.

The following primitives are important to build MPC protocols (ENISA – European Union Agency for Cybersecurity, 2014; Evans et al., 2018):

- **Secret Sharing** – Secret sharing refers to methods for distributing a secret amongst a group of participants, to each of whom is allocated a share of the secret. The secret can be reconstructed only when a sufficient number of shares, possibly with different types, are combined. Informally, a  $(t, n)$ -secret sharing scheme splits the secret  $s$  into  $n$  shares, such that any  $t - 1$  of the shares reveal no information about  $s$ , while any  $t$  shares allow complete reconstruction of the secret  $s$ .

Verifiable secret sharing schemes allow participants to be certain that no other players are lying about the contents of their shares, up to a reasonable probability of error. A verifiable secret sharing protocol could be used to implement any MPC protocol if a majority of the participants are honest. Furthermore, the MPC protocols thus achieved are information-theoretically secure: the privacy of the inputs is guaranteed without making any computational assumptions.

- **Oblivious Transfer** – An oblivious transfer is a protocol in which a sender transfers one of the potentially many pieces of information to a receiver, but remains oblivious as to what piece, if any, has been transferred. It has been shown that oblivious transfer is complete for MPC, which means that given an implementation of oblivious transfer, it is possible to securely evaluate any polynomial-time computable function without any additional primitive.

- Commitment – A commitment scheme allows a sender to commit to a secret value and reveal it later to a receiver. Such a scheme must ensure the following properties:
  - Hiding – The receiver should learn nothing about the committed value before it is revealed by the sender.
  - Binding – The sender should not be able to change its choice of value after committing.
- Zero-Knowledge Proof – A zero-knowledge proof allows a prover to convince a verifier that it knows  $x$  such that  $C(x) = 1$  ( $C$  is a public predicate), without revealing any further information about  $x$ .

Different cryptographic techniques can be used to perform secure computations (Volgushev et al., 2019; Bayatbabolghani and Blanton, 2018), such as:

- Homomorphic Encryption – Operations are performed over encrypted data, as described in the previous section.
- Garbled Circuit Evaluation (GCE) – In garbled circuits, one party encrypts each input bit to create a “wire label”. Then, the same party converts the computation into a circuit of binary gates, each expressed as a “garbled truth table” comprising a few ciphertexts. A single evaluator party receives the circuit and the encrypted wire labels for all input bits. At each gate, the evaluator combines the inputs to produce an encrypted output using the garbled truth table. Garbled circuits require no communication between parties during the evaluation, but their state is far larger than the input data, which makes the processing of large data impractical.
- Secret Sharing – It splits each sensitive input (i.e., each integer, rather than each bit) into “secret shares”, which in combination yield the original data. In a common encoding, secret shares cancel out into the plaintext value when added together. Each computing party works on one secret share. The additions happen locally and without communication, but multiplications require interaction before or during the evaluation. Secret sharing only multiplies state size by the number of shares, but the communication during computation limits scalability. Batching communication helps reduce overheads, but a multiplication still requires sending at least one bit between parties.

VARIATIONS/EXTENSIONS Several protocols can be implemented using MPC techniques (The Royal Society, 2019), such as:

### C.9. Secure Multi-Party Computation

- Private Information Retrieval (PIR) – Allows a user to query a database whilst hiding the identity of the data retrieved. Google, for example, employs PIR to warn a user that their password might be unsafe.
- Private Set Intersection (PSI) – Two or more parties compare datasets without revealing them in an unencrypted form. In the end, each party knows which items they have in common with the others. There are some scalable open-source implementations of PSI available.

**PRIVACY BREACH** An MPC protocol is expected to be shown secure against a formal security definition specifying the adversarial model. The two most fundamental security models correspond to modeling the computation participants as (Bayatbabolghani and Blanton, 2018):

- Semi-honest, honest-but-curious, or passive – A semi-honest participant is trusted to follow the prescribed computation, but might save and analyze intermediate results in the attempt to learn unauthorized information.
- Malicious or active – A malicious participant can arbitrarily deviate from the protocol specification in the attempt to breach security.

**GDPR** MPC gives substance to several GDPR concepts (Segers, 2020):

- Data protection by design and by default – The sole requirement for confidentiality to hold throughout the business process is proper segregation of duties between the MPC servers. MPC provides a significant level of protection since a malicious actor needs to compromise multiple MPC servers to steal or leak data.
- Processing means any operation performed on personal data – The intermediate data communicated between MPC servers are considered non-personal data. Legal experts agree that operations performed on the secret-shared data are not considered processing of personal data due to MPC's unique data protection properties.
- Purpose limitation – The parties involved in the computation have explicit control over what is calculated. If the MPC servers do not agree to some computation, it cannot be executed.
- Data minimization – The selected parties receive the result of the calculation and will not know more than this result.
- Rights of the data subject – As personal data is not exchanged, the data owner maintains control over the data.

## APPLICATION EXAMPLE

## Sharemind

Sharemind, a secure distributed database system developed by Cybernetica, uses MPC to enable organizations to perform analysis on shared data. The data from input parties is securely encrypted and distributed, remains private during computation by computing parties, and results are revealed to authorized result parties only.

The first real-world application of Sharemind was the analysis of key performance indicators for the Estonian Association of Information Technology and Telecommunications (ITL). The ITL proposed collecting certain financial metrics and analyzing them to gain insights into the state of the sector. The member companies expressed concerns over the confidentiality of the metrics, as they would be handing them out to competitors, which prompted the use of MPC. In some examples, with hundreds of thousands of data records or more, the processing can take some hours.

Sharemind has been applied to a range of cases, including to allow social studies on tax and education records, for tax-fraud detection, to predict satellite collisions in Low Earth Orbits, and to demonstrate the feasibility of genome-wide association studies with multiple data providers.

*Font: The Royal Society (2019)*