Malleable privacy-enhancing-technologies for privacy-preserving identity management systems

Souha Masmoudi

▶ **To cite this version:**

Souha Masmoudi. Malleable privacy-enhancing-technologies for privacy-preserving identity management systems. Cryptography and Security [cs.CR]. Institut Polytechnique de Paris, 2022. English. NNT : 2022IPPAS023 . tel-04861138

HAL Id: tel-04861138

https://theses.hal.science/tel-04861138v1

Submitted on 2 Jan 2025

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2022IPPAS023

Thèse de doctorat

Malleable Privacy-Enhancing-Technologies for Privacy-Preserving Identity Management Systems

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Telecom SudParis

École doctorale n°626 Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Evry, le 09 Décembre 2022, par

SOUHA MASMOUDI

Composition du Jury :

Benjamin Nguyen Professeur, INSA Centre Val de Loire, France	Président
Estelle Cherrier Maîtresse de conférences, ENSICAEN, France	Rapporteure
Melek Onen Maîtresse de conférences, EURECOM, France	Rapporteure
Eric Totel Professeur, Télécom SudParis, France	Examinateur
Olivier Blazy Professeur, Ecole Polytechnique, France	Examinateur
Sébastien Canard Ingénieur de recherche, Orange Labs, France	Examinateur
Maryline Laurent Professeure, Télécom SudParis, France	Directrice de thèse
Nesrine Kaaniche Maîtresse de conférences, Télécom SudParis, France	Co-encadrante de thèse

Abstract

For long decades, identity management solutions have been implemented to deal with users' digital identities and control their access rights to services and resources. During the past few years, and especially, from the beginning of the Covid-19 pandemic, an explosion of the use of digital identities has been driven by the emergence of new online services. Digital identities are, nowadays, used at a large scale (i.e., in public services, social medias, at work, online shopping, etc.). This brings usability issues as users are constrained to deal with multiple identities and attributes for access control and data sharing objectives. In addition, security and privacy challenges have arisen as the interacting entities, those that issue, process and collect these identities can, due to their behavior or security deficiencies, lead to identity theft, massive data collection and tracking of users' behaviors on the Internet. These challenges have significant influence on the security and the privacy of identity management systems.

This thesis aims at finding the best trade-off between security, privacy and usability for identity management systems, based on cryptographic primitives. The first two contributions focus on identity management for access control and consider real identities and attributes that contain personal (e.g., age) and sensitive (e.g., biometric traits) information.

The first contribution proposes a user-centric and privacy-preserving identity management system, named Pima, in which users keep control over their attributes. A user, that receives attributes certified by an identity provider, is able to interact, in a pseudonymized manner, with a service provider and prove the authenticity of the provided attributes while ensuring that he discloses only the minimum number of attributes. This solution is based on a new malleable signature scheme that allows users to modify the certificate issued by the identity provider on his attributes in a restricted and controlled manner. It also preserves privacy by satisfying the unlinkability property between curious service providers that try to link different transactions to the same user. An implementation of the solution demonstrates a high efficiency, while considering resource-constrained devices, i.e., Android-12 smartphone. Processing times do not exceed few milliseconds for a message of 100 attributes.

The second contribution presents a new biometric authentication scheme, named Puba, that offers robustness and privacy guarantees. Three steps are required. First, the user physically visits the identity provider that pushes an encrypted and certified biometric template onto his smartphone. Then he remotely enrolls at a service provider, in an anonymous manner. Finally, he authenticates offline to the service provider that captures a new biometric template in order to be locally verified via the smartphone. By relying on malleable signatures, the proposed solution prevents the use of fake biometric identities and guarantees the authentication soundness. Unlinkability and anonymity are also preserved, even when considering a collusion between curious identity and service providers. Experimental results, over biometric templates of 600 samples show acceptable processing times (i.e., no more than 6 seconds for one algorithm).

The third contribution provides a solution to meet the need of data sharing in

an identity management system. In particular, it studies the management of users ephemeral attributes in the context of proximity tracing for e-healthcare systems. The proposed solution, named Spot, ensures data consistency and integrity and preserves the privacy of users who share their contact information with people in proximity. Alerts are issued to users who have been in contact with infected persons. The use of a hybrid architecture, which relies on a centralized server and decentralized proxies, allows to prevent malicious users from injecting false alerts, and to prevent the linkability of contact information to the same user and the re-identification of users involved in contact with an infected person. The implementation of the protocol shows acceptable computation times that reach, for a 128-bits security level, 4 seconds to ensure the integrity of contact information, and 19 seconds to verify its authenticity. In an effort to improve Spot performances, we propose a group signature scheme that offers an efficient aggregated and batch verification over multiple proofs of knowledge. When being applied to the proposed proximity-tracing protocol, we prove a significant improvement in performances with up to 50% of gain during the verification of contact information authenticity.

Résumé

Depuis de nombreuses décennies, les solutions de gestion des identités sont développées pour gérer les identités numériques des utilisateurs et contrôler leurs droits d'accès aux services et aux ressources. Au cours de ces dernières années, et en particulier depuis le début de la pandémie du Covid-19, l'utilisation des identités numériques a subi une explosion conduite par l'émergence de nouveaux services en ligne. Les identités numériques sont, de nos jours, utilisées à grande échelle (par exemple, dans les services publics, les réseaux sociaux, au travail, les achats en ligne, etc.). Cela n'est pas sans poser des défis d'utilisabilité car les utilisateurs sont contraints de gérer de multiples identités et attributs pour des objectifs de contrôle d'accès et de partage de données. En outre, se posent des défis en sécurité et respect de la vie privée du fait que les entités en interaction, celles qui délivrent, traitent et collectent ces identités peuvent du fait de leur comportement ou d'insuffisances de sécurité aboutir aux vols d'identité, à la collecte massive de données et au traçage des utilisateurs. Ces défis ont une influence considérable sur la sécurité et la préservation de la vie privée dans les systèmes de gestion des identités.

Cette thèse vise à trouver le meilleur compromis entre sécurité, préservation de la vie privée et utilisabilité pour les systèmes de gestion des identités, en s'appuyant sur des primitives cryptographiques. Les deux premières contributions s'intéressent à la gestion des identités pour le contrôle d'accès et considèrent des identités et attributs réels qui contiennent des informations personnelles (ex : âge) et sensibles (ex : caractéristiques biométriques).

Pour répondre à cette préoccupation, la première contribution propose un système de gestion des identités centré sur l'utilisateur et respectueux de la vie privée, appelé Pima, dans lequel les utilisateurs gardent le contrôle sur leurs attributs. Un utilisateur, qui reçoit des attributs certifiés par un fournisseur d'identité, peut interagir de façon pseudonymisée avec un fournisseur de services et lui prouver l'authenticité des attributs présentés tout en minimisant le nombre de ces attributs. Cette solution s'appuie sur un nouveau schéma de signature malléable qui permet aux utilisateurs de transformer le certificat issu du fournisseur d'identités sur ses attributs de façon restreinte et contrôlée. Elle préserve aussi la vie privée en satisfaisant les propriétés de non-associabilité entre des fournisseurs de services curieux qui tenteraient d'associer différentes transactions à un même utilisateur. Une mise en œuvre de la solution démontre une grande efficacité, tout en considérant des dispositifs à capacités limitées, comme un smartphone Android-12. Les temps de calcul ne dépassent pas quelques millisecondes pour un message de 100 attributs.

La deuxième contribution porte sur un nouveau schéma d'authentification biométrique, appelé Puba, qui offre des garanties de robustesse et de respect de la vie privée. Trois étapes sont nécessaires. Tout d'abord, l'utilisateur se rend physiquement chez le fournisseur d'identités qui pousse le modèle biométrique chiffré et certifié sur son smartphone. Puis il s'enregistre à distance auprès d'un fournisseur de services, de façon anonyme. Enfin, il s'authentifie hors ligne auprès du fournisseur de services qui capture la modalité biométrique, cette modalité étant vérifiée localement via le smartphone. En s'appuyant

sur des signatures malléables, la solution proposée empêche l'utilisation de fausses identités biométriques et garantit la fiabilité de l'authentification. La non-associabilité et l'anonymat, sont aussi préservées. Les résultats expérimentaux, sur des modèles biométriques de 600 échantillons, montrent des temps de calcul acceptables qui ne dépassent pas 6 secondes pour un seul algorithme.

La troisième contribution apporte une solution au besoin de partager des données dans un système de gestion des identités, et en particulier étudie la gestion des attributs éphémères des utilisateurs dans le contexte du traçage de proximité pour les systèmes d'e-santé. La solution proposée, appelée Spot, assure la cohérence et l'intégrité des données et préserve la vie privée des utilisateurs qui partagent leurs informations de contact avec les personnes à proximité. Des alertes sont émises vers les personnes ayant été en contact avec des personnes infectées. L'architecture hybride utilisée qui repose sur un serveur centralisé et des proxies décentralisés empêche les utilisateurs malveillants d'injecter de fausses alertes, et empêche de relier toute information de contact à un même utilisateur et de réidentifier les utilisateurs impliqués dans un contact avec une personne infectée. L'implémentation du protocole montre des temps de calcul acceptables qui atteignent, pour un niveau de sécurité de 128 bits, 4 secondes pour assurer l'intégrité des informations de contact, et 19 secondes pour vérifier leur authenticité. Dans le but d'améliorer les performances de Spot, nous proposons un schéma de signature de groupe qui offre une vérification agrégée par lot efficace sur plusieurs preuves de connaissance. En appliquant ce schéma au protocole de traçage de proximité proposé, nous prouvons une amélioration significative des performances avec un gain allant jusqu'à 50% pendant la vérification de l'authenticité des informations de contact.

Acknowledgement

First and foremost, I would like to express my deepest gratitude to Maryline LAURENT, my thesis director, for her assistance, her encouragement, her trust and her relevant advices throughout the thesis. It is absolutely difficult to succeed in the process of finding and developing an idea without the help of a person who generously provided knowledge and expertise.

I am deeply indebted to Nesrine KAANICHE, my thesis co-supervisor, for her moral and scientific indescribable help, her valuable advice, her extraordinary availability and her remarkable communicability. It was a real pleasure to work with her.

I would like to extend my sincere thanks to the Chair Values and Policies of Personal Information "VP-IP" for the funding of my thesis. Thanks to all the members and partners for their very interesting presentations and for the valuable discussion that we had during workshops.

I would like to thank the members of the jury for their interest to my research work and for being part of the jury of my thesis. Special thanks to Dr. Estelle CHERRIER and Dr. Melek Onen for their thorough reading of the dissertation and for all the advice they gave me.

I am very thankful to Damien VERGNAUD for reading the first versions of Pima and for his relevant advice to improve the security of the scheme. Thanks to Fadel RADJI for his implementation to Spot and his significant effort to improve the performances of the solution.

I would be remiss in not mentioning my family, especially, my parents, my husband, my siblings and my nephew. Their belief in me has kept my spirits and motivation high during all my academic career. Thanks for providing the best conditions, love, generous care, and support to achieve my thesis.

Thanks a lot to all my friends, and all the people who from near or far have supported me and contributed to the accomplishment of this thesis.

I can not conclude this acknowledgement without thanking my baby Fedi who I look forward to carrying in my arms, for supporting my rough days of work and for the push he gives me even before coming to this life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Every new beginning comes from some other beginning's end.

– Seneca

Digital identity is emerging as one of the most pervasive technology trends worldwide. An acceleration in the use of digital identities, has been driven by the explosion of online services due to the Covid-19 crisis, leading to a massive collection of sensitive data by different actors [48], and several attacks performed on data collected [49].

To illustrate this massive collection, let us consider, a travelling experience by airplane, during the Covid-19 pandemic, in France. During this period, several governments, in particular France, have encouraged the use of digital services, such as teleworking, online purchasing using credit cards, remote medical diagnosis, etc. To comply to these new trends for travelling, a traveler Alice first, purchases the flight ticket online. She is then asked to provide identifiable information, like name, date of birth, passport number and many other personal data. She also provides her credit card details to pay the ticket fees. A confirmation e-mail is sent by the airline company including, the flight's details, the traveler's personal information given during booking and the payment details like the last four digits of the credit card number. If Alice is using a Google e-mail account for the booking, then the flight's information are uploaded to her digital calendar. Due to restrictions imposed by governments during the pandemic, Alice might be asked to perform some additional steps before departure as an extra border's check. For instance, she has to get tested for the Covid-19. For this test, Alice is asked, at the medical laboratory, to provide again her personal information (e.g., name, date of birth, social security number) often stored on her health card. Alice uses her credit card to pay the test fees, thus, the last four digits of her credit card number are again available at the laboratory. The payment receipt could be also sent by e-mail. To retrieve results, Alice has access to a centralized platform where she finds a certificate over her personal information and the tests details (i.e., result and date). Thanks to a QR code, she is able to upload the certificate on an application, installed on

her smartphone, constituting her certificates' wallet (i.e., the wallet includes vaccination and tests certificates). On the day of the flight, Alice takes public transportation to reach the airport. In order to be notified in case of contact with infected people, she relies on a proximity-tracing application managed by a centralized authority. This application broadcasts her contact information (i.e., a Bluetooth identifier) and collects that of others in proximity and notifies him in case of risk. At the airport, using her passport and her booking ticket, Alice receives her boarding pass. A control of the Covid-19 test and vaccination certificates is reinforced at the airport. Alice provides the QR code and the passport to enable officers to check the validity of certificates (i.e., through a specific application) and the correct matching with her identity. After passing through the security check steps that may take at least two hours, Alice is waiting for boarding. Meanwhile, she buys some gifts at the airport shops. For this purpose, she is asked to provide her boarding pass to check whether an exoneration of the 20% VAT charge can be applied. She also connects to the airport WIFI by communicating her e-mail and her flight number, in order to post some photos on social media. After a last check of the boarding pass, the passport and maybe the vaccination certificate at boarding, Alice is finally sitting on the plane and ready for the taking-off.

In this simple example where the use of digital identities is ubiquitous, it is important to summarize the issues at stake. First, when buying a ticket, Alice is not only providing identity information and personal details to the airline company. But, when approving the website's terms of use and policies, she may also give her consent to allow the airline company to collect, store and share her data with other parties (e.g., news and advertising companies). Therefore, Alice may be identified by these companies according to the given identifiable information.
Second, at the laboratory, when providing the health card, officers can access to more data than they need to know. For instance, they can see Alice's history of medical visits and many other sensitive information. Thus, Alice can be involuntarily traced by the laboratory or any other party that requests her health card.
Third, when using the proximity-tracing application, Alice does not only release her contact information, but involuntarily, she also enables the centralized authority to recognize people she is getting in contact with. A social graph can be constructed when combining several users' contact information.
Additionally, showing the same vaccination certificate to different parties, in different contexts, leads to possible tracking of the traveler's movements.
Finally, the use of the same information, like name and credit card number, has become the target of attackers that link information across multiple transactions. Hence, partial or full profiles can be built about the traveler. All the aforementioned issues undermine users' privacy.

Identity management (IDM) is the core concept dealing with users' information on the Internet. Several criteria have been defined to design identity management systems. They include the entity responsible for the identity governance, the storage location of identities and attributes, the power given to central entities like identity providers and the level of trust given to the different actors. According to these criteria, different identity models have been rolled out during the past decades [35]. First, the centralized model fully trusts the identity provider, responsible for delivering users' identities and

attributes, to authenticate users at several service providers. This role gives him the capability to track users' behaviors. Second, when different service providers aim at collaborating with each other, the federated model has evolved. Indeed, users can go through different services by authenticating at only one service provider belonging to the federated domain. However, users can be always traced and profiled by service providers [131]. Growing awareness about users' privacy, thanks to the emergence of new regulations (e.g., the General Data Protection Regulation (GDPR) [57]), has led to the development of the user-centric model. This concept puts users into full control over their identities. This model has raised additional concerns about the authenticity of data provided by users. When choosing to use one of the aforementioned models, the users' first goal is to benefit from a smooth management of their identities and information on the Internet.

1.1 Problem Statement and Objectives

For a long time, digital identity management solutions have been used inside companies and institutes. They aim at controlling users' access rights to the different services and resources. Nowadays, digital identity faces new challenges as its use witnesses an explosion that covers the whole society [141]. For instance, regal digital identity programs are increasingly being adopted by governments. Some of these programs include biometric identities, namely electronic passports that have rapidly emerged with more than one billion passports in circulation nowadays [138].
The definition of digital identity does not only include digital representations of an individual, but also covers all the traces that he leaves on the Internet and that are attached to him. The digital identity is then becoming the link between the user and the society (i.e., individuals, government, groups and organizations, social medias, etc.). As such, users find themselves overwhelmed by multiple identities delivered and/or managed by these different parties. The challenge here is to create interoperable and coherent identity solutions to improve the user experience and facilitate the use of digital identities, which refers to the **usability** of the identity management system.
To meet this challenge, it is agreed that providing the user with the full control over his identities is a promising solution, which is in compliance with regulations (e.g., GDPR [57]) and standards (e.g., ISO 18013-5 [133]). Indeed, the user is the common party to all identity systems. As he is always using digital services, he is increasingly aware about privacy issues [1].

IDM is faced with a large scale of frauds, attacks [48] (i.e., usually performed by malicious users or system's outsiders) and data collection [49] (i.e., by curious service providers and/or identity providers). For instance, from the beginning of the Covid-19 crisis till now, the Federal Trade Commission (FTC) has handled more than 70 thousands of identity fraud reports for online shopping services [48]. Moreover, according to [47], Google, for example, collects 10 exabytes of data per day.
To deal with these issues, it is crucial to establish a framework of trust between the entities involved in an IDM solution, namely identity providers, users and service

providers. This framework should guarantee the protection of the data and their owners, referred to as data **security** and users' **privacy**:

- Data security: deals with two mains concerns. The first one is the confidentiality to keep data secret from unauthorized parties. The second concern is data consistency and integrity. It guarantees that fake or modified data can neither be shared with other parties, nor used to get access to services and resources. This suggests that a rigorous control should be applied on data to verify its validity. Corrupted and invalid data should be rejected/ignored.
- Users' privacy: suggests that even several parts of user' information are collected by several parties, they cannot be used to track his behavior or build a partial/full profile of him. Privacy also implies that users should not be re-identified according to the communicated information and known by multiple independent parties.

In short, IDM is faced with challenges is terms of security, privacy and usability.

In this thesis, proposed in partnership with the Chair Values and Policies of Personal Information (VP-IP), our goal is to gain a better understanding of identity management and its requirements. This, is with the aim of providing concrete solutions that guarantee the trade-off between security, privacy and usability. That is, designing an identity management system where a user holding certified identities/attributes is able to manipulate them and to prove to service providers their validity. As such, the user is able to access services and resources or share data with different parties, without privacy threats. This aim can be translated into the following objectives:

- ***Objective A*** – affording users the empowerment and the full control over their identities and attributes in order to select and disclose the minimum amount of data that satisfies service providers' policies.
- ***Objective B*** – controlling and verifying the validity of data and computations performed by users over their data, through rigorous mechanisms, before authorizing them to access services and resources.
- ***Objective C*** – designing a secure mechanism to manage ephemeral identities.
- ***Objective D*** – guaranteeing the users' anonymity/pseudonymity and the unlinkability between their transactions in order to avoid involuntarily identification, tracking and profiling of users.
- ***Objective E*** – providing formal or informal proofs of the security and the privacy properties of the proposed constructions.
- ***Objective F*** – providing a prototype of the proposed solutions and implementing the different algorithms, in order to validate their feasibility and to evaluate their efficiency on real hardware.

To satisfy the aforementioned objectives, we set two assumptions about the level of anonymity and the cryptographic mechanisms. Indeed, we assume to use *(i)* pseudonyms to interact with other parties, and *(ii)* malleable Privacy-Enhancing Technologies (PETs) as cryptographic tools to develop our solutions. Hereafter we explain our choices.

- *(i) Pseudonyms* – Pseudonyms are used to mitigate the threats presented when users' real identities are used during interaction with other parties [16]. Indeed, transactions and accounts associated to the same identity can be linked to each other among different service providers which undermine users' privacy. Thus, pseudonyms can be used to prevent such linkability issues. However, there are many situations where a controlled linkability of pseudonyms, by the same service provider, is desirable. For instance, a user can re-use the same pseudonym with a specific service provider, with no need to re-calculate the proof over already verified attributes. Also, in a chat-bot use case scenario, web services prefer to maintain state information per user in order to keep a conversation thread with the same person that they started it with [16].
- *(ii) Malleable PETs* – Using PETs as a building block of a particular scheme helps ensure its preservation of privacy. Malleability is a very useful feature that enables to introduce controlled modifications on the data, mainly on its form, its content or both. Indeed, malleable PETs allow an authority to issue a cryptographic element that has certain properties and that can then be manipulated without losing its properties.

1.2 Contributions

When designing identity management solutions for **access control** and **data sharing**, we take into consideration two main functional requirements: *(i)* user-friendly experience referring to the simplicity of managing multiple attributes and identities and, *(ii)* trusted/verifiable identities approving the consistency of users' identities and attributes, i.e., either service providers grant access only to authorized and legitimate users, or users share only correct and unaltered data. The contributions of this dissertation are summarized below.

- *Contribution 1* – design of a privacy-preserving identity management system based on pseudonyms and an unlinkable malleable signature [94]. The proposed system allows to **control users' access** to services and resources. That is, every user receives a certificate, over his attributes, delivered by a trusted identity provider. Thanks to malleable signatures, he is able to remove attributes that he does not want to disclose and keeps only the minimum data required to get access to services. The user accordingly adapts the certificate. This solution provides users with full control over their identities and attributes, while allowing service providers to verify the authenticity of the data provided, through pseudonymized sessions. As such, service providers are not able to link users' transactions. An implementation of the proposed solution demonstrates very good performances.

Indeed, when considering a message of 100 blocks, an amount of to 50% of admissible blocks and an amount of 25% (of all blocks) of modified blocks, the computation times, obtained on both a laptop and a smartphone, are of the order of a few milliseconds (*Objective A*, *Objective B*, *Objective D*, *Objective E* and *Objective F*).

- *Contribution 2* – proposition of a privacy-preserving biometric authentication scheme for identity management systems [95], ensuring the link to the physical person as a physical **access control**. That is, a user receives a template representing his biometric identity. This template is encrypted and certified by an identity provider. The user is then able to modify these credentials without compromising their authenticity, and to anonymously register at a service provider. To authenticate, the user is prompted to present a new biometric template to the service provider, which calculates the difference with the one provided at registration. The proposed solution demonstrates the resistance against the use of fake biometric identities, the soundness of the authentication and the users' privacy preservation support. When considering a biometric template of 600 samples, acceptable processing times, that do not exceed 6 seconds for one algorithm, prove the efficiency of the solution (*Objective A*, *Objective B*, *Objective D*, *Objective E* and *Objective F*).

- *Contribution 3* – design of a privacy-preserving protocol for **data sharing**. This protocol deals with a specific type of attributes, i.e., ephemeral attributes, in the context of proximity-tracing for e-healthcare systems [93]. The proposed solution relies on a hybrid architecture that involves a centralized server and decentralized proxies. It allows users to share their contact information with other persons and to be notified in the case of being in proximity with infected people. Thanks to the security of the protocol, users are ensured to only receive correct alerts, i.e., false positive alerts are detected. A detailed security analysis demonstrates the resistance against linkability and identification trials. An implementation of the solution with an effort to minimize the computational costs of the proposed protocol shows its efficiency. For instance, to guarantee the non-falsification of a contact, the computation time reaches 4 seconds for a security level of 128-bits. The verification of the authenticity of this contact requires 19 seconds on a machine with low capacities (*Objective A*, *Objective C*, *Objective D*, *Objective E* and *Objective F*).

- *Contribution 4* – proposition of a group signature scheme that offers an efficient, aggregated and batch verification over multiple proofs of knowledge [96]. It ensures integrity during **data sharing**. The implementation of the solution demonstrates a high efficient of the aggregated and batch verification in comparison with a naive group signature verification, with no loss in terms of security and privacy (*Objective D*, *Objective E* and *Objective F*).

1.3 Thesis Organization

This thesis is structured into six chapters. Chapter 2 presents an introduction to identity management systems and some cryptographic tools that support IDM systems' requirements. In Chapters 3 and 4, we consider identity management solutions that deal with users' real/permanent or moderately static identities and attributes like age and biometric traits, to control access to services and resources. Chapter 5 focuses on users' data sharing among different entities. and explores the privacy-preserving management of users ephemeral attributes.

- **Chapter 2 – Malleable encryption and signature schemes for Privacy-preserving Identity Management Systems** – first, gives a comprehensive review of identity management systems process and analyses the commonly used identity models from security, privacy and usability perspectives. Second, it evaluates malleable privacy-enhancing technologies with respect to their relevance in dealing with identity management challenges.
- **Chapter 3 – A Privacy-preserving Identity Management System based on an Unlinkable Malleable Signature** – is an updated version of the paper "PIMA: A Privacy-preserving Identity Management Systems based on an Unlinkable Malleable Signature" [94]. This chapter describes Contribution 1. A novel user-centric identity management system is introduced allowing users to derive different pseudonyms from a local pseudonym. Users prove, to service providers, the authenticity of their attributes associated to the derived pseudonym, in a strong privacy-preserving manner.
- **Chapter 4 – A privacy-preserving and User-centric Biometric Authentication protocol through Malleable Signatures** – is an updated version of the paper "PUBA : Privacy-preserving and User-centric Biometric Authentication through Malleable Signatures" [95]. The paper is under submission. This chapter presents Contribution 2. It introduces a novel biometric protocol that ensures the correctness of the biometric information and the soundness of an offline authentication without relying on users' devices security and trust on entities.
- **Chapter 5 – A secure and Privacy-preserving Proximity-tracing Protocol for e-healthcare Systems** – is an updated and extended version of the paper "SPOT: Secure and Privacy-preserving Proximity-tracing Protocol for E-healthcare Systems" [93] with respect to the paper "SEVIL: Secure and Efficient Verification over Massive Proofs of Knowledge" [96]. This chapter is a fusion of Contributions 3 and 4. A novel privacy-preserving proximity-tracing protocol is proposed to prevent the injection of false positive alerts. That is, through a centralized server and a group of decentralized proxies, the authenticity of contact information is guaranteed.
- **Chapter 6 – Conclusions and Perspectives** – concludes the dissertation with a summary of contributions and gives an overview about our perspectives for future work.

Chapter 2

Malleable encryption and signature schemes for Privacy-preserving Identity Management Systems

We define our identity always in dialogue with, sometimes in struggle against, the things our significant others want to see in us.

– Charles Taylor

2.1 Introduction

The introduction of digital identity management systems has been considered as a crucial step towards a more transparent and trustworthy environment where individuals can remotely interact with other society components. This remote interaction comes with challenges regarding both the security of information exchanged and users' authentication. Concurrently, the cooperation between different entities to interconnect their databases, aggregate metadata and massively collect information, has induced individuals' tracking and surveillance. This leads to a paradox where identity management systems aim at improving the users' experience and securing exchanges on the one hand, but may compromise users' privacy on the other hand. To address this dilemma, different identity models have been deployed while supporting various criteria, i.e., the storage location of users' identities, the entities responsible for either governing or managing identities when accessing services, the potential of entities to trace users, etc. According to the growing awareness about users' privacy, a great attention has been put on modern cryptographic technologies, namely Privacy-Enhancing Technologies (PET), in order to design privacy-preserving identity management systems.

In this chapter, we present a state of the art about identity management systems focusing on privacy-preserving needs. For this purpose, we first provide a comprehensive review of identity management systems while presenting the different involved actors, enumerating properties that must be fulfilled and identifying identity models. Then, we present an introductory compendium to some cryptographic mechanisms and technologies often used to manage identities and protect sensitive data, i.e., malleable PETs.

Next, we give a general introduction to identity management systems in Section 2.2. Then, we review, in Section 2.3, the technical advances in identity management systems contributing to better management and control over users' multiple identities. A particular attention is given to malleable privacy-enhancing technologies.

2.2 Identity Management Systems

In this section, we first introduce identity and identity management systems (Section 2.2.1). Second, we enumerate the requirements of identity management systems in the digital era (Section 2.2.2). Then, we present and compare identity models that have been developed to fulfill the desired properties (Section 2.2.3).

2.2.1 Identity and Identity Management

This section introduces the concept of digital identity being at the core of identity management systems and details the evolution of identity management systems' role with the emergence and the evolution of new digital services and trends.

2.2.1.1 Digital Identity

The term digital identity often refers to a representation of an entity within a scope [6]. An entity may have one or several identities in a particular domain. For example, a person may have two identities in a bank system because he is both an employee and a customer at the bank.

Digital identity represents the computer and technological means that enable this entity to project itself into the digital world, and this projection may take several forms depending on its context (administrative, professional, social networks, etc.) [111]. From a technical point of view, this representation consists in associating a set of numerical data, referred to as attributes and credentials[1], with an individual. These attributes include permanent traits like ethnicity and date of birth. Attributes may be also moderately static or very difficult to change over a person's lifetime, e.g., surname, first name, voice print and eye color. Attributes also relate to highly dynamic traits, e.g., the individual's centers of interest, the person's geolocation, communication and housing information (i.e., phone number, email and address). New forms of digital attributes have evolved, including biometrics (e.g., fingerprints, facial recognition, iris scan, etc.) that enable to create unique identifiers for individuals, for example when validating a payment transaction on a mobile device.
Thus, a digital identity can be considered as the mean to distinguish one entity from others online. In other words, each entity has several identities simultaneously in different contexts. Each identity consists of a set of attributes. This set may contain an attribute that uniquely identifies the entity within a scope, called a name or an identifier [77].

2.2.1.2 Identity Management

The Identity Management (IDM) concept was introduced in order to simplify the management of digital identities within a specific domain or system. IDM is a framework used in computer systems that implements the enrollment of users into the system (to assign them unique digital identities), the authentication and authorization management, the access control to services and resources, and the technical measures to provide the security and privacy of identities [40].

Identity management systems have been existing for a long time and present several advantages according to different usages.

On the one hand, it serves to enhance security, within companies first, to better manage individuals' identities (i.e., employees' identities) and to provide them with a range of services with respect to a predefined access policy. In the corporate context, identity management is known as Identity and Access Management (IAM). In a first

[1] A credential represents the information elements used in the authentication of the claimed attributes or identity belonging to a particular entity [105]

phase, IAM creates and manages credentials and attributes, and in a second phase, it evaluates these attributes to make a decision about giving access to a particular service or resource.

On the other hand, identity management has evolved to establish and increase trust between several entities in different contexts. For example, identity management plays an important role in the context of social networks and content sharing platforms (e.g., Facebook, Twitter, YouTube, Tik Tok, etc.) as it enables them to increase trust towards their users by providing a favorable environment with a limited number of spoofers. As a result, users feel freer to express themselves and share opinions, photos, comments, likes and dislikes, etc. Inside this space of freedom, some users want to preserve their privacy (intimacy). However, platforms have financial interest in collecting data and they are compelled to apply regulations (e.g., censorship of racist statements, bringing the individuals accused of racist comments on their platform before a court of law) and to find out the real identity of accused users (i.e., accountability) so that justice can be informed and rendered [102, 137, 136].

Additionally, identity management systems provide higher trust in a platform context, e.g., Uber, Airbnb, for individuals to be put in contact with each other in order to provide services for them. Such platforms want to guarantee trust between the user (i.e., the entity requesting a service) and the service provider (i.e., the entity offering the requested service and willing to be paid for it). Inside this space of trust, users want to be ensured that their requests remain confidential and are not available to other users willing to collect data about them (e.g., a boss wanting to collect information about his employees' Uber courses).

Moreover, as several governments have provided full online regal services, there is a need for identity management to ensure mutual trust between users revealing their information (e.g., properties, income, etc.) and governments providing state services (e.g., online tax reporting).

Entities An identity management system includes three main entities, namely user ($\mathcal{U}$), service provider ($\mathcal{SP}$) and identity provider ($\mathcal{IP}$) and an optional entity called attribute provider ($\mathcal{AP}$):

- **User:** is a physical person, an organization or a device, which is provided by one or several digital identities managed by one or several identity providers. These identities give him the eligibility to get access to services and resources provided by one or several service providers.

- **Service Provider:** is the entity responsible for providing online services to users such as e-commerce, e-banking, e-health, cloud storage, etc. For this purpose, $\mathcal{SP}$ should authenticate the user by making sure that he is the one claimed to access the service. With respect to predefined access policies, service providers collect, from users, attributes that characterize them. In such a case, a service provider must refer to a trusted third party.

- **Identity Provider:** is responsible for delivering identities and basic attributes to users (i.e., attributes constituting the user's identity). $\mathcal{IP}$ maintains and confirms the attributes required in the process of users' authentication towards service providers (e.g., name, bank account/card number, social security number, etc.).
- **Attribute Providers:** represent optional actors that provide additional attributes describing the user. Thus, while the $\mathcal{IP}$ confirms to $\mathcal{SP}$ the digital identity itself and the basic attributes, the $\mathcal{AP}$ only confirms additional attributes [111] such as tax income or professional practice.

Note that the identity provider and attribute providers can be merged into a single entity referred to as a trusted third party.

2.2.2 Identity Management Systems' Requirements

In the digital world, a range of issues face identity management systems seeking to establish trust between the different entities (i.e., users, identity, attribute and service providers). Indeed, IDM systems aim at *(i)* securely dealing with the huge amount of users' information and personal sensitive data (e.g., health-related data, biometric identities), *(ii)* preserving users' privacy with respect to regulations and *(iii)* providing a better user experience. These issues are formalized into three types of requirements referred to as security, privacy and usability requirements.

2.2.2.1 Security Requirements

In this age of data leakage, identity theft, phishing and hacking, identity management systems should ensure the security of communications and protect the data exchanged between the different entities. We define hereafter the common security issues [123, 132] to deal with.

- **(S1) Confidentiality** – ensures that users' sensitive data are protected during its transfer and when being stored (e.g., in the cloud) and can only be read by authorized entities.
- **(S2) Data integrity** – ensures that users' data has not been altered by unauthorized entities.
- **(S3) Authenticity** – ensure that users' data is original and it reaches its destination as it was sent by its owner.
- **(S4) Peer-entity authentication** – ensures that the peer entity in a communication is the one claimed.
- **(S5) Data origin authentication** – ensures that the source of data received is the one claimed.

- **(S6) Authorization** – determines the users' rights and privileges to access services, resources, files, programs, etc.
- **(S7) Availability** – ensures that the system properly operates whenever and wherever entities need it.
- **(S8) Non-repudiation** – ensures that an entity having executed a transaction can not deny the action.

2.2.2.2 Privacy Requirements

While using identity management systems, information about users, as well as the contents of their transactions on the Internet, are collected, processed and stored either by identity providers or by service providers. As a result, the user's privacy may be threatened due identification, tracing and profiling attempts. To deal with this problem, the following privacy requirements should/can be fulfilled:

- **(P1) Anonymity** – ensures that a user is able to access to a service or a resource while maintaining his identity anonymous (not identifiable). As a result, the user cannot "be identified directly or indirectly" [72] by the $\mathcal{IP}$ or $\mathcal{SP}$.
- **(P2) Pseudonymity** – represents a lower level of anonymity. In fact, during the interactions, the user is known to the $\mathcal{IP}$s or $\mathcal{SP}$s through a (temporary or permanent) alias which can be reversed to the identity of the user. The user is identifiable to identity providers by an "alias" [72]. Pseudonymity helps to support the concept of a user holding multiple virtual identities.
- **(P3) Data minimization** – is a fundamental privacy principle and a GDPR requirement that requires services and applications to process only the minimum amount of information strictly necessary for the service or for a particular transaction [32]. The aim is to minimize the amount of personal data collected and used by online service providers, for example, to reduce the risk of profiling based on the user's behavior.
- **(P4) Unlinkability** – refers to the inability to link two or more distinct pieces of information (records, messages, URLs, actions, identifiers) about a user or a group of users. On the one hand, unlinkability may concern the inability to link proofs of identity to the original identity/credential, relying on the information provided during the identity issuing process. Such a property is called ***issue-show unlinkability***. On the other hand, the ***multi-show unlinkability*** refers to the inability to link several identity proofs generated over the same identity/credential and transmitted over several sessions.

2.2.2.3 Usability Requirements

Considering the increasing number of online services, users find themselves overloaded with several identities and passwords that they should memorize, thus often resulting

in people using the same password in different contexts [104]. To solve this problem and to improve the users' experience, identity management systems have to achieve the usability requirements [5] presented hereafter.

- **(U1) Multiple identities' management** – is a principle that has been strongly supported by the chair Values and Policies of Personal Information (VP-IP) [79]. It consists in establishing the best practices to help users to manage their multiple identities that they use to present themselves to others in different contexts. For example, the same user can be presented as a father, a teacher, a bank customer, a music fan, a diabetic person, etc. For this purpose, different practices could be taken into consideration including pseudonymization technologies depending on the context. Users can select the attributes to be disclosed and associate them to a pseudonym specific to each service provider.

- **(U2) Unique identity for multiple transactions** – states that a user may have a unique identity that contains all the possible credentials he needs to perform transactions with several service providers. The user is then given the possibility to manage his credentials in order to provide only the required ones when performing a particular transaction.

2.2.3 Identity Management Systems' Categories

In this section, we describe traditional and current identity models, namely centralized, federated and user centric.

2.2.3.1 Centralized Identity

In centralized identity management systems, a unique identifier and several credentials generated by the same identity provider, are used by every service provider [76]. Figure 2.1 shows the centralized identity model. A user provides the same identifier and credentials, stored at the identity provider, to multiple service providers in order to access services. To authenticate a user, the service provider asks the identity provider, considered as a trusted party, to verify the user's identity.

The centralized identity model is mainly deployed in companies where identities and credentials are stored in a central database and managed by a central entity.
This identity model is also used in the context of the regalian identity (e.g., inserted in an electronic identity card) for accessing to online administrative governmental services (e.g., civil status, taxes, ...). This identity is certainly delivered by a trusted third party, such as the Ministry of the Interior or the tax administration. When a user logs on to an online regalian service, the service provider asks the identity provider to confirm the attributes of the provided identity. Thus, the identity provider is at the core of all users' transactions and knows their contents. We note that in some cases, namely identity loss or theft, the identity provider is able to revoke the identity.

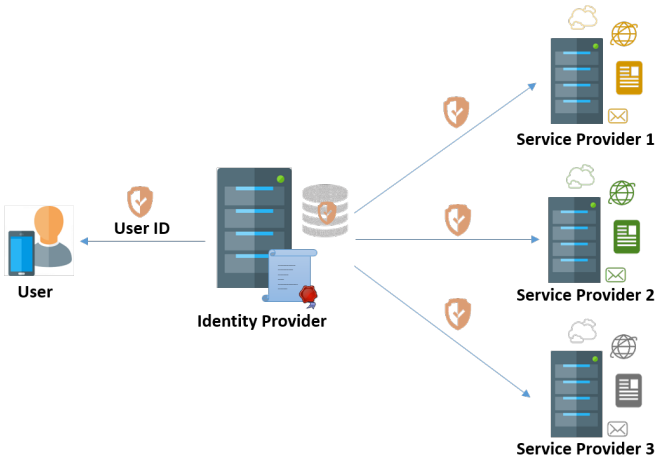


Figure 2.1: Centralized Identity Model

2.2.3.2 Federated Identity

In federated identity management systems, a group of service providers establish a set of agreements and standards so that users' identifiers are recognized from one service provider to another inside the federation referred to as a trust domain [40]. The federated identity model, as presented in Figure 2.2, suggests that a user may have different identifiers for each service provider but he could instead use a single identifier to authenticate to all service providers belonging to a particular federated domain. Indeed, upon accessing a service, he should authenticate to the corresponding service provider using his unique identifier. Through this unique authentication, the user is also able to access to other services offered by other service providers within the federated domain, as there is a mapping done between identifiers belonging to the same user.

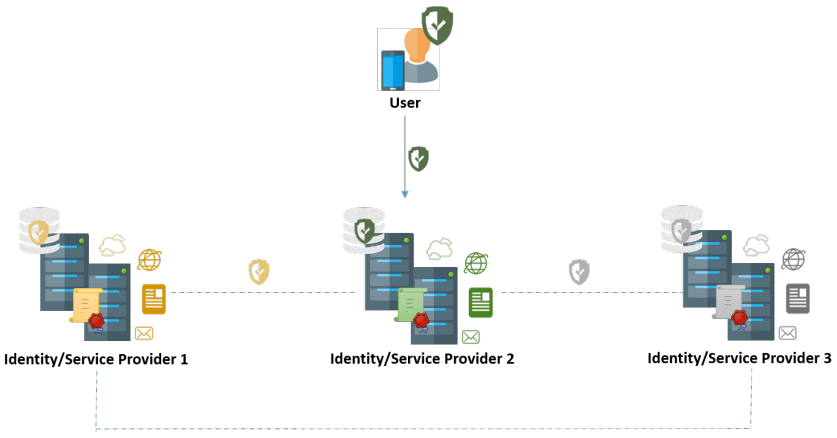


Figure 2.2: Federated Identity Model

This model is used in the Single Sign-On (SSO) solution. For example, in the case of a digital identity based on a social network such as Facebook Connect [100] or Google+ Sign-In [52], the user creates an online account while providing personal

data (for example at $\mathcal{IP}_2/\mathcal{SP}_2$ as illustrated in Figure 2.2). During this phase, the user sets an identifier (e.g., e-mail address) and a password that can be later used for identification and authentication. When the same user wants to buy clothes online, e-shopping website (i.e., service provider $\mathcal{IP}_3/\mathcal{SP}_3$, as depicted in Figure 2.2) asks him to authenticate using his digital identity attached to his social network (i.e., service provider). The authentication is then delegated to the social network that verifies the provided credentials (login and password). Other standards for federated identity have been implemented including OASIS (Organization for the Advancement of Structured Information Standards), SAML[2] (Security Assertion Markup Language), the Liberty Alliance framework[3], the open source implementation Shibboleth[4] [40] and the OpenID Connect framework built on OAuth2[5].

2.2.3.3 User-centric Identity

The user-centric identity management model, illustrated in Figure 2.3, was introduced to automate and support users' identity management at the user side. Users are given the control over their identities [76]. They are able to select information to disclose and to be notified when their information are collected. Users' consent is also required for any type of analysis and manipulation over their collected information. The user identity and attributes can be stored in a hardware device (e.g., smart card, portable personal device). As such, the user only needs to memorize one credential (i.e., to access to the hardware device) instead of remembering several identifiers and credentials.

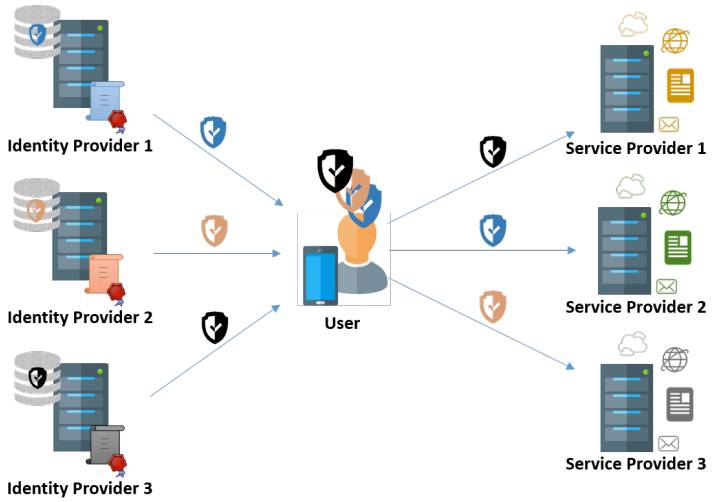


Figure 2.3: User centric Identity Model

[2] https://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf
[3] http://www.projectliberty.org/liberty/content/download/318/2366/file/draft-liberty-idff-arch-overview-1.2-errata-v1.0.pdf
[4] http://people.cs.vt.edu/~kafura/cs6204/Readings/Authentication/ShibbolethArchitecture&Protocols.pdf
[5] https://auth0.com/docs/authenticate/protocols/openid-connect-protocol

Among user-centric identity management system implementations, we can mention the OpenID framework that is already used in well-known Web platforms such as Drupal[6] and WordPress[7]. In the OpenID framework, a user can be identified by an URL (Uniform Resource Locator) or an XRI (EXtensible Resource Identifier) address. The uniqueness of URLs, resp. XRI, makes the user uniquely identified. OpenID enables a user to choose his identity provider as well as his identity [54]. In fact, he has the ability to select his digital address (identity) he wants to send to a service provider. Afterwards, the service provider redirects the user to the identity provider. Once authenticated, the $\mathcal{IP}$ redirects the user to the $\mathcal{SP}$ with a credential that proves the user's URL and the data he has released.
On its side, Microsoft has adopted the user-centric identity model when developing InfoCard/CardSpace [97]. Using this technology, all the identities issued by identity providers are stored at the user's side. Then, when a user wants to access to a service, he should prove his identity to the service provider with a credential, called security token, that contains one or more claims including user's information like user name, home address, etc. Indeed, once being requested, the service provider sends to the user his security policy containing the information and the claims needed for authentication. The security policy is then forwarded to the appropriate identity provider which is sending back the generated security token for the user to authenticate to the service provider. CardSpace is compatible with any technology or platform that supports web services.
IBM has also supported the user-centric identity model in the Higgins Project[8]. This open source framework relies on the concept of a "selector" which involves a set of information cards (i.e., also called i-cards) and is integrated with the user's browser. These cards can be either generated by the user himself or by an i-card provider website. Each one contains a set of claims about the user including his preferences and interests. To log into websites and access services, the user selects the i-card which contains the claims that he want to disclose.

2.2.3.4 Evaluation of Identity Models

In this section, we give a comparison between presented identity models, as depicted in Table 2.1 and Figure 2.4, w.r.t. their advantages, drawbacks and the identity management requirements they satisfy.

Note that in Table 2.1, we respect the chronological development of identity models, i.e., from the centralized model to the user-centric model going through the federated model.

Advantages and Drawbacks based Discussion We summarize in Table 2.1 the advantages and the drawbacks of identity management categories.

[6] https://www.drupal.org/
[7] https://wordpress.com/
[8] https://projects.eclipse.org/projects/technology.higgins

Table 2.1: Comparison between Identity Models

Identity model	Advantages	Drawbacks
Centralized	◆ A single credential is needed to access many services and resources. ◆ Users cannot authenticate using fake identities and credentials. ◆ Users' pseudonymous authentication is possible.	◆ Too much power is given to a centralized identity provider that is able to track and create full profiles of users. ◆ The centralized model represents a single point of failure (i.e., the identity provider) which makes it that vulnerable to DoS attacks. ◆ The same identity is used with all services providers, resulting in linkability issues.
Federated	◆ No need to have different identifiers and credentials for each service or resource. ◆ No need to re-authenticate when moving from one service to another within a federated domain. ◆ Users cannot authenticate using fake identities and credentials.	◆ In case of unexpected identity theft, the damages are worse as the attack can extend to the whole IDM system, resulting in impersonation attacks. ◆ Users' privacy is threatened as their personal information are commonly shared among service providers and their transactions are tracked. ◆ Users are not able to select attributes to be disclosed as they are managed by service providers.
User-centric	◆ Users are given the full control over their identities and are able to select the information to be disclosed. ◆ Users can remain anonymous and unlinkable among both identity and service providers. ◆ Users can use different identities ans credentials at different service providers.	◆ Impersonation attacks are possible if no authentication is required between the user and the service provider. ◆ Users can use fake identities and attributes if no rigorous control is applied. ◆ In case of anonymity without revocation, illegitimate transactions cannot be traced back to their origin.

In the centralized model, the user is not required to memorize multiple identifiers. He possesses one single credential that is used when authenticating at different service providers. This credential cannot be forged to authenticate to service providers, unless the forgery is detected. Moreover, some centralized identity-based solutions offers pseudonymous authentication to users (e.g., .Net Passport of Microsoft) [18].
However, the identity provider can be considered as a single point of failure rendering the system vulnerable to Denial of Service (DoS) attacks. Additionally, as the user is uniquely identified inside the IDM system, many privacy threats may raise. For instance, a centralized identity provider is able to track users' transactions among several service providers and build full users' profiles. Service providers are also able to link several transactions of the same user.

For the federated identity model, an improved user experience is provided as only one

credential is required inside a federated domain. As identities are managed by service providers, users are not allowed to authenticate using fake identities and credentials. Nevertheless, impersonation attacks are possible due to identity theft performed against identity providers. Indeed, once a member of the federation is affected by an identity theft, the damages are worse, as the attack can extend to the whole IDM system. Furthermore, members of the same federated domain are able to track users' transactions and to build users' profiles inside the domain. The selective disclosure property is also not provided by the federated model.

The user-centric model gives users the full control over their identities. It comes to alleviate some privacy issues. Indeed, users are able to select data to be disclosed and to know any type of analysis performed on his collected data. Moreover, they can remain anonymous and use different identities and credentials at different service providers. As such, neither service providers nor identity providers are able to link their transactions to each other or identified them.
However, in case of no authentication required between the user and service providers, impersonation attacks are possible by malicious outsider, e.g., replaying previously generated credentials. Fake identities can be also used by malicious users, unless rigorous control is applied. Finally, for some solutions, where users' anonymity cannot be revoked, illegitimate transactions cannot be traced back to their origin.

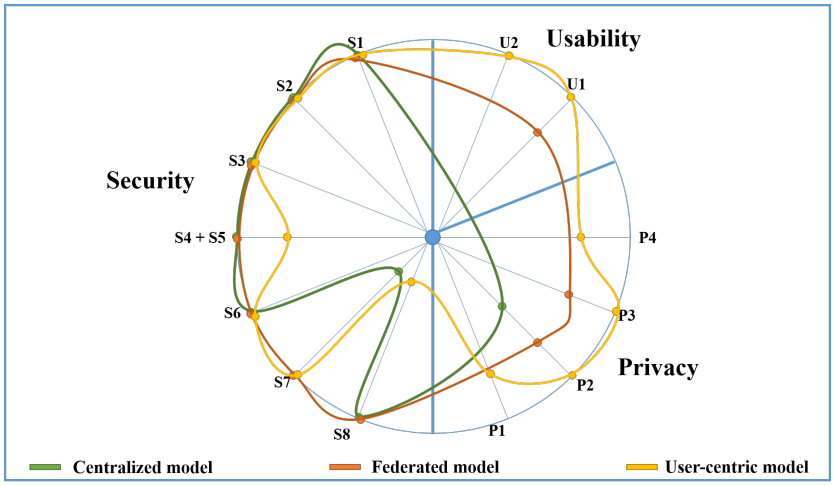


Figure 2.4: Spider Diagram - Properties According to Identity Models

NOTE: The codification of properties used in the figure is based on Section 2.2.2

Discussion based on Satisfied Properties Taking into consideration security, privacy and usability properties, as defined in Section 2.2.2, we illustrate in Figure 2.4, the properties that are supported by the different identity management model.

In terms of security, the Figure shows that all the identity models support the main properties. However, according to systems' designs and constructions, some properties could not be achieved. For instance, for the centralized model, the vulnerability to the DoS attack affects the availability property. Moreover, w.r.t. the system' design and

concrete construction and the underlying cryptographic technology, the non-repudiation property may not be achieved by some user-centric IDM systems [34].

When it comes to privacy, except from pseudonymity, other properties are not satisfied by the centralized model. Indeed, users are uniquely identified and can be tracked among several service providers by the centralized identity provider. The federated model, as it is currently well-used, has evolved to be compliant with new regulations like the GDPR, thus supporting some privacy requirements like pseudonymity and data minimization. The user-centric model seems to be the most promising one to support the desired requirements as the user gains the core role in the identity management model. Indeed, when dealing with their attributes and credentials, users are able to remain anonymous/pseudonymous towards service providers. They can also prevent providers from linking their transactions. These privacy properties can be ensured w.r.t. the user-centric system design and the cryptographic tools implemented.

Finally, in terms of usability, although the federated model can support the concept of users' multiple identities, the user-centric identity seems to be the best model to ensure both IDM usability requirements, helping users to deal either with their multiple identities or with one identity and multiple attributes.

According to this comparison, we can deduce that the user-centric identity is the model that is able to cover all requirements of identity management systems. Thus, in the next section, we will explore the different categories that can derive from the user-centric model.

2.2.3.5 User-centric Identity Model: A Promising Privacy-preserving Approach

In this section, we put emphasis on the user-centric model as a promising approach for privacy-preserving identity management systems. For this purpose, a user-centric identity management taxonomy is provided in Figure 2.5.

A user-centric identity management system can be centralized or decentralized. On the one hand, in a centralized user-centric identity management system, the governance role is given to a centralized entity, i.e., the identity provider is responsible for establishing the identity management policies. These policies include the location of the identities' storage, the user's capabilities to manipulate his credentials, privacy policies, etc. On the other hand, in a decentralized user-centric identity management system, the governance is distributed among different entities, (e.g., multiple identity providers, a consortium of institutions, etc.). The identity management policies are defined through an agreement established between the involved entities. Generally, the decentralized category is based on peer-to-peer architecture-based solutions, e.g., blockchain, InterPlanetary File System (IPFS)[9].

For the centralized category, different schemes have been proposed and adopted. From a privacy perspective, centralized user-centric identity management schemes can be categorized into three blocks, namely non privacy-preserving, pseudonymous and

[9] https://docs.ipfs.tech/concepts/what-is-ipfs/#participation

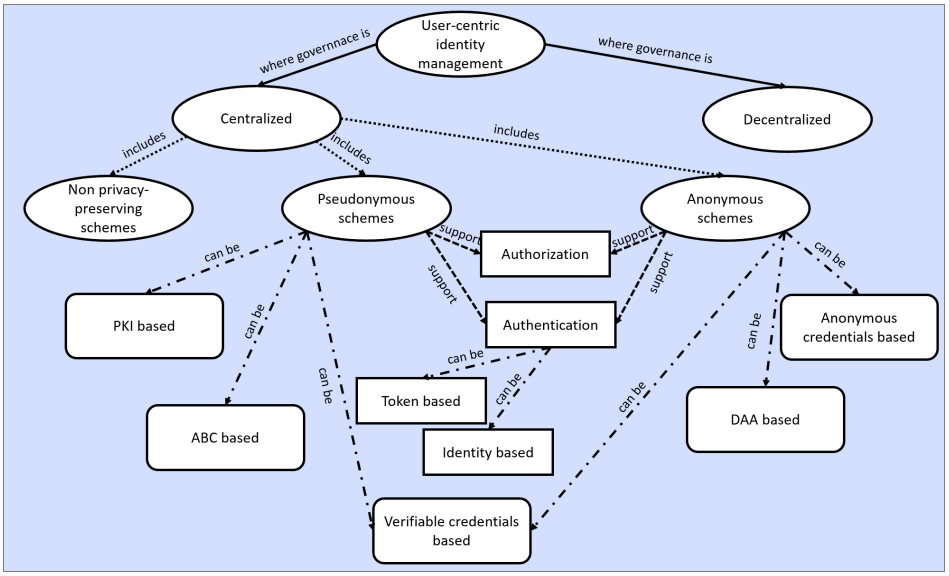


Figure 2.5: Taxonomy of User-centric Identity Management Systems

anonymous schemes. Indeed, non privacy-preserving schemes have been designed for only affording users the control over their identities. Users' activities can be traced by different entities (i.e., the identity provider and service providers). Hence, pseudonymous and anonymous schemes have been designed in order to satisfy privacy requirements, in compliance with regulations. For example, pseudonymous schemes allow users to interact with service providers using different pseudonyms. A user can even choose to use different pseudonyms with the same service provider for linkability concerns. As such, users' transactions remain unlinkable at the same service provider. Unlike anonymous schemes, for some pseudonymous schemes, the identity provider is able to trace back a user's pseudonymous activity to his identity [108, 30].

Both pseudonymous and anonymous schemes support authorization and authentication properties. For user-centric privacy-preserving identity management, two authentication factors are considered, namely the possession and the inherence categories[10], referring respectively to something the user has (e.g., token, identification device, etc.) and something the user is (e.g., biometric characteristics). That is, the authentication can be either token-based or identity-based.
On the one hand, token-based authentication belongs to the possession category. It has been mainly used to log in to websites that support the OpenID solution as a user-centric framework [119]. A user-centric identity management system that supports token-based authentication, ensures the data origin authentication property.
On the other hand, identity-based authentication belongs to the inherence category and allows to verify that the user is who he claims to be, before giving him access to services and resources. Thus, a user-centric identity management system that relies on an identity-based authentication, satisfies the peer-entity authentication property. Biometric identities are the most used ones for identity-based authentication. Indeed,

[10]The knowledge category is often used in the centralized or the federated model where the user can be uniquely identified through something he knows (e.g., a password, identification number, etc.

according to recent statistics carried out by iProov [71], more than two-thirds of people prefer using biometrics as they are easier and faster and they offer secure and robust identity verification.

To satisfy the authorization and authentication requirements, different mechanisms have been proposed designing pseudonymous and anonymous schemes.
On the one hand, pseudonymous schemes can be represented by Public Key Infrastructure-based systems [21, 53] or Attribute-based (ABC) [30, 108] or Verifiable Credentials (VC). The concept of PKI-based user-centric identity management allows to avoid the use of a centralized certification authority. Indeed, users obtain auto-signed public key certificate over their public keys, from local registration authorities (i.e., no hierarchy is established between registration authorities). Certificates are published at a central electronic notary. As such, the user is able to use his keys to authenticate or to encrypt messages with no need to a certification authority (i.e., the electronic notary). In Attribute-based credentials, users obtain credentials over their attributes, from an identity provider known as the issuer, and then prove the possession of these credentials, inside presentation tokens, to service providers known as verifiers. Verifiable credentials (VC) will be discussed later in this section.
On the other hand, among anonymous schemes, we can find Direct Anonymous Attestation (DAA) [22, 8], Anonymous Credentials (AC) [78, 34] and also Verifiable Credentials. Indeed, DAA is a digital signature scheme where different secret keys are used for signing and one common public key is used for verification. In the literature, anonymous credentials have been used to describe the same concept as attribute-based credentials. However, in this taxonomy, we differentiate between them according to the level of anonymity that can be ensured. Indeed, attribute-based credentials were initially designed to support pseudonymity [16] with possible revocation of the user's anonymity. For anonymous credentials, we consider that the user's identity cannot be retrieved, neither by services providers nor by the identity provider.

Verifiable credentials is a common mechanism used for both anonymous and pseudonymous schemes. The concept of verifiable credentials covers all the user-centric identity management solutions that allow service providers to verify the validity and the integrity of attributes provided by users. In verifiable credentials, attributes include any trait or information belonging to a user. Attributes can be processed *(i)* as they were originally provided by an $\mathcal{IP}$ or an $\mathcal{AP}$), *(ii)* through a generalized form (e.g., for a date of birth 09/12/2000, the generalized form can be either 12/2000 or 2000) or *iii* through a randomized form (i.e., if randomization does not impact the original value of the attribute). The level of anonymity supported by verifiable credentials depends on the cryptographic tools used to build the identity system. To enhance users' privacy, several cryptographic tools emerged, referred to as Privacy Enhancing Technologies (PETs).

In the next section, we give detailed definitions of the commonly used PETs. A big focus is given to PETs that support malleability features, referred to as malleable PETs.

2.3 Malleable Privacy Enhancing Technologies

Privacy-enhancing technologies (PETs) offer a range of technical solutions constituting strong tools to design privacy-preserving applications. Using these tools, the designed applications aim at satisfying security, privacy and functional requirements at once. PETs include some cryptographic techniques, namely digital signatures and encryption technologies. In this section, we focus on malleable cryptographic technologies, i.e., signature and encryption schemes that offer malleability features like randomization, modification, etc.

2.3.1 Digital Signatures

Digital signatures have been used in identity management systems for many purposes, namely for authenticating the source of data, e.g., attributes, and identifiers belonging to a user, and ensuring data authenticity and integrity.

2.3.1.1 Group Signatures

Group signatures were first introduced by Chaum and van Heyst [42]. The fundamental principle of group signatures is depicted in Figure 2.6.

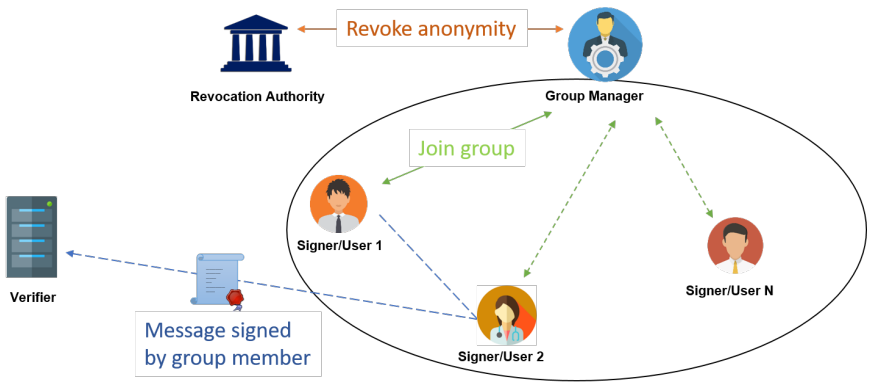


Figure 2.6: Group Signature - Interaction between Entities

Group signatures enable any group member (referred to as a user) to sign a message on behalf of the group, while remaining anonymous. As such, verifiers authenticate the user as member of the group, but are not able to identify him, i.e., peer-entity authentication. Each group is composed of a group manager and members (i.e., users), and is characterized by a group private key with which the message is signed, and the verifier checks the validity of the signature using the corresponding group public key. Group signature scheme may include a optional entity referred to as a revocation authority responsible for revoking the anonymity of users.

Definition 1. Group Signature – A group signature scheme relies on five probabilistic polynomial-time (PPT) algorithms referred to as Setup, Join, Sign, Verify and Open algorithms, as depicted in Table 2.2.

Table 2.2: Group signature - Algorithms and Features

Algorithm	Objective	Running entities	Inputs	Outputs
Setup	Set-up the group	Group Manager	A security parameter λ	The master secret key `gsk` and the group public key `gpk`
Join	Join the group by a user of index i	Group manager and Signer/User	The group manager secret key `gsk` and the group public key `gpk`	Signer's keys `key`$[i]$
Sign	Sign a message on behalf of the group	Signer/User	The group public key `gpk`, the signer's key `key`$[i]$ and a message m	A group signature σ
Verify	Verify the group signature	Verifier	The group public key `gpk`, the message m and the signature σ	$b \in \{0, 1\}$
Open	Identify the signer	Revocation Authority	The master secret key `gsk`, the message m and the signature σ	The index i of the signer

Static group signature schemes [10, 14] have been developed, where the number of group members is fixed, by the group manager, at the creation of the group. Later, dynamic group signatures [15, 80, 51] have been proposed in order to meet scalability requirements. Users are able to select their secret keys and ask the group manager to certify the corresponding public key in order to join the group.

Both variants of group signature schemes is assumed to support the following properties in terms of privacy and security:

- *Anonymity*: introduced by [42], states that it is not possible to recover the signer's identity from her signature. This property also covers the incapacity to link two or several signatures issued by the same user.
- *Traceability*: proposed by [42], ensures that an adversary is not able to forge a signature without being traced.
- *Unforgeability*: introduced by [9], guarantees that it is not possible to produce a valid message/signature pair unless a valid secret signing key is known.
- *Non-frameability*: proposed by [44], avoids that a group of users (either members or non members of the group) are able to generate a signature that traces back to an innocent member.

Note that the anonymity of signers is revocable by a designated trusted party (i.e., the revocation authority).

2.3.1.2 Sanitizable Signatures

A signature is said to be sanitizable if it is possible, through the knowledge of the appropriate secret keys, to efficiently derive, from a signature σ on a message

m=(m_1,m_2,...,m_n) (i.e., n is the number of blocks of the message m), a signature σ' on a message m'=(m'_1,m'_2,...,m'_n), where m' is an admissible transformation of m [41]. Sanitizable signatures, have been introduced by Ateniese *et al.* [11] allowing a designated party, called the sanitizer, to change some parts, chosen by the signer, of a signed message while the corresponding signature remains valid under the signer's key and the authenticity of the message m' is guaranteed. Figure 2.7 presents an example workflow of a sanitizable signature, where the message m is set to (P,E,T). According to the admissible modifications $adm = \{\{2\}, 21\}$, the sanitizer is able to modify the second block of the message with the letter of index 21 in the alphabet, i.e., the letter U. Thus, the output message is (P,U,T).

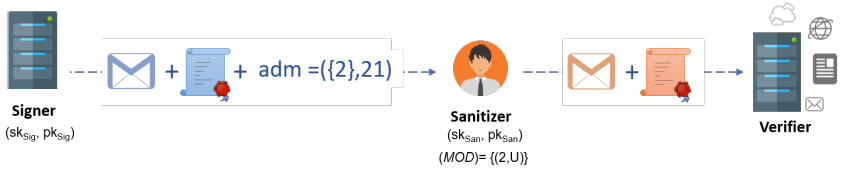


Figure 2.7: Sanitizable Signature - Example Workflow with a Message m Set to (P,E,T) and Sanitized as m'=(P,U,T)

Definition 2. Sanitizable Signature – A sanitizable signature scheme involves six main PPT algorithms referred to as Setup, Sig_KeyGen, San_KeyGen, Sign, Sanit, Verify and two optional ones, namely Proof and Judge, depicted in Table 2.3.

Table 2.3: Sanitizable signature - Algorithms and Features

Algorithm	Objective	Running entity	Inputs	Outputs
Setup	Set-up the system	Trusted Authority	A security parameter λ	The public parameters pp
Sig_KeyGen	Generate the signer's keys	Trusted Authority	The public parameters pp	The signer's pair of keys ($\mathtt{sk}_{sig}$,$\mathtt{pk}_{sig}$)
San_KeyGen	Generate the sanitizer's keys	Trusted Authority	The public parameters pp	The sanitizer's pair of keys ($\mathtt{sk}_{san}$,$\mathtt{pk}_{san}$)
Sign	Sign a message	Signer	A message m, the signer's secret key $\mathtt{sk}_{sig}$, the sanitizer's public key $\mathtt{pk}_{san}$ and admissible modifications adm	A signature σ
Sanit	Sanitize a message	Sanitizer	The message m, the signature σ, the modifications MOD, the signer's public key $\mathtt{pk}_{sig}$, the sanitizer's secret key $\mathtt{sk}_{san}$	Sanitized message m' and signature σ'
Verify	Verify a signature	Verifier	The message m, the signature σ, the signer's public key $\mathtt{pk}_{sig}$ and the sanitizer's public key $\mathtt{pk}_{san}$	$b \in \{0,1\}$
Proof	Generate a proof over the message/signature pair	Signer	The signer's secret key $\mathtt{sk}_{sig}$, the message m, the signature σ and the sanitizer's public key $\mathtt{pk}_{san}$	A proof $\pi \in \{0,1\}^*$
Judge	Decide which party is accountable for the message/signature pair	Anyone	The message m, the signature σ, the signer's public key $\mathtt{pk}_{sig}$, the sanitizer's public key $\mathtt{pk}_{san}$ and the proof π	A decision $d \in \{Sig, San\}$

Ateniese *et al.* [11] introduced five security requirements that have been later formalized by Brzuska *et al.* [25]. Additional properties of sanitizable signatures have

been suggested and formalized in [61, 26, 85, 23, 27]. We list hereafter the required properties of sanitizable signatures, i.e., unforgeability and immutability and the desired ones, i.e., privacy, transparency, accountability, unlinkability and invisibility.

- *Unforgeability*: states that a sanitizer which neither holds the appropriate keys nor has been authorized by the signer, is not able to produce valid message/signature pair.
- *Immutability*: ensures that sanitizers can perform only admissible modifications over the message received from the signer.
- *Privacy*: guarantees that any party not holding the appropriate private keys is not able to derive any information about sanitized parts of a message.
- *Transparency*: states that any party not holding the appropriate private keys is not able to decide whether a signature represents an original version or a sanitized one. This property is even stronger than privacy.
- *Accountability*: requires that when a proof is generated over a message/signature pair, it should not point to the wrong party. Proofs generated by the signer points to the accountable party.
- *Unlinkability*: guarantees that a sanitized signature cannot be linked to the original one, which implies that two or several sanitized signatures cannot be linked together.
- *Invisibility*: states that any party not holding the appropriate private keys is not able to decide which parts of the message are admissible.

Additional features have been later proposed for sanitizable signatures. Indeed, In [33], Canard *et al.* propose an extension to sanitizable signatures supporting multi signers and sanitizers. In order to avoid the sanitizer selection at the signature generation phase, Samelin and Slamanig propose, in [125], a policy-based sanitizable signature scheme that allows a sanitizer to modify a signature based on a policy over a set of attributes it has. The proposed signature scheme does not support the unlinkability property. In [20], Bossuat and Bultel present an extension of sanitizable signature that allows to fix not only the admissible blocks, but also the number of admissible blocks that can be sanitized in a single sanitization.

2.3.1.3 Redactable Signatures

Redactable signatures have been introduced by Steinfeld *et al.* [134] allowing any party to remove certain parts of a signed message without interacting with the original signer. The new message remains authentic and the corresponding signature is kept valid under the signer's key. An example of redaction is illustrated in Figure 2.8.

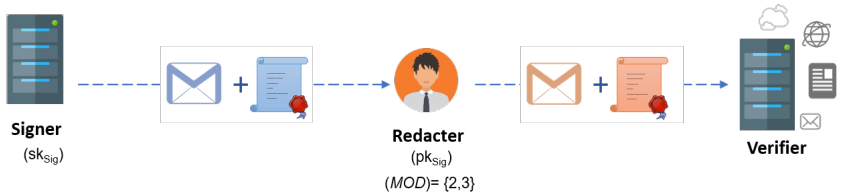


Figure 2.8: Redactable Signature - Example Workflow with a Message m Set to (I,do,not,like,pets) and Redacted as m'=(I,⊥,⊥,like,pets), (I,⊥,like,pets) or (I,like,pets)

Table 2.4: Redactable signature - Algorithms and Features

Algorithm	Objective	Running entity	Inputs	Outputs
Setup	Set-up the system	Trusted Authority	A security parameter λ	The public parameters pp
Sig_KeyGen	Generate the signer's keys	Trusted Authority	The public parameters pp	The signer's pair of keys ($\mathtt{sk}_{sig}$,$\mathtt{pk}_{sig}$)
Sign	Sign a message	Signer	A message m, the signer's secret key $\mathtt{sk}_{sig}$ and admissible modifications adm	A signature σ and redaction information red
Redact	Redact a message	Redacter	The message m, the signature σ, the modifications MOD, the signer's public key $\mathtt{pk}_{sig}$ and the redaction information red	Redacted message m' and signature σ' and modified redaction information red'
Verify	Verify a signature	Verifier	The message m, the signature σ and the signer's public key $\mathtt{pk}_{sig}$	$b \in \{0,1\}$

Definition 3. Redactable Signature – A redactable signature scheme involves five PPT algorithms referred to as Setup, Sig_KeyGen, Sign, Redact and Verify presented in Table 2.4.

In [24], Brzuska *et al.* formalize the security properties supported by redactable signatures that include unforgeability, privacy and transparency. Camenisch *et al.* [31] introduce a redactable signature scheme fulfilling the unlinkability property. Accountability is also taken into consideration, in [116], by inheriting some primitives of sanitizable signature scheme (i.e., Proof and Judge primitives). We give below definitions of the required (i.e., unforgeability) and desired properties (i.e., privacy, transparency, unlinkability and accountability).

- *Unforgeability*: states that a party not holding any secret key is only able to derive signatures for messages by redaction and not to produce new valid ones.
- *Privacy*: guarantees that any party not holding any private keys is not able to derive any information about redacted parts of a message.
- *Transparency*: states that any party not holding any private keys is not able to decide whether a signature represents an original version or a redacted one. This property is even stronger than privacy.
- *Unlinkability*: as for sanitizable signatures, guarantees that a redacted signature cannot be linked to the original one and implies that two or several redacted

signatures cannot be linked together.

- *Accountability*: states that a trusted party (other than the signer and the verifier) is able to trace redacters based on their keys.

2.3.2 Encryption Technologies

In the digital era, encryption presents an essential cryptographic mechanism used to secure data. The primary purpose of such a mechanism is to ensure the confidentiality of data transmitted over the network as well as data stored on computer systems. Several advanced encryption technologies have been studied so far, including homomorphic and polymorphic encryption schemes. These schemes allow entities to perform computations in the encrypted domain, in order to protect the privacy of sensitive data, e.g., predictive analytics can be performed by cloud servers over encrypted sensitive medical data without compromising their privacy.

2.3.2.1 Homomorphic Encryption

Homomorphic encryption technologies provide a special form of encryption. They state that third parties are able to apply some functions on an encrypted data in a blind manner (i.e., without getting access to the original content of the ciphertext), while the ciphertext can be decrypted by a single key (i.e., the user's secret key). The basic principle of homomorphic encryption schemes is illustrated in Figure 2.9.

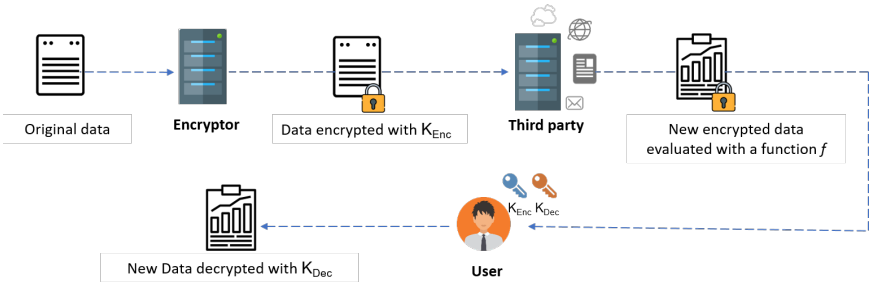


Figure 2.9: Homomorphic Encryption - Interaction between Entities

Definition 4. Homomorphic Encryption – An homomorphic encryption scheme involves four PPT algorithms referred to as KeyGen, Encrypt, Decrypt and Evaluate presented in Table 2.5.

Homomorphic encryption technologies involve three methods referred to as, partially, somewhat and fully homomorphic encryption. Indeed, Partially Homomorphic Encryption (PHE) includes homomorphic encryption schemes that support a single operation, i.e., addition or multiplication. The same operation can be performed on a ciphertext several times. Well known encryption schemes support partially homomorphic encryption features namely RSA [121] and ElGamal [55] for multiplication

Table 2.5: Homomorphic encryption - Algorithms and Features

Algorithm	Objective	Running entity	Inputs	Outputs
KeyGen	Generate the encryption/decryption keys	Trusted Authority	A security parameter λ	The pair of encryption/decryption keys ($\mathtt{K}_{enc}$,$\mathtt{K}_{dec}$)
Encrypt	Encrypt a plaintext	Anyone	A plaintext m and the encryption key $\mathtt{K}_{enc}$	A ciphertext c
Decrypt	Decrypt a ciphertext	Decryption key holder	The ciphertext c and the decryption key $\mathtt{K}_{dec}$	The plaintext m
Evaluate	Evaluate a ciphertext	Third party	The ciphertext c and a function $f()$	A new ciphertext c'

and Paillier [107] for addition. Somewhat Homomorphic Encryption (SHE) refers to homomorphic encryption schemes where a limited number of different operations can be performed on ciphertexts [50, 19]. Finally, Fully Homomorphic Encryption combines the advantages of the two previous types. It involves encryption schemes supporting an unbound number of operations, mainly addition and multiplication, for an unlimited number of times [122, 130].

The concept of homomorphic encryption has been applied to several applications namely cloud-computing [98, 114, 43], healthcare systems [101] and biometric identification and verification [63, 58]. The reasons behind the use of homomorphic encryption technologies are explained as follows:

- *Availability of data* – enables to share data through multiple third parties without compromising its security.
- *Data confidentiality* – states that third parties are able to process users' data without getting access to their content unless decryption key is known.

2.3.2.2 Polymorphic Encryption

Polymorphic encryption is a novel type for encrypting data. It reproduces the same idea of homomorphic encryption, however, the key difference is that the ciphertext can be decrypted by multiple keys. Indeed, the main principle of polymorphic encryption is depicted in Figure 2.10, where an encryptor encrypts a plaintext data using an encryption key K_{enc}, and transfers the ciphertext data to an intermediate party called blind transcryptor. The transcryptor blindly transforms the encrypted data and designates who can decrypt it while re-keying the ciphertext.

Definition 5. Polymorphic Encryption – A polymorphic encryption scheme relies on five PPT algorithms referred to as MasterKeyGen, KeyGen, Encrypt, Transform and Decrypt illustrated in Table 2.6.

Polymorphic encryption schemes support the same properties of homomorphic ones namely availability and confidentiality of data. The use of polymorphism provides additional functional features presented as follows:

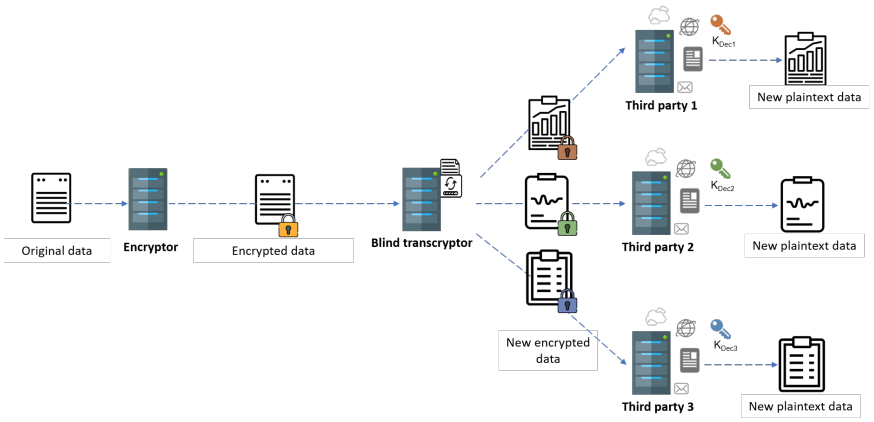


Figure 2.10: Polymorphic Encryption - Interaction between Entities

Table 2.6: Polymorphic encryption - Algorithms and Features

Algorithm	Objective	Running Entity	Inputs	Outputs
MasterKeyGen	Generate the master secret and public keys	Trusted Authority	A security parameter λ	The pair of master secret and public keys (`msk`,`mpk`)
KeyGen	Generate a pair of secret and public keys for a party i	Trusted Authority	The master secret key `msk`	The pair of secret and public keys of the party i (`sk`,`pk`)
Encrypt	Encrypt a plaintext	Anyone	A plaintext m and the master public key `mpk`	A ciphertext c
Transform	Transform a ciphertext	Blind transcryptor	The ciphertext c and two functions $f_1()$ and $f_2()^a$	A new ciphertext c'
Decrypt	Decrypt a ciphertext	A party i	The ciphertext c' and the secret key `sk` of party i	A new plaintext m'

Note: ^{a} indicates that the functions $f_1()$ and $f_2()$ allow to transform the ciphertext content and to re-key it (i.e., to make the new ciphertext decipherable by party i), respectively.

- Data can be encrypted, without prior determination of who can decrypt.
- At any later stage, data can be made decipherable, by any party of a system using blind transcription.

The concept of polymorphism has been applied on the ElGamal cryptosystem to manage sensitive personal data that are especially used in healthcare [142]. The ElGamal cryptosystem supports polymorphic properties, namely re-randomization, re-shuffling and re-keying defined as follows.

- Re-randomization – allows to change the appearance of a ciphertext that remains decipherable by the original master secret key.
- Re-shuffling – allows to change the content of the ciphertext such that when being decrypted with the original master secret key, a new plaintext is obtained (i.e., different from the original plaintext data).

- Re-keying – allows to make the ciphertext decipherable by a new secret key (i.e., derived from the master secret key). The decryption results in the original plaintext data.

Note that two or three properties can be combined to ensure strong security and privacy properties.

2.3.3 Evaluation of Malleable PETs and their Application to Identity Management Systems

This section first provides a comparison between the reviewed malleable PETs in terms of advantages and limits, as illustrated in Table 2.7. Then, it gives in Section 2.3.3.2, examples of malleable PETs-based identity management systems.

2.3.3.1 Advantages and Drawbacks based Discussion

It is worth mentioning that malleability features supported by the reviewed PETs alter either the content of the signed/encrypted data (i.e., sanitizable and redactable signatures, homomorphic and polymorphic encryption) or its form/presentation (i.e., group signatures and polymorphic encryption).

On the one hand, sanitizable and redactable signatures allow to modify or remove parts of a message, thus the output message is different from the original one. As such, the data minimization property is satisfied while providing only the minimum amount of data.
Homomorphic encryption schemes allow third parties to extract relevant information from encrypted and evaluated data thanks to Evaluate primitive. Thus, original sensitive data is kept private from third parties. Hence, the confidentiality of data is preserved at third parties, and cannot be compromised even in case of data breaches on third parties' systems.

On the other hand, group signatures do not allow to modify the content of the signed data, however different presentations of the same signed data can be provided as group members often generate zero-knowledge proofs over signed messages [56, 148]. As such, users are able to prove the possession of secret information without disclosing them, which refers to satisfying the data minimisation principle. Additionally, different presentations of the same data can remain unlinkable among different entities.
Polymorphic encryption schemes provide malleability over both content and presentation of data. In fact, different parties are able to extract relevant information from encrypted data while the confidentiality of the original data is preserved. Also, having access to different presentations of the same data, different parties are not able to link them.

In spite of the aforementioned advantages, it is noticeable from Table 2.7 that malleable PETs present some limits according to their designs which raise privacy and security issues. First, regarding group signatures, the anonymity property is limited

Table 2.7: Comparison between Malleable Privacy Enhancing Technologies

Technology	Advantages	Drawbacks
Group signatures	Individuals are able to authenticate without disclosing their identities.	Signers' anonymity can be revoked by a revocation authority thus weakening the anonymity property.
Sanitizable signatures	Individuals are able to modify certified messages while certificates remain valid.	The original message can be reconstructed relying on several sanitizations by the same sanitizer.
Redactable signatures	Individuals are able to remove parts of certified messages while certificates remain valid.	Redaction can be performed by any party thus traceability is not possible.
Homomorphic encryption	Relevant information can be extracted from ciphertexts while data privacy is ensured.	The encryption key is the same used through different parties thus presenting linkability threats.
Polymorphic encryption	Sensitive data can be shared with different parties without harming its security and privacy.	The transcryptor represents a single point of failure in polymorhpic encryption schemes.

to a revocation-restricted anonymity since a revocation authority is able to identify the signer. Second, relying on several sanitizations of the same message performed by the same sanitizer, the original message can be reconstructed. Third, for redactable signatures, the redaction can be performed by anyone, such that the redactor cannot be traced in case of illegitimate transaction. Additionally, for homomorphic encryption, the encryption key is often known by third parties that performs computation in the encrypted domain. Thus, being revealed to different parties, this key enable them to link encrypted data of the same user.

Finally, the concept of polymorphic encryption introduces security issues. Indeed, the transcryptor is the central party responsible for performing all transformations, thus presenting a single point of failure.

2.3.3.2 Malleable PETs for Identity Management

Malleable PETs have been used extensively to design privacy-preserving identity management systems like group signatures [30, 73, 148], sanitizable signatures [45, 34], and redactable signatures mainly for blockchain-based solutions [144, 83]. Indeed, they have been applied in anonymous credentials systems. That is, a user is able to anonymously prove to a service provider the possession of certified attributes. Two categories of the reviewed malleable PETs have been used to design concrete anonymous credentials systems, namely group and sanitizable signatures. A well-known example of anonymous credentials systems, based on a group signature variant, is called Idemix [30]. This industrialized solution suggests that an issuer (i.e., identity provider) generates a signature over the user's attributes. Afterwards, to anonymously prove to a service provider the possession of a particular attribute, the user, being a member of a group, generates a zero-knowledge proof to show that he owns a valid signature over the requested attribute(s). The same concept has been proposed in [34] relying on a sanitizable signature scheme. Indeed, the issuer provides a sanitizable signature to a designed user that is able to modify parts of the signed message (i.e., w.r.t. to a set of admissible

modifications) and accordingly adjusts the signature. This concept allows the user to disclose only the requested attributes to a service provider.

Different applications that deal with users identities, attributes and sensitive data, have also relied on malleable PETs. We list hereafter some examples:

- Electronic voting systems – To ensure the integrity of votes and the anonymity of voters, group signatures have been used to design privacy-preserving electronic voting systems [92], where a voter can anonymously poll votes.
- Biometric authentication systems – To protect users' sensitive biometric data from abuse, biometric authentication schemes have been developed relying on homomorphic encryption [63, 68]. As computations can be performed, on encrypted biometric data, third parties are able to authenticate users without compromising their privacy.
- Recommendation systems – To preserve users' privacy, homomorphic encryption schemes have be used in to support personalized services through recommendation systems [81, 106]. In fact, instead of analyzing user's private data (e.g., ratings, items, categories, etc), recommenders rely on computation over encrypted data and homomorphic properties to retrieve relevant information without compromising data confidentiality.
- E-healthcare systems – To offer privacy-preserving personalized services, e-healthcare systems have been built upon polymorphic encryption schemes [142]. They allow to protect sensitive medical data and to use different users' pseudonyms at different parties thanks to the concept of polymorphic pseudonymization.

2.4 Conclusion

In this first chapter, we present a general introduction to identity management systems and we highlight the different issues they raise in terms of security, privacy and usability. According to a comparison between identity models, we select the user-centric identity as a promising model to achieve a trade-off between security, privacy and usability. Then, we investigate the usage of privacy-enhancing technologies in identity management systems allowing to satisfy a variety of properties. As a specific variant of PETs, we focus on malleable PETs, thanks to their attractive features. For instance, apart from the fulfilled security properties, they provide malleability over either the content of the data or its presentation.

In the next chapter, we present a first solution that allows to ensure the trade-off between security, privacy and usability, when accessing services. Indeed, we propose a privacy-preserving identity management system, called Pima, based on unlinkable malleable signatures. Indeed, users receive a malleable signature over their attributes from an identity provider. As such, they can remove some attributes when requesting service providers, through pseudonymous sessions.

Chapter 3

A Privacy-preserving Identity Management System based on an Unlinkable Malleable Signature

And loss of control is always the source of fear. It is also, however, always the source of change.

– James Frey

Referring to the travelling experience described in the Introduction, we propose, in this chapter, a solution for Alice that allows her to have the full control over her flight information, to disclose only the required data and to remain pseudonymous at different service providers. Indeed, the proposed solution allows the airline company to deliver a pseudonym and a certified and malleable boarding pass to Alice that can be used in different contexts with different service providers. For instance, when buying goods at the airport shops, Alice hides all identifiable information on the boarding pass. She only needs to prove that she is travelling to an international destination on that day to benefit from the 20% VAT charge exoneration. Additionally, to connect to the airport WIFI, Alice can prove to the Internet provider, from information given in her boarding pass, that she is a traveler at the airport. Two different pseudonyms can be used with the airport shop and the Internet provider. As such, Alice can benefit from the two services while remaining non identifiable and unlinkable between the two service providers.

3.1 Introduction

In order to be authorized to access services and resources, privacy-preserving identity management systems have been proposed with the objective to empower users with the full control over their identities and personal data while preserving their privacy as well.

Sanitizable signatures are promising candidates of interest for privacy-preserving identity management systems. They have been implemented in identity management systems, for the first time, by Canard and Lescuyer [34]. They allow users to modify, in a controlled way, their certified attributes' values while certificates remain valid. Nevertheless, in existing sanitizable signature-based identity management solutions, attributes' names remain accessible to service providers while only attributes' values, that users do not want to disclose, are hidden (e.g., by special characters). As such, service providers are given the ability to distinguish modified attributes and to decide whether a message has been modified. Unfortunately, these solutions do not match our needs to disclose only the necessary attributes and attributes' values and also to achieve a controlled level of linkability. Moreover, further security and privacy properties need to be addressed when empowering users with the full control over their identities and attributes, as defined below:

- **Controlled and restricted modifications** – users should not be allowed to neither perform arbitrary modifications nor permutations on attributes' values in the same message.

- **Privacy** – when having access to a modified signature, no party is able to determine whether the message was modified or not, which and how many parts have been modified. As a result, profiling possibilities are mitigated[1].

To meet all the above challenges, we present our first contribution [94], named Pima. With respect to the taxonomy presented in Chapter 2, Section 2.2.3.5, this contribution is considered as a pseudonymous centralized user-centric identity management solution, supporting authorization, and belonging to the verifiable credentials category (cf. Figure 3.1).

Indeed, we propose a user-centric identity management system, based on a novel unlinkable malleable signature ($\mathcal{UMS}$) that satisfies both sanitizable and redactable signature schemes' features. With the combination of both primitives, we aim to enable a designated sanitizer to remove admissible parts of a message in a controlled way (i.e., he can neither modify admissible blocks with arbitrary choices nor exchange attributes' values of admissible blocks), while his public key is required for verification. For this purpose, we propose that each attribute may have several values. The sanitizer is able

[1]If modified blocks are known, a service provider relying on several sessions initialized by the same user, is able to combine several modified versions of the same original message and to retrieve the accurate profile.

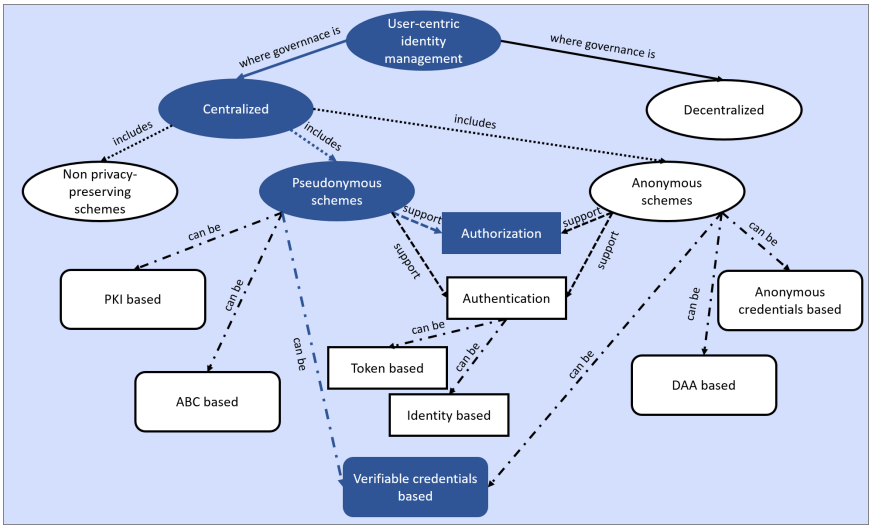


Figure 3.1: PIMA Features w.r.t. the Taxonomy of User-centric Identity Management Systems

to choose one of the values of the attribute to be disclosed and remove the others (i.e, attributes and attributes' values).

In the perspective of designing a malleable signature as a main single building block of the proposed system, the signer's role is assigned to the identity provider ($\mathcal{IP}$) and the sanitizer's role to the user ($\mathcal{U}$). $\mathcal{U}$ receives from $\mathcal{IP}$ a signed message certifying his identity (i.e., attributes), associated with a pseudonym. To authenticate with a service provider ($\mathcal{SP}$), for a particular session, $\mathcal{U}$ is able to generate a randomized pseudonym and modify his attributes w.r.t. admissible modifications and/or remove the unnecessary ones. $\mathcal{SP}$ can then verify the received information, relying on the public parameters of the identity provider, considered as trusted, thus ensuring the data origin authentication property **(S5)** (cf. Chapter 2, Section 2.2.2.1).
Let us illustrate with the example of the traveler Alice who is travelling from France to the United States. In the boarding pass, the airline company provides Alice with a pseudonym 'ICE' and a signature σ on her attributes (i.e., her information and the flight's details) constituting the message $m = \{a_1 =$ 'Name: Alice', $a_2 =$ 'Company Name:Air France', $a_3 =$ 'Flight Number:AF084', $a_4 =$ 'Date of flight:09/12/2020', $a_5 =$ 'Time:10:40', $a_6 =$ 'Destination:United States', $a_7 =$ 'Destination:International', $a_8 =$ 'Seat:28F', $a_9 =$ 'Gate:12'$\}$. At the airport shop, Alice wants to benefit from the exoneration of the 20% VAT charge. She first, derives from 'ICE' a specific pseudonym and removes the blocks of the message she does not want to disclose, resulting in the new following message $m' = \{a_2 =$ 'Company Name:Air France', $a_2 =$ 'Date of flight:09/12/2020', $a_3 =$ 'Destination:International'$\}$. Alice also modifies σ to fit m'. The airport shop checks the validity of the signature under the airline company's keys and verifies the eligibility of Alice to the exoneration.

The proposed solution satisfies several properties of interest. First, the user is able to control his identity and the attributes disclosed to service providers, thanks to sanitization and redaction features. These modifications can only be performed by a

designated user over the signature received from an identity provider. Second, from a service provider perspective, our proposal supports the legislation requirements, i.e., data minimization, while querying only necessary information to access services. Third, users may rely on self-generated pseudonyms for personalizing interactions with service providers, thus they remain under pseudonymity and unlinkable across providers. Fourth, the proposed malleable signature scheme supports strong privacy and unlinkability properties at a constant pairing computational cost. A complete implemented prototype shows the practical usability of our proposal for identity management systems adapted to resource-constrained devices. A concrete construction is proposed relying on a modified variant of the Pointcheval-Sanders signature (PS) scheme [113].

This chapter is organized as follows. Section 3.2 gives the motivation for the proposed system through three practical and topical illustrative scenarios. Section 3.3 gives a comparison between our proposal and closely-related work. Section 3.4 presents an overview of the proposed unlinkable malleable signature $\mathcal{UMS}$. Section 3.5 gives a high-level description of Pima and presents the identified threat models. Section 3.6 introduces the main building blocks and 3.7 details the concrete construction of Pima. Section 3.8 gives a formal security analysis w.r.t. the proposed threat models. Section 3.9 demonstrates, over both laptop and smartphone devices, high level performance measurements of the proposed construction, before concluding in Section 3.10.

3.2 Motivation Through Topical Illustrative Scenarios

In addition to the travelling scenario presented in the preliminaries of the chapter, this section illustrates the practicality of the proposed solution through three other use case scenarios.

The first one is a company scenario where employees are eligible to remotely access multiple services offered by several service providers. For this purpose, the company X (acting as an identity provider) provides each employee (i.e., referred to as a user) with a certified identity over his attributes (e.g. name, age, phone number, position/roles in the company, fields of expertise, information about his laptop). To keep his information protected, the company X is accustomed to updating the antivirus installed on the employee's devices when being connected to its network. Now that teleworking is the norm, the company encourages their employees to use their identity to authenticate to the antivirus provider (acting as a service provider) and to benefit from the recent update. The employee usually finds himself obliged to reveal personally identifiable information, which can be sold to third parties, thus exposing him to profiling or unwanted job advertising, while he only needs to prove that he is using a device owned by the company X. Using the proposed solution, the employee, previously provided with an electronic document certified by the company X, is able to modify the document for only showing the attribute "his laptop is owned by the company X" and proving so, thanks to the still valid certificate.

The second one is a healthcare application where a health organization (i.e., identity provider) provides their patients (i.e., users) with certified credentials including sensitive

health information (e.g. X-rays, treatments, pathologies) known as Electronic Healthcare Records (EHR), and personally identifiable data (e.g. name, home address, date of birth, and Social Security Number). Patients share these credentials with medical applications (i.e., service providers) to get relevant diagnosis and recommendations through their mobile devices without visiting a doctor. Although, patients have gained several advantages, e.g., saving time through these applications, their medical data have been exposed to increasing security and privacy risks. On the one hand, service providers are collecting sensitive data that can be sold to third parties, which harms patients' privacy. On the other hand, the huge amount of patients' data collected are exposed to high risk of leakage. These risks can be mitigated using the novel Pima system. Indeed, patients are given the full control over their sensitive medical data and they can select information to be disclosed when accessing the medical applications. For example, for a diabetes application, the patient can only reveal, in a pseudonymous session, the results of the diabetic balance sheet and the age group to which he belongs. Hence, the patient can get relevant diagnosis while preserving his privacy. In case of medical data breaches, he cannot be identified and his requests cannot be linked together across several medical applications.

The last illustrative scenario is about online purchase. Let us consider a student (i.e., user) who wants to benefit from the student fare when buying a transport ticket/pass online. The transport agency (i.e., service provider) may ask him to provide an enrollment certificate or a student card delivered by his university (i.e., identity provider) to ensure that he is a student under the age of 25. It is known that the enrollment certificate contains more personal information than needed, e.g. nationality, studies' level, specialty. Such information can be used for profiling and tracking. To tackle this situation, Pima assumes that the student is given an enrollment certificate signed by his university, and he is able to modify his enrollment certificate to only show the transport agency that he belongs to that university and that he is younger than 25 without revealing his date of birth. As such, the student remains anonymous inside the group of students belonging to the same university and being younger than 25 years old.

3.3 Related Work

This section presents a detailed comparison of the Pima system and signature primitive $\mathcal{UMS}$ with closely-related work.

There are several works designing privacy-preserving IDM systems for disclosing only the necessary information (i.e., users' attributes) and hiding the others. For instance, in [45], Sherman *et al.* propose a privacy-preserving IDM system that relies on two group signatures (Water's signatures and Groth-Sahai proof system) as a technique to hide the users' attributes. The main idea behind this work is that a user registers only once with the identity provider and obtains a source certificate that he locally stores. Then, upon requesting a service from a service provider, he randomizes and sanitizes the certificate. The service provider is able to authenticate the user by checking the validity of the certificate. The problem is that the disclosure of attributes with some

real values being revealed and shared between transactions can prevent transactions to remain unlinkable.
That is why, later on, sanitizable signatures have been introduced, for the first time, by Canard and Lescuyer [34] to design an original anonymous credential system. In their construction, authors use the signature of knowledge for avoiding the signature validity verification to require the sanitizer's key, thus ensuring the anonymity of the sanitizer. However, [34] introduces a tracing algorithm to re-identify the user as a sanitizer of a particular signature, thus, unlinkability is limited to trace-restricted unlinkability. Additionally, Canard and Lescuyer suggest that only the attributes' values are hidden with a symbol (like "#") covering the full length of the values, while the attributes' names remain visible. Thus, based on the names of attributes, their numbers, the length of their hidden values and the values of disclosed attributes during different sessions, transactions can be linkable between different service providers, and beyond that, service providers can try to extract some user's features. Service providers are also able to distinguish sanitized signatures from original ones based on the message form, which contradicts the privacy requirement.
Sanitizable features were also used to support privacy properties in smart mobile medical scenarios [145]. In their scheme, Xu *et al.* consider that data collected from patients' smart medical devices are signed by a signer (i.e., a doctor) before being stored at medical servers belonging to medical institutions. The signer is able to authorize medical institutions' members, referred to as sanitizers, to hide patients' identifiable information and accordingly adapt the signature before sharing it with verifiers (e.g., insurance companies, scientific research centers, etc.). In [145], sanitizers are considered as honest parties, which makes the scheme weakly unforgeable. Also, unlinkability and immutability are not supported.
Later, in [152], Zhu *et al.* propose a redactable signature scheme that allows users, that receive certified medical data from health monitoring sensors, to remove sensitive parts of the data when sharing them with third parties. The proposed scheme does not support neither unlinkability nor anonymity or pseudonymity. [152] only ensures that third parties are not able to distinguish a redacted signature from an original one.
In [89], authors propose a new scheme based on redactable signature to protect the privacy of sensitive data when requesting information stored on cloud servers. Indeed, users upload certified information that contains their sensitive data, on cloud servers. Afterwards, cloud servers are able to replace parts related to users' sensitive data with wildcards and accordingly adapt the signature before sharing them with data applicants. [89] fully trusts the sanitizer (i.e., cloud server), which weakens the unforgeability of the scheme, and does not support anonymity of users and unlinkability between several transactions. The privacy of the scheme is only about the impossibility of extracting information about users' sensitive data that has been redacted.

Table 3.1 presents a comparison of Pima and the $\mathcal{UMS}$ signature scheme with closely-related work.

The first part of Table 3.1 identifies the security an privacy properties achieved by identity management systems relying on sanitization and redaction, w.r.t. IDM systems' properties defined in Chapter 2, Sections 2.2.2.1 and 2.2.2.2, namely,

Table 3.1: Comparison between PIMA / $\mathcal{UMS}$ Schemes and Related Work

		PIMA	[45]	[34]	[152]	[145]	[89]	$\mathcal{UMS}$	[85]	[61]	[27]	[126]
Identity management systems properties	Unforgeability	✓	✓	✓	✓	✓[e]	✓[e]					
	Unlinkability	✓	✓[a]	✓[c]	✗	✗	✗					
	Strong privacy	✓	✗	✗	✓[d]	✗	✓[d]					
	Ano.(1)/Pym.(2)	(2)	(1)	(1)	✗	✗	✗					
Malleable signatures properties	Unforgeability							✓	✓	✓	✓	✓
	Immutability							✓	✓	✓	✓	✗
	Unlinkability							✓	✓[a]	✓[a]	✓[b]	✓
	Invisibility							✓	✗	✗	✓	N.A

NOTE: Ano. and Pym. denote respectively anonymity and pseudonymity; E and P stand for group exponentiations and pairing costs respectively; N.A is the abbreviation for Not Applicable; l denotes the message length; s denotes the length of the modifications' set; [a], [b] and [c] respectively indicate that unlinkability is limited to (i) the incapacity to link several transactions of the same user, (ii) the incapacity to link sanitized signatures to their origin and (iii) trace-restricted unlinkability; [d] states that only the impossibility to extract information about the modified parts is satisfied; [e] indicates that the property is limited to weak unforgeability as the sanitizer is considered as trusted; δ and γ respectively refers to the computation costs of a signature of knowledge generation and verification.

- *(i)* **unforgeability** of signatures unless secret keys are known, which refers to the data integrity **(S2)** and authenticity **(S3)** properties,
- *(ii)* **unlinkability** between either modified signatures or modified signature and their origin, which refers to the unlinkability property **(P4)**, including the two sub-properties,
- *(iii)* **strong privacy** of users' information (i.e., no information can be extracted more than what was disclosed), which refers to both anonymity property **(P1)** and the multi-show unlinkability of property **(P4)**, and
- *(iv)* **anonymity/pseudonymity** of users towards service providers, which refers to anonymity and pseudonymity properties **(P1)** and **(P2)**.

Note that a formal definition of security and privacy properties is given in Section 3.5.3.

Table 3.1 shows that, in [145] and [89], only weak unforgeability is satisfied. Indeed, the proposed schemes fully trust the sanitizer and offer no mean to verify whether signatures have been sanitized by authorized sanitizers and in an appropriate manner or not.
From Table 3.1, it also noticeable that [45] ensures partially unlinkability as a sanitized signature can be linked to its origin. For [34], the unlinkability property is limited to trace-restricted unlinkability. That is, when two or several sanitized signatures are traced to the same sanitizer, they can be linked to each other. Other schemes [152, 145, 89] do not support unlinkability, which raises critical privacy concerns as users' profiles and original messages can be extracted from their different transactions.
The strong privacy is also partially satisfied in [152, 89]. Indeed, it is not possible to extract information about sanitized parts as their are omitted. However, it is possible to deduce whether a message has been sanitized and who has performed the sanitization. Note that, unlike [34] and [45] that support users' anonymity, users can be identified by

third parties, in other works.

The second part of Table 3.1 presents the security properties satisfied by the $\mathcal{UMS}$ signature scheme vs other existing sanitizable and redactable signature schemes (i.e., [85], [61], [27] and [126]). The security and privacy properties that have been addressed in this comparison include **unforgeability** of signatures by unauthorized entities, **immutability** of modifications (i.e., only admissible modifications can be performed), **unlinkability** of modified signatures to each other or to their origin, and **invisibility** of what is modifiable or what has been modified in a message (cf. Chapter 2, Section 2.3.1.2). Table 3.1 shows that, apart from [126], $\mathcal{UMS}$ and other works satisfy security properties (i.e., unforgeability and immutability). In terms of privacy, the unlinkability property is fully satisfied only in [126]. Other works only guarantee partial unlinkability, and only [27] satisfies the invisibility requirement.

We deduce that relying on the reviewed schemes, a user is not able to prove, through his attributes, his eligibility to access a service or a resource without giving the possibility to retrieve extra information. Thus, his privacy may be compromised through identification, traceability and linkability between his different transactions with different parties. Thus, in PIMA, we propose a novel identity management system based on an unlinkable malleable signature $\mathcal{UMS}$ to allow users to access services and resources while strong privacy is preserved. PIMA also prevent malicious users from performing unauthorized modifications on their attributes.

In the following, we will give details about both $\mathcal{UMS}$ and PIMA. Through a detailed threat model and a concrete construction, we will prove that PIMA satisfies the required properties. The efficiency of PIMA will be also proven through a developed performance analysis.

3.4 Overview of the Unlinkable Malleable Signature $\mathcal{UMS}$

The $\mathcal{UMS}$ scheme relies on the following PPT algorithms (Setup, KeyGen, Sign, Modify, Verify) inspired from redactable signature scheme [17]:

$\mathsf{Setup}(1^\lambda) \rightarrow (pp, \mathtt{pk}_{sig}, \mathtt{sk}_{sig}, \mathtt{w}, \mathcal{W})$ takes as input a security parameter λ and outputs the public parameters pp. It also generates $(\mathtt{pk}_{sig}, \mathtt{sk}_{sig})$ the signer's pair of keys, and $\mathtt{w}$ a set of private weights associated with each type of message (i.e., attributes), and it derives the set of public weights $\mathcal{W}$.

$\mathsf{KeyGen}(pp, \mathtt{pk}_{sig}, \mathtt{sk}_{sig}) \rightarrow (\mathtt{pk}_{san}, \mathtt{sk}_{san})$ is run by the signer. It takes as input the public parameters pp and the signer's keys $(\mathtt{pk}_{sig}, \mathtt{sk}_{sig})$ and outputs the pair of keys $(\mathtt{pk}_{san}, \mathtt{sk}_{san})$ of users referred to as sanitizers.

$\mathsf{Sign}(pp, m, \mathtt{w}, \mathtt{sk}_{sig}, \mathtt{sk}_{san}, ADM) \rightarrow (m, \sigma)$ takes as input the signer's and sanitizer's private keys ($\mathtt{sk}_{sig}$ and $\mathtt{sk}_{san}$), a message m and a description of admissible modifications ADM. It then associates each message block with a specific weight in order to prevent the sanitizer either from modifying the fixed part or adding non admissible values in

the modifiable part. Finally, it signs the message with the two keys and outputs the message m and a signature σ.

$\mathsf{Modify}(pp, m, MOD, \sigma, \mathtt{sk}_{san}, \mathtt{pk}_{sig}) \rightarrow (m', \sigma', \mathtt{pk}'_{san}, \mathtt{sk}'_{san})$ first checks that a set of modifications MOD is admissible (i.e., $ADM(MOD)$=1) and accordingly modifies the message. It then randomizes the sanitizer's pair of keys w.r.t. a random r and adjusts the signature according to the new keys. The whole signature is randomized in order to ensure unlinkability between different sanitized signatures of the same message that are generated by the same source. The algorithm outputs the modified message $m' \leftarrow MOD(m)$ and the signature σ'.

$\mathsf{Verify}(pp, m, \sigma, \mathtt{pk}_{sig}, \mathtt{pk}'_{san}) \rightarrow b$ takes as input the signature σ, the message m, the signer's public key $\mathtt{pk}_{sig}$ and the sanitizer's randomized public key $\mathtt{pk}'_{san}$. It outputs a bit $b \in \{0, 1\}$ to state whether σ is valid or not.

The $\mathcal{UMS}$ signature scheme has to support the several security properties, namely **unforgeability**, **immutability**, **unlinkability** and **invisibility**, as defined in Chapter 2, Section 2.3.1.2. These properties will be proven, in Section 5.6, to be satisfied w.r.t. formal proofs of Pima's properties.

3.5 System and Threat Models

This section describes Pima system model, including the involved entities and the high-level algorithms, and it formally defines the threat model.

3.5.1 System Overview

The proposed IDM system involves three main entities namely the user ($\mathcal{U}$), the identity provider ($\mathcal{IP}$) and the service provider ($\mathcal{SP}$), as depicted in Figure 3.2. $\mathcal{U}$ receives, from $\mathcal{IP}$, a pseudonym and a certified identity $\sigma_{\mathcal{U}}$ over his attributes that he locally stores, during the Identity_Issue phase.

Afterwards, during the Service_Request phase, $\mathcal{U}$ is able to select the attributes he wants to disclose, to derive a new credential from $\sigma_{\mathcal{U}}$ and to pseudonymously interact with the $\mathcal{SP}$. $\mathcal{SP}$ verifies $\mathcal{U}$'s credential in order to give him access to services.

3.5.2 System Phases

The proposed IDM solution relies on the three following phases: Setup, Identity_Issue and Service_Request. The architecture of the new system is depicted in Figure 3.2.

Setup – This phase consists of setting up and initializing the whole system. The Pima.`Setup` algorithm runs the $\mathcal{UMS}$.`Setup` algorithm that outputs the system global parameter pp, likely to the $\mathcal{UMS}$ scheme. The $\mathcal{IP}$ key pair $(\mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}})$ corresponds to

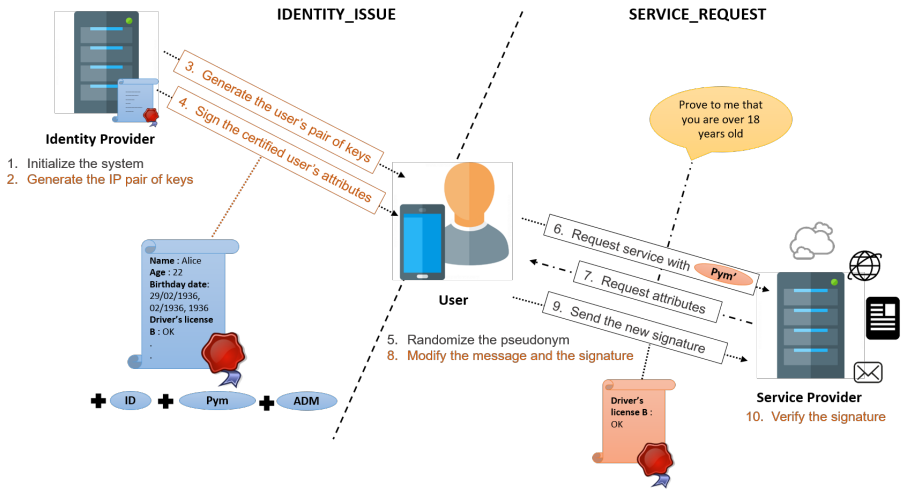


Figure 3.2: PIMA Architecture and Key Interactions between Entities

the signer's one generated during the $\mathcal{UMS}$.Setup algorithm. Next, the $\mathcal{IP}$ generates a set of public parameters $\{P_n\}_{n=1}^{N}$ (N is the maximum number of possible types of attributes delivered by the $\mathcal{IP}$) that correspond to the weights associated with each type of attribute [2]. Then, $\mathcal{IP}$ performs the PIMA.UserKeyGen algorithm using the $\mathcal{UMS}$.KeyGen one in order to generate the key pair $(\mathtt{pk}_{\mathcal{U}_j}, \mathtt{sk}_{\mathcal{U}_j})$ of a user $\mathcal{U}_j$ (referred to as the sanitizer in the $\mathcal{UMS}$.KeyGen algorithm).

IDENTITY_ISSUE – This phase occurs when $\mathcal{IP}$ issues the identity of a requesting user $\mathcal{U}_j$. Indeed, $\mathcal{IP}$ generates an identifier ID to which he associates a pseudonym Pym and a set of attributes $Attr = \{a_k\}_{k=1}^{l}$ where l is the number of attributes describing the identity of $\mathcal{U}_j$ and $Attr \subseteq \mathtt{A}$ ($\mathtt{A}$ is the attributes' universe). $\mathcal{IP}$ then defines a message m in the form of $m = \{ID_{\mathcal{IP}}, D, opt, Attr\}$, where $ID_{\mathcal{IP}}$ is the identifier of $\mathcal{IP}$, D is a set of date values such as the identity issuing and expiration dates and opt represents other options specific to $\mathcal{IP}$. These elements form the fixed part of the message m. Next, $\mathcal{IP}$ performs the PIMA.Sign algorithm by using the exact same algorithm as $\mathcal{UMS}$.Sign, taking as input the message m, a set of admissible modifications ADM, the secret keys of respectively $\mathcal{IP}$ and $\mathcal{U}_j$, the pseudonym Pym and a set of weights associated with the $\mathcal{U}_j$'s attributes. It produces a signature $\sigma_{\mathcal{U}_j}$. Finally, $\mathcal{IP}$ sends the tuple $(ID, Pym, m, \sigma_{\mathcal{U}_j}, ADM)$ to $\mathcal{U}_j$ who locally stores them.

SERVICE_REQUEST – This phase occurs once $\mathcal{U}_j$ needs to get access to a service offered by a service provider $\mathcal{SP}_j \in \mathtt{SP}$ where $\mathtt{SP}$ is the $\mathcal{SP}$'s universe. Then $\mathcal{U}_j$ first solicits $\mathcal{SP}_j$ using a new pseudonym that he derives from his local pseudonym Pym. This new pseudonym is written as $Pym_{\mathcal{U}_j}@\mathcal{SP}_j$ for the pseudonym of $\mathcal{U}_j$ at $\mathcal{SP}_j$. We assume that each user has different pseudonyms at different $\mathcal{SP}$s in order to ensure unlinkability between several transactions. Once receiving the pseudonym $Pym_{\mathcal{U}_j}@\mathcal{SP}_j$,

[2] The objective behind the use of weights associated with each type of attribute is to prevent a malicious sanitizer from (i) modifying the message blocks with non admissible inputs as each block is associated with a specific weight known to the $\mathcal{SP}$, and (ii) exchanging the attributes' values, e.g. the knowledge of the user's name attribute value "Florence" (provided with a weight P_1) cannot help to change the town attribute value to "Florence" as it is associated to a weight $P_2 \neq P_1$.

$\mathcal{SP}_j$ picks a set of attributes $AttrReq = \{attr_j\}_{j=1}^{n}$ allowing $\mathcal{U}_j$ to have access to the requested service if he succeeds in proving their possession. We assume that $AttrReq$ contains the minimum number of attributes that must be provided by the user which fits the data minimization requirement w.r.t. the GDPR [57]. Next, according to the $AttrReq$ received set, $\mathcal{U}_j$ defines a set of possible modifications MOD (w.r.t. the set of admissible modifications ADM) and accordingly modifies the original signature by running the PIMA.`Modify` algorithm which is the exact same algorithm as $\mathcal{UMS}$.Modify, taking as input the message m, the signature $\sigma_{\mathcal{U}_j}$, the $\mathcal{IP}$ public key $\mathtt{pk}_{\mathcal{IP}}$, the $\mathcal{U}_j$'s secret key $sk_{\mathcal{U}_j}$ and the modifications' set MOD. The algorithm then outputs the modified message m', the modified signature $\sigma'_{\mathcal{U}_j}$ and the randomized pair of keys $(\mathtt{sk}'_{\mathcal{U}_j}, \mathtt{pk}'_{\mathcal{U}_j})$ of $\mathcal{U}_j$. $\mathcal{U}_j$ then sends the tuple $(m', \sigma'_{\mathcal{U}_j}, \mathtt{pk}'_{\mathcal{U}_j})$ to $\mathcal{SP}_j$. This latter takes these elements with the $\mathcal{IP}$ public key $\mathtt{pk}_{\mathcal{IP}}$ as inputs for the PIMA.`Verify` algorithm which relies on both the $\mathcal{UMS}$.Verify algorithm and two other verifications w.r.t. the user's randomized public key. $\mathcal{SP}_j$ checks whether the attributes of $\mathcal{U}_j$ are certified by $\mathcal{IP}$ and provides the service if the verification succeeds, otherwise he rejects the request.

3.5.3 Threat Model

This section first presents the adversaries considered in PIMA, and then, gives formal definitions of the different security and privacy properties.

Two main adversaries are identified as follows:

- **A malicious user ($\mathcal{U}$)**: attempts to override their rights and authorizations in order to generate valid credentials.
- **A honest but curious service provider ($\mathcal{SP}$)**: tries to link users' transactions to each other and to collect extra information about users in order to identify the entity behind the request.

Malicious users are considered against security requirements, namely unforgeability, while honest but curious service providers are considered against privacy requirements, i.e., unlinkability and strong privacy.

3.5.3.1 Unforgeability

In the PIMA system, unforgeability means that it is not possible to produce a valid message/signature pair unless the private key of the $\mathcal{IP}$ and the secret values of attributes' weights are known. Formally, this is defined in a game $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ where an adversary $\mathcal{A}$, acting a malicious user, has access to a PIMA.`Sign` oracle. Then, $\mathcal{A}$ chooses a message m^* that has neither given before nor obtained by modifying given messages. $\mathcal{A}$ succeeds if it outputs a valid message/signature pair (m^*, σ^*) such that the PIMA.`Verify` verification holds.

Definition 6. Unforgeability – We say that PIMA satisfies the unforgeability property, if for every PPT adversary $\mathcal{A}$, there exists a negligible function κ such that $Pr[\mathbf{Exp}_{\mathcal{A}}^{unforg}(1^{\lambda}) = 1] \leq \kappa(\lambda)$ where $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ is detailed hereafter.

$\boldsymbol{Exp}_{\mathcal{A}}^{unforg}(\lambda)$
$(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}}, \mathtt{w}, \mathcal{W}) \leftarrow$ PIMA.`Setup`(λ)
$(\mathtt{pk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{U}}) \leftarrow$ PIMA.`UserKeyGen`$(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}})$
$\mathcal{O} \leftarrow$ PIMA.`Sign`$(\cdot,\cdot,\mathtt{sk}_{\mathcal{IP}},\cdot)$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}}$ $(\mathtt{pk}_{\mathcal{U}},\mathtt{sk}_{\mathcal{U}},\mathtt{pk}_{\mathcal{IP}},pp,\mathcal{W}, ADM)$
letting m^i *and* $\sigma_{\mathcal{U}}^i$ *denote the queries and answers to and from oracle* PIMA.`Sign`
and $m^* \notin \{MOD(m^i) \mid MOD \text{ with } ADM^i(MOD) = 1\}$
If PIMA.`Verify`$(pp, m^*, \sigma^*, \mathtt{pk}_{\mathcal{IP}},\mathtt{pk}_{\mathcal{U}}, \mathcal{W}) = 1$
return 1
Else return 0

Remark 1. Note that we say that PIMA guarantees the strong unforgeability property as it satisfies not only the original definition of a signature scheme unforgeability, but also ensures the unforgeability of admissible modifications, i.e., which refers to the immutability property of $\mathcal{UMS}$.

3.5.3.2 Unlinkability

The unlinkability property covers two sub-properties: (i) the *multi-transactions unlinkability* guarantees that two or several service providers are not able to collude and link several modified signatures derived from a signature over the same message and transmitted over several transactions, (ii) the *to-original unlinkability* ensures that an adversary cannot link the modified signature to the original one even if this latter is known. Note that the *multi-transactions unlinkability* and the *to-original unlinkability* refer to the ***multi-show unlinkability*** and ***issue-show unlinkability*** of (P4) (cf. Chapter 2, Section 2.2.2.2).
Formally, the *multi-transactions unlinkability* property is defined in a game $\mathbf{Exp}_{\mathcal{A}}^{MT-Game}$ where an adversary $\mathcal{A}$, acting as colluding curious $\mathcal{SP}$s, has access to a PIMA.`Modify` oracle on the same message m^* and modifications MOD^* for two service providers $\mathcal{SP}_1$ and $\mathcal{SP}_2$. A left-or-right oracle LoRMT is initialized with a secret random bit b and returns to $\mathcal{A}$ two modified signatures derived either from the same signature or two different signatures over the same message m^* and modifications MOD^*. The adversary wins the game if he successfully predicts the bit b.

Definition 7. Multi-Transactions Unlinkability – We say that PIMA satisfies the *multi-transactions* unlinkability, if for every PPT adversary $\mathcal{A}$, there exists a negligible function κ, such that $Pr[\mathbf{Exp}_{\mathcal{A}}^{MT-Game}(1^{\lambda}) = 1] = \frac{1}{2} \pm \kappa(\lambda)$, where $\mathbf{Exp}_{\mathcal{A}}^{MT-Game}$ is presented as follows.

$\boldsymbol{Exp}_{\mathcal{A}}^{MT-Game}(\lambda)$
$(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}}, \mathtt{w}, \mathcal{W}) \leftarrow$ PIMA.`Setup`(λ)
$(\mathtt{pk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{U}}) \leftarrow$ PIMA.`UserKeyGen`$(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}})$
$(m^*, \sigma) \leftarrow$ PIMA.`Sign` $(pp, m^*, \mathtt{w}, \mathtt{sk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{U}}, ADM^*)\}$
$b \leftarrow \{0, 1\}$
$\mathcal{O} \leftarrow \{$PIMA.`Modify` $(\cdot, \mathtt{sk}_{\mathcal{U}}, \cdot)$ for $\mathcal{SP}_1$ and $\mathcal{SP}_2$, LoRMT$(\cdot, \cdot, b)\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}$ $(pp, \mathtt{pk}_{\mathcal{IP}}, \mathcal{W})$
If $b = b'$
  return 1
Else return 0

LoRMT $(pp, m^*, MOD^*, \sigma, \mathtt{sk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}}, \mathtt{w}, MOD^*, b)$

if $(b = 0)$ **then** {
$(m'^*, \sigma_1'^*) \leftarrow$ PIMA.`Modify`$(pp, m^*, MOD^*, \sigma, \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{IP}})$ for $\mathcal{SP}_1$
$(m'^*, \sigma_2'^*) \leftarrow$ PIMA.`Modify`$(pp, m^*, MOD^*, \sigma, \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{IP}})$ for $\mathcal{SP}_2$ }
else {
$(m'^*, \sigma_1'^*) \leftarrow$ PIMA.`Modify`$(pp, m^*, MOD^*, \sigma, \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{IP}})$ for $\mathcal{SP}_1$
$(m^*, \sigma') \leftarrow$ PIMA.`Sign`$(pp, m^*, \mathtt{w}, \mathtt{sk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{IP}}, \sigma, ADM^*)$
$(m'^*, \sigma_2'^*) \leftarrow$ PIMA.`Modify`$(pp, m^*, MOD^*, \sigma', \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{IP}})$ for $\mathcal{SP}_2$ }
return $(\sigma_1'^*, \sigma_2'^*)$

The *to-original* unlinkability holds if $\mathcal{A}$, acting as a curious service provider[3], is given access to PIMA.`Modify` oracle on the same message m^* and modifications MOD^* and two signatures σ_0 and σ_1 for the same user. $\mathcal{A}$ also gets access to a left-or-right oracle LoRTO which is initialized with a secret random bit $b \in \{0, 1\}$. $\mathcal{A}$ is given back a modified signature over the same message m^*, modifications MOD^* and signature σ_b. To win this game, $\mathcal{A}$ should successfully predict b.

Definition 8. To-Original Unlinkability – We say that PIMA satisfies the *to-original* unlinkability property, if for every PPT adversary $\mathcal{A}$, there exists a negligible function κ, such that $Pr[\mathbf{Exp}_{\mathcal{A}}^{TO-Game}(1^{\lambda}) = 1] = \frac{1}{2} \pm \kappa(\lambda)$, where $\mathbf{Exp}_{\mathcal{A}}^{TO-Game}$ is defined as follows.

$\boldsymbol{Exp}_{\mathcal{A}}^{TO-Game}(\lambda)$
$(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}}, \mathtt{w}, \mathcal{W}) \leftarrow$ PIMA.`Setup`(λ)
$(\mathtt{pk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{U}}) \leftarrow$ PIMA.`UserKeyGen`$(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}})$
$(m^*, \sigma_0) \leftarrow$ PIMA.`Sign` $(pp, m^*, \mathtt{w}, \mathtt{sk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{U}}, ADM^*)\}$
$(m^*, \sigma_1) \leftarrow$ PIMA.`Sign` $(pp, m^*, \mathtt{w}, \mathtt{sk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{U}}, ADM^*)\}$
$b \leftarrow \{0, 1\}$
$\mathcal{O} \leftarrow \{$PIMA.`Modify` $(\cdot, \sigma_0, \mathtt{sk}_{\mathcal{U}}, \cdot)$, $\{$PIMA.`Modify` $(\cdot, \sigma_1, \mathtt{sk}_{\mathcal{U}}, \cdot)$, LoRTO$(\cdot, \cdot, b)\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}$ $(pp, \mathtt{pk}_{\mathcal{IP}}, \mathcal{W})$
If $b = b'$
  return 1
Else return 0

LoRTO $(pp, m^*, MOD^*, \sigma_0, \sigma_1, \mathtt{sk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{IP}}, MOD^*, b)$

$(m'^*, \sigma_b'^*) \leftarrow$ PIMA.`Modify`$(pp, m^*, MOD^*, \sigma_b, \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{IP}})$
return $\sigma_b'^*$

[3]The adversary considered against the to-original unlinkability property refers to as a curious service provider or colluding service providers. A curious $\mathcal{IP}$ could be also considered against this property. This assumption does not pose plausible threats to the proposed system unless a collusion between $\mathcal{IP}$ and different $\mathcal{SP}$s occurs in order to trace users' transactions. Nevertheless, this assumption of collusion between the $\mathcal{IP}$ and service providers contradicts the fact that the $\mathcal{IP}$ is honest.

3.5.3.3 Strong Privacy

The strong privacy property refers to the impossibility to extract information about neither the user nor the exchanged messages, more than what was voluntarily revealed. Indeed, it is unfeasible for a curious service provider neither to identify a particular user based on various transactions' information, nor to decide which data have been modified or deleted from a given message, i.e., modified signature. This property will be discussed informally in Section 3.8.

3.6 Building Blocks

This section presents the Pointcheval-Sanders scheme as a main building block and introduces the proposed modifications to be securely integrated in the PIMA system.

Pointcheval-Sanders (PS) Signature Scheme – The PS signature [113] works in an asymmetric bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, g_1, g_2)$. It is inspired from Camenisch and Lysyanskaya (CL) signature scheme [29] and proposes the same features while avoiding the linear-size drawback of CL-signatures, hence, it is considered as more efficient in terms of processing times for multi-block messages signature. The Pointcheval-Sanders signature scheme includes three main primitives, namely KEY GENERATION, SIGN and VERIFY.
KEY GENERATION – The secret key is selected as $(x, y) \leftarrow \mathbb{Z}_q^{*2}$ and the corresponding public key is computed as $(X, Y) \leftarrow (g_2^x, g_2^y)$.
SIGN – To sign a message $m \in \mathbb{Z}_q^*$, the signer picks a random $h \leftarrow \mathbb{G}_1^*$, computes $a := h$ and $b := h^{(x+y.m)}$, and outputs the signature $\sigma = (a, b)$.
VERIFY – To verify the validity of the signature, a verifier checks if $e(a, X.Y^m) = e(b, g_2)$ holds.
Relying on the LRSW assumption, the Pointcheval-Sanders signature scheme is proven to be existential unforgeable under chosen message attacks (EUF-CMA) [62].

Modification 1 to Pointcheval-Sanders scheme – As the original scheme does not support the unlinkability feature [4], based on [151] works, we propose a modified version as follows.
The KEY GENERATION primitive remains the same. The SIGN primitive for a message $m \in \mathbb{Z}_q^*$, requires the signer to select a random $h \leftarrow \mathbb{G}_1^*$ and a random $s \leftarrow \mathbb{Z}_q^*$, to compute $a = h^s$ and $b = h^{(x+y \cdot m)}$, and to output the signature $\sigma = (s, a, b)$. To verify that the signature is valid, a verifier checks if Equation 3.1 holds.

$$e(a, X \cdot Y^m) = e(b, g_2)^s \tag{3.1}$$

Correctness. $e(a, X \cdot Y^m) = e(h^s, g_2)^{x+ym} = e(h^{x+ym}, g_2)^s = e(b, g_2)^s$

[4] The randomization of the PS scheme states that based on a random $t \in \mathbb{Z}_q^*$, the randomized signature is denoted as $\sigma' = (a^t, b^t)$. Relying on pairing functions, a randomized signature can be linked to its origin, which contradicts the *To-Original unlinkability* property

Modification 2 to get a randomizable signature scheme – To make unlinkable several presentations of the same signature, this latter has to be randomizable. This can be obtained as follows. Given a signature $\sigma = (s, a, b)$, the signer (or any other entity) chooses two randoms $r_1, r_2 \in \mathbb{Z}_q^*$, computes $s' = r_2 s$, $a' = a^{r_1 r_2}, b' = b^{r_1}$ and outputs the new signature $\sigma' = (s', a', b')$. We can easily check that the equation 3.1 still holds.

Modification 3 to get randomizable keys – Randomization of keys aims at hiding the signer's identity. This can be achieved as follows. Given a signature $\sigma = (s, a, b)$, the signer selects a random $\rho \in \mathbb{Z}_q^*$, runs two algorithms `RandSK` and `RandPK` for randomizing respectively the signer's secret and public key and outputs $(x', y') = \texttt{RandSK}((x, y), \rho) = (x\rho, y\rho)$ and $(X', Y') = \texttt{RandPK}((X, Y), \rho) = (g_2^{x'}, g_2^{y'}) = (g_2^{x\rho}, g_2^{y\rho})$. Then, the signer computes $\tilde{b} = b^\rho = h^{(x\rho + my\rho)} = h^{(x' + my')}$ and outputs the signature $\tilde{\sigma} = (s, a, \tilde{b})$, which can be verified if 3.1 holds using the public key (X', Y').

3.7 PIMA Algorithms

This section details the concrete construction of PIMA based on the modifications of the PS signature scheme proposed in Section 3.6.

3.7.1 Setup

This phase includes two main algorithms referred to as PIMA.`Setup` (cf., Algorithm 1) and PIMA.`UserKeyGen` (cf., Algorithm 2).

Algorithm 1 PIMA.`Setup` algorithm

1: **Inputs:** the security parameter λ
2: **Output:** the parameters pp, the $\mathcal{IP}$'s keys ($\texttt{pk}_{\mathcal{IP}}$,$\texttt{sk}_{\mathcal{IP}}$) and the weights $(\texttt{w}, \mathcal{W})$
3: set an asymmetric bilinear group of type 3 environment $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e)$ where $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$ is an asymmetric type 3 pairing function;
4: pick at random $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and $x, y \in \mathbb{Z}_q^*$ and compute $X := g_2^x$; $Y := g_2^y$;
5: set $pp = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, g_1, g_2)$; $\texttt{pk}_{\mathcal{IP}} = (X, Y)$ and $\texttt{sk}_{\mathcal{IP}} = (x, y)$;
6: pick at random $\{p_j\}_{j\in[1,P]}, \{q_j\}_{j\in[1,P]} \in \mathbb{Z}_q^*$ where P is the maximum number of attributes' types supported by the $\mathcal{IP}$;
7: **for all** $j \in \{1, \cdots, P\}$ **do**
8: $\quad P_j := g_2^{p_j y}$; $Q_j := g_2^{q_j}$;
9: **end for**
10: $\texttt{w} = \{(p_j, q_j)\}_{j\in[1,P]}$; $\mathcal{W} = \{(P_j, Q_j)\}_{j\in[1,P]}$ where w (resp. $\mathcal{W}$) is the set of secret (resp. public) weights associated to attributes
11: **return** $(pp, \texttt{pk}_{\mathcal{IP}}, \texttt{sk}_{\mathcal{IP}}, \texttt{w}, \mathcal{W})$

Note that the tuple $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, g_1, g_2, X, Y, \mathcal{W})$ is known by all the system's entities while x, y and w are kept secret at the $\mathcal{IP}$. Afterwards, the $\mathcal{IP}$ generates the pair of keys for each of its users by running the PIMA.`UserKeyGen` algorithm (cf., Algorithm 2).

Note that the pair of keys $(\texttt{pk}_{\mathcal{U}}, \texttt{sk}_{\mathcal{U}})$ associated with the user's identifier $ID_{\mathcal{U}}$ is

Algorithm 2 PIMA.`UserKeyGen` algorithm

1: **Inputs:** the public parameters pp and the key pair $(\mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{IP}})$ of $\mathcal{IP}$
2: **Output:** the pair of public and private keys $(\mathtt{pk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{U}})$ of $\mathcal{U}$
3: pick two random values $\alpha, \beta \leftarrow \mathbb{Z}_q^*$;
4: compute $x_u = \alpha x$; $y_u = \beta y$; $X_u = g_2^{x_u}$; $Y_u = g_2^{y_u}$;
5: set $\mathtt{pk}_{\mathcal{U}} = (X_u, Y_u)$ and $\mathtt{sk}_{\mathcal{U}} = (x_u, y_u)$;
6: **return** $(\mathtt{pk}_{\mathcal{U}}, \mathtt{sk}_{\mathcal{U}})$

stored at both the $\mathcal{IP}$'s and the user's sides[5].

3.7.2 Identity_Issue

This phase relies on the PIMA.`Sign` algorithm that is run by $\mathcal{IP}$, upon receiving a list of attributes $Attr=\{a_j\}_{j\in[1,N]}$ from $\mathcal{U}_j$, where N is the number of user's attributes. The $\mathcal{IP}$ generates an $ID_{\mathcal{IP}}$ and the corresponding pseudonym Pym by picking a random value $s_u \leftarrow \mathbb{Z}_q^*$, and derives a message m, as follows:
The message can be written as $m = m_{FIX}+m_{MOD}$ where m_{FIX} is the fixed part (i.e., non-modifiable blocks introduced by the $\mathcal{IP}$) and m_{MOD} is the modifiable part (i.e., the set of attributes that can be modified by the user according to the admissible modifications). This part can be presented as follows: $m_{MOD} = \sum_{j\in[1,N]}(a_j + \sum_k mod_k(a_j))$ where mod_k are the different possible modifications for a given attribute. The set of possible admissible modifications ADM consists of the indices of the modifiable blocks. In our case, it is considered to be the same as $[1, N]$ since we assume that all attributes can be modified to one of the given modifications mod_k.
In the following, we assume that m can be written as $m = \{m_j\}_{j\in\{1..n\}}$ where n is the length of the message m, and m_j denotes either the attribute a_j, its possible modification or the fixed parts of the message. This message is signed by $\mathcal{IP}$ as shown in Algorithm 3.

Algorithm 3 PIMA.`Sign` algorithm

1: **Inputs:** the public parameters pp, the message m, the set of secret weights $\mathtt{w}$, secret keys $\mathtt{sk}_{\mathcal{IP}}$ and $\mathtt{sk}_{\mathcal{U}}$ of respectively $\mathcal{IP}$ and $\mathcal{U}$ and admissible modifications ADM
2: **Output:** the message m and the corresponding signature $\sigma_{\mathcal{U}}$
3: pick at random $s_u \in \mathbb{Z}_q^*$ and $h_u \leftarrow \mathbb{G}_1^*$ where h_u is reinitialized for each issued signature ;
4: compute $a_u = h_u^{s_u}$; $b_u = h_u^{(x+\sum_{j=1}^{n}(m_j y + p_j m_j y + q_j))}$; $c_u = h_u^{(\alpha x + \sum_{j=1}^{n} m_j \beta y)}$;
5: set $\sigma_{\mathcal{U}} = (s_u, a_u, b_u, c_u)$;
6: **return** $(m, \sigma_{\mathcal{U}})$

Note that in order to modify the signature, $\mathcal{U}$ needs further elements than only the delivered signature. Thus, $\mathcal{IP}$ generates a set $\mathcal{H}_{\mathcal{U}} = \{H_j\}_{j\in ADM}$ where $H_j = h_u^{p_j m_j y + q_j}$ and sets the tuple $(m, \sigma_{\mathcal{U}}, g^{\alpha}, g^{\beta}, h_u^y, \mathcal{H}_{\mathcal{U}})$ that he sends to $\mathcal{U}$.

[5]The fact that the user's secret key is generated by and stored at the $\mathcal{IP}$ does not affect the security of PIMA as the $\mathcal{IP}$ is considered as a fully trusted authority.

3.7.3 Service_Request

This phase involves two algorithms, namely PIMA.`Modify` (cf., Algorithm 4) and PIMA.`Verify` (cf., Algorithm 5). When $\mathcal{U}$ wants to access a service offered by a given $\mathcal{SP}_1$, he first selects a random $r_{\mathcal{SP}_1} \in \mathbb{Z}_q^*$, computes $s_{\mathcal{SP}_1} = r_{\mathcal{SP}_1} s_u$ to be the pseudonym of $\mathcal{U}$ at $\mathcal{SP}_1$ (cf., Section 3.5) and interacts with $\mathcal{SP}_1$ via the generated pseudonym. $\mathcal{SP}_1$ sends back the set of requested attributes $AttrReq \in \{0,1\}^*$. Then, according to $AttrReq$ and ADM, $\mathcal{U}$ defines the set of modifications MOD to be performed on m, and runs PIMA.`Modify`.

Algorithm 4 PIMA.`Modify` algorithm

Inputs: the message m, the signature $\sigma_{\mathcal{U}}$, the set of modifications MOD, the $\mathcal{H}_{\mathcal{U}}$ set, the tuple $(g^{\alpha}, g^{\beta}, h_u^y)$, the secret key $\mathtt{sk}_{\mathcal{U}}$ of $\mathcal{U}$ and the public key $\mathtt{pk}_{\mathcal{IP}}$ of $\mathcal{IP}$
Output: the modified message m', the corresponding signature $\sigma'_{\mathcal{U}}$ and the randomized pair of keys $(\mathtt{pk}'_{\mathcal{U}}, \mathtt{sk}'_{\mathcal{U}})$ of $\mathcal{U}$
set $m' = \{m_j\}_{j \in \{1..n\} \setminus MOD}$;
pick a random value $r_{\mathcal{SP}_1} \leftarrow \mathbb{Z}_q^*$;
compute $a'_u = a_u^{r_{\mathcal{SP}_1}}$; $b'_u = b_u \cdot \prod_{k \in MOD} (H_k{}^{-1})(h_u^y)^{-m_k}$; $c'_u = c_u \cdot \prod_{k \in MOD} (h_u^{\beta y})^{-m_k}$);
pick two random values $\rho, z \in \mathbb{Z}_q^*$;
set $\mathtt{sk}'_{\mathcal{U}} = (x'_u, y'_u) = (\alpha\rho x, \beta\rho y)$; $\mathtt{pk}'_{\mathcal{U}} = (X'_u, Y'_u) = (g_2^{\alpha\rho x}, g_2^{\beta\rho y}) = (X_u^{\rho}, Y_u^{\rho})$;
compute $g_1^{\alpha z}$; $g_1^{\beta z}$; $g_1^{\frac{z}{\rho}}$;
pick a random value $r_1 \in \mathbb{Z}_q^*$;
compute $s_{\mathcal{SP}_1} = s_u \cdot r_{\mathcal{SP}_1}$; $\tilde{a_u} = {a'_u}^{r1}$; $\tilde{b_u} = {b'_u}^{r_1}$; $\tilde{c_u} = {c'_u}^{\rho r_1}$; $\tilde{d_u} = \tilde{b_u} \cdot \tilde{c_u}$;
set $\sigma'_u = (s_{\mathcal{SP}_1}, \tilde{a_u}, \tilde{d_u})$;
return $(m', \sigma'_u, \mathtt{pk}'_{\mathcal{U}}, \mathtt{sk}'_{\mathcal{U}})$

Note that the user keeps secret the randomized private key $\mathtt{sk}'_{\mathcal{U}}$ and sends the tuple $(m', \sigma'_{\mathcal{U}}, X'_u, Y'_u, g_1^{\alpha z}, g_1^{\beta z}, g_1^{\frac{z}{\rho}})$ to $\mathcal{SP}_1$. This latter should check the validity of the received signature, by running the PIMA.`Verify`.

Algorithm 5 PIMA.`Verify` algorithm

Inputs: the public parameters pp, the message m', the signature $\sigma'_{\mathcal{U}}$, the public keys $\mathtt{pk}_{\mathcal{IP}}$ and $\mathtt{pk}'_{\mathcal{U}}$ of $\mathcal{IP}$ and $\mathcal{U}$, the $\mathcal{W}$ set, and the elements $g^{\alpha z}$, $g^{\beta z}$ and $g^{\frac{z}{\rho}}$
Output: a bit $b \in \{0,1\}$
extract a subset $\mathcal{W}_l$ of length l from $\mathcal{W}$ according to the message m' blocks;
if ($e(\tilde{a_u}, X \cdot X'_u \cdot \prod_{j=1}^{l} Q_j (P_j \cdot Y \cdot Y'_u)^{m'_j}) = e(\tilde{d_u}, g_2)^{s_{\mathcal{SP}_1}}$
    and $e(g_1^{\frac{z}{\rho}}, X'_u) = e(g_1^{\alpha z}, X)$
    and $e(g_1^{\frac{z}{\rho}}, Y'_u) = e(g_1^{\beta z}, Y)$)
then ($b = 1$)
else ($b = 0$)
return b

Correctness. The first verification equation holds as
$e(\tilde{a_u}, X \cdot X'_u \cdot \prod Q_j (P_j \cdot Y \cdot Y'_u)^{m_j}) = e(h_u^{s_u r_{\mathcal{SP}_1} r_1}, g_2)^{(x + \alpha\rho x + \sum q_j + (p_j y + y + \beta\rho y) m_j)}$
$= e(h_u^{r_1 (x + \alpha\rho x + \sum (m_j y + p_j m_j y + m_j \beta\rho y + q_j))}, g_2)^{s_{\mathcal{SP}_1}}$
$= e(\tilde{d_u}, g_2)^{s_{\mathcal{SP}_1}}$

The second one holds as $e(g_1^{\frac{z}{\rho}}, X'_u) = e(g_1^{\frac{z}{\rho}}, g_2^{\alpha x \rho}) = e(g_1^{\alpha z}, g_2^{x}) = e(g_1^{\alpha z}, X)$, and the last one holds as $e(g_1^{\frac{z}{\rho}}, Y'_u) = e(g_1^{\frac{z}{\rho}}, g_2^{\beta y \rho}) = e(g_1^{\beta z}, g_2^{y}) = e(g_1^{\beta z}, Y)$.

3.8 Security Analysis

This section shows that PIMA satisfies the unforgeability, unlinkability and privacy requirements w.r.t. the threat models defined in Section 3.5.3. Furthermore, it deduces from the formal proofs of PIMA's properties that the proposed $\mathcal{UMS}$ satisfies the properties listed in Section 3.4. Indeed, on the one hand, the strong unforgeability property of the PIMA system implies unforgeability and immutability of $\mathcal{UMS}$. On the other hand, unlinkability and strong privacy properties of PIMA imply the unlinkability and the invisibility of $\mathcal{UMS}$.

3.8.1 Unforgeability

Theorem 1 (Unforgeability)**.** The PIMA system satisfies the unforgeability requirement, with respect to $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ experiment.

Proof. In this proof, we suppose that $\mathcal{A}$ is allowed to query, as many times as he wants, the PIMA.`Sign` oracle on two one-block messages m_1^i and m_2^i. Thus for each session i, $\mathcal{A}$ receives two signatures σ_1^i and σ_2^i over the messages, respectively, for a user ($\mathcal{U}$). The signatures can be parsed as follows.

$$\begin{cases} \sigma_1^i = (s, {a_{1u}}^i = (h_{1u})^s, {b_{1u}}^i = (h_{1u})^{x+m_1^i y+pm_1^i y+q}, {c_{1u}}^i = (h_{1u})^{\alpha * x+m_1^i \beta y}) \\ \sigma_2^i = (s, {a_{2u}}^i = (h_{2u})^s, {b_{2u}}^i = (h_{2u})^{x+m_2^i y+pm_2^i y+q}, {c_{2u}}^i = (h_{2u})^{\alpha * x+m_2^i \beta y}) \end{cases}$$

with $h_{1u} = g^{v_{1u}}$ and $h_{2u} = g^{v_{2u}}$, where v_{1u} and $v_{2u} \in Z_q^*$.

We suppose that $\mathcal{A}$ is extremely strong that he knows the exponents of the signature's elements. Then, $\mathcal{A}$ gets access to the following linear system.

$$v_{1u}(x + m_1^i y + pm_1^i y + q), \forall i \quad (3.2)$$
$$v_{1u}(\alpha * x + m_1^i \beta * y), \forall i \quad (3.3)$$
$$v_{2u}(x + m_2^i y + pm_2^i y + q), \forall i \quad (3.4)$$
$$v_{2u}(\alpha * x + m_2^i \beta * y), \forall i \quad (3.5)$$

$\mathcal{A}$ deduces the values of v_{1u} and v_{2u} as it knows αx and βy and obtains:

$$A_i = x + m_1^i y + pm_1^i y + q, \forall i \qquad (3.6) \qquad B_i = x + m_2^i y + pm_2^i y + q, \forall i \qquad (3.7)$$

$\mathcal{A}$ is asked to produce a new valid pair message/signature (m^*, σ^*), where m^* has not been given before. Thus, $\mathcal{A}$ aims at forging $x + m^*y + pm^*y + q$. Computing (3.7)-(3.6), the linear system becomes:

$$C_i/y = 1 + p, \forall i \qquad (3.8) \qquad D_i = x + q, \forall i \qquad (3.9)$$

Thus, for each session i, $\mathcal{A}$ attempts to solve the same linear system, independent from m_1^i and m_2^i, with 2 equations and 4 variables (i.e., x, y, p and q), which is infeasible. As such, $\mathcal{A}$ is not able to forge $x + m^*y + pm^*y + q$. Thus, PIMA satisfies the unforgeability property. □

3.8.2 Unlinkability

Theorem 2 (Unlinkability)**.** The PIMA system satisfies the unlinkability requirement, with respect to *multi-transactions* and *to-original* unlinkability.

Proof. Let us start with the *multi-transactions* unlinkability presented in Section 3.5.3. First, $\mathcal{A}$ is allowed to query, as many times as he wants, the PIMA.`Modify` oracle on the same message m^* and modifications MOD^* for two service providers $\mathcal{SP}_1$ and $\mathcal{SP}_2$. We suppose that the modification of the message m^* results in a one-block message m (i.e., $MOD^*(m^*) = m$). Thus, for each session i, $\mathcal{A}$ receives the sanitized signatures $\sigma_1'^i$ for $\mathcal{SP}_1$ and $\sigma_2'^i$ for $\mathcal{SP}_2$. The signatures can be respectively parsed as

$$\begin{cases} \sigma_1'^i = (s_{1u} = S_1, \tilde{a_{1u}}^i = (h_u)^{r'_{1_i} S_1}, \tilde{d_{1u}}^i = (h_u)^{r'_{1i}(E+\rho_{1i}F)}) \\ \sigma_2'^i = (s_{2u} = S_2, \tilde{a_{2u}}^i = (h_u)^{r'_{2i} s_{2u}}, \tilde{d_{2u}}^i = (h_u)^{r'_{2i}(E+\rho_{2i}F)}) \end{cases}$$

with $E = x + my + mpy + q$, $F = \alpha x + m\beta y$, $h_u = g^{v_u}$ where $v_u \in \mathbb{Z}_q^*$.

We suppose that $\mathcal{A}$ is extremely strong that he knows the exponents of the signature's elements, and gets access to the following linear system of $4i$ equations and $4i + 3$ variables.

$$A_{1i}/S_1 = v_u r'_{1i}, \forall i \qquad (3.10) \qquad A_{2i}/S_2 = v_u r'_{2i}, \forall i \qquad (3.12)$$

$$D_{1i} = v_u r'_{1i}(E + \rho_{1i}F), \forall i \qquad (3.11) \qquad D_{2i} = v_u r'_{2i}(E + \rho_{2i}F), \forall i \qquad (3.13)$$

Afterwards, $\mathcal{A}$ has access to a left-or-right LoRMT that returns two modified signatures σ'_1 for $\mathcal{SP}_1$ and σ'_2 for $\mathcal{SP}_2$, over the same message m^* and modifications MOD^*. The two modified signatures can be respectively parsed as:

$$\begin{cases} \sigma'_1 = (s_{1u} = S_1, \tilde{a_{1u}} = (h_u)^{r'_1 S_1}, \tilde{d_{1u}} = (h_u)^{r'_1(E+\rho_1 F)}) \\ \sigma'_2 = (s_{2u} = S_2, \tilde{a_{2u}} = (h_{2u})^{r'_2 S_2}, \tilde{d_{2u}} = (h_{2u})^{r'_2(E+\rho_2 F)}) \end{cases}$$

with $h_{2u} = g^{v_{2u}}$, i.e., $v_{2u} \in \mathbb{Z}_q^*$.

To break the *multi-transactions* unlinkability property, $\mathcal{A}$ should compare the values of v_u and v_{2u} and if $v_u = v_{2u}$, $\mathcal{A}$ can deduce that the two modified signatures come from the same signature, else from two different signatures. Thus, $\mathcal{A}$, being a strong adversary, establishes the following linear system and tries to make the right choice, while relying on the previous sessions results.

$$A_1/S_1 = v_u r'_1 \quad (3.14) \qquad A_2/S_2 = v_{2u} r'_2 \quad (3.16)$$

$$D_1 = v_u r'_1 (E + \rho_1 F) \quad (3.15) \qquad D_2 = v_{2u} r'_2 (E + \rho_2 F) \quad (3.17)$$

Combining the two linear systems, $\mathcal{A}$ attempts to solve a linear system with $4i + 4$ equations and $4i + 8$ unknown variables, i.e., r'_1, r'_2, ρ_1, ρ_2, E, F, v_u, v_{2u}, r'_{1i}, r'_{2i}, ρ_{1i} and ρ_{2i} $\forall i$, which is unfeasible.

For the *to-original* unlinkability, $\mathcal{A}$ tries to link sanitized signatures to their original signature, while relying on different sessions. That is, $\mathcal{A}$ first, receives two signatures σ_0 and σ_1 over the same one-block message m^*, for the same user ($\mathcal{U}$). The signatures σ_0 and σ_1 can be parsed as:

$$\begin{cases} \sigma_0 = (s_u, a_{1u} = (h_{1u})^{s_u}, b_{1u} = (h_{1u})^{E}, c_{1u} = (h_{1u})^{F}) \\ \sigma_1 = (s_u, a_{2u} = (h_{2u})^{s_u}, b_{2u} = (h_{2u})^{E}, c_{2u} = (h_{2u})^{F}) \end{cases}$$

with $E = x + m^* y + m^* p y + q$, $F = \alpha x + m^* \beta y$, $h_{1u} = g^{v_1}$ and $h_{2u} = g^{v_2}$ where $v_1, v_2 \in \mathbb{Z}_{||}^*$.

$\mathcal{A}$ is allowed to query, as many times as he wants, the PIMA.`Modify` oracle on the same message m^* and modifications MOD^* and the signatures σ_0 and σ_1. We suppose that $MOD(m^*) = m^*$, so that for each session i, $\mathcal{A}$ receives, two modified signatures σ'^i_1 from σ_1 and σ'^i_2 from σ_2, that can be respectively parsed as:

$$\begin{cases} \sigma'^i_1 = (S = r_s s_u, \tilde{a_{1u}}^{\,i} = (h_{1u})^{r'_{1i} S}, \tilde{d_{1u}}^{\,i} = (h_{1u})^{r'_{1i}(E+\rho_{1i} F)}) \\ \sigma'^i_2 = (S = r_s s_u, \tilde{a_{2u}}^{\,i} = (h_{2u})^{r'_{2i} S}, \tilde{d_{2u}}^{\,i} = (h_{2u})^{r'_{2i}(E+\rho_{2i} F)}) \end{cases}$$

We consider $\mathcal{A}$ as a strong adversary that can compute the exponents of b_{1u}, c_{1u},

b_{2u} ,c_{2u}, $\tilde{a_{1u}}^i$, $\tilde{a_{2u}}^i$, $\tilde{d_{1u}}^i$ and $\tilde{d_{2u}}^i$. As such, $\mathcal{A}$ knows the following linear system of $4i+4$ equations and $4i+4$ variables, where the first four equations with 4 variables gives an infinite number of solutions (v_1, v_2) with $v_2 = B_2/B_1 v_1 = C_2/C_1 v_1$.

$$B_1 = v_1 E \quad (3.18) \qquad A_{1i}/S = v_1 r'_{1i}, \forall i \quad (3.22)$$

$$C_1 = v_1 F \quad (3.19) \qquad D_{1i} S/A_{1i} = E + \rho_{1i} F, \forall i \quad (3.23)$$

$$B_2 = v_2 E \quad (3.20) \qquad A_{2i}/S = v_2 r'_{2i}, \forall i \quad (3.24)$$

$$C_2 = v_2 F \quad (3.21) \qquad D_{2i} S/A_{2i} = E + \rho_{2i} F, \forall i \quad (3.25)$$

Afterwards, $\mathcal{A}$ has access to a left-or-right oracle LoRTO that returns a modified signature σ'_b from either σ_1 or σ_2. The signature can be parsed as $\sigma'_b = (S = r_s s_u, \tilde{a_{bu}} = (h_{bu})^{r'_b S}, \tilde{d_{bu}} = (h_{bu})^{r'_b (E + \rho_b F)})$ with $h_{bu} = g^{v_b}$. The strong adversary $\mathcal{A}$ then knows the following two equations.

$$A_b/S = v_b r'_b \quad (3.26) \qquad D_b S/A_b = E + \rho_b F \quad (3.27)$$

To successfully link the modified signature to the associated origin, $\mathcal{A}$ should determine whether $v_b = v_1$ or $v_b = v_2$. As such, $\mathcal{A}$ tries to solve the linear system with $4i+6$ equations and $4i+7$ unknown variables, i.e., r'_b, ρ_b, v_b, E, F, v_1, v_2, r'_{1i}, r'_{2i}, ρ_{1i} and ρ_{2i} $\forall i$, which is unfeasible.

We conclude that PIMA satisfies the unlinkability property. □

3.8.3 Strong Privacy

Theorem 3 (Strong Privacy)**.** The PIMA system satisfies the strong privacy requirement, with respect to the invisibility property of the $\mathcal{UMS}$ scheme.

Proof. We first detail the support of the privacy property. Then, we prove that PIMA guarantees the **strong** privacy.

In a nutshell, the notion of privacy is related to the (i) indistinguishability of signatures and (ii) the anonymity of the originator.
(i) The indistinguishability of signatures is inherited from the multi-transactions unlinkability property satisfied by PIMA, as proven in Theorem 2.
(ii) For the anonymity of the originator, we consider an adversary $\mathcal{A}$ that is allowed to request, as many times as he wants, the PIMA.`Modify` on the same message m^* and the modifications MOD^* for two users $\mathcal{U}_0$ and $\mathcal{U}_1$. Note that for each session, $\mathcal{A}$

receives randomized versions of the users' public keys and pseudonyms (i.e., $\mathcal{A}$ has no access to the original public keys and pseudonyms of $\mathcal{U}_0$ and $\mathcal{U}_1$). $\mathcal{A}$ is then given a new sanitized signature on the message m^* and modifications MOD^* either for user $\mathcal{U}_0$ or user $\mathcal{U}_1$. $\mathcal{A}$ outputs a bit $b \in \{0,1\}$ and wins the game if he can successfully guess with a probability greater than $\frac{1}{2}$ the user originating the modified signature. Let us emphasize that thanks to the randomization of both public key and pseudonym, $\mathcal{A}$ is not able to decide which user originated the signature with a probability greater than $\frac{1}{2}$.

For the **strong** privacy feature, we show that a curious provider should not be able to decide which data has been modified or deleted. Indeed, $\mathcal{A}$ can query the PIMA.`Modify` on a same message m^* and different modifications $MOD_i \in ADM^*$. Note that the message m^* is secret from the adversary, i.e., $\mathcal{A}$ has only access to admissible modifications ADM^*.
For a given challenge pairs of modifications MOD_0 and MOD_1 on the same message m^*, such that $\{MOD_0, MOD_1\} \notin \bigcup_i MOD_i$ and $|MOD_0(m^*)| = |MOD_1(m^*)|$, $\mathcal{A}$ should be able to decide which modifications are applied on the message m^*. $\mathcal{A}$ receives a new message $m_b^{'*} = MOD_b(m^*)$, such that $m_b^{'*} \notin \{MOD_i(m^*)\}$. As such, $\mathcal{A}$ has access to a new message that has not been given before and there is no combination of $\{MOD_i(m^*)\}$ that allows to derive $m_b^{'*}$. Thus, $\mathcal{A}$ is not able to successfully guess the modifications that have been applied with a probability greater than $\frac{1}{2}$. As such, PIMA satisfies the strong privacy property. □

3.9 Performance Analysis

This section presents a proof of concept of PIMA including a full implementation of the different algorithms. It, first, introduces PIMA test-bed and then, discusses the theoretical and experimental results for the five algorithms, as depicted in Table 3.2.

Table 3.2: Communication, Storage and Computation Costs of PIMA's Algorithms

Algorithm	Running device	Communication cost	Storage cost	Computation cost	Computation time in ms		
					$l=10$	$l=50$	$l=100$
PIMA.`Setup`	Device 1 ($\mathcal{IP}$)	$\|\mathbb{Z}_q\|+\|\mathbb{G}_1\|+(3+2l)\|\mathbb{G}_2\|+\|\mathbb{G}_3\|$ [a]	$\|\mathbb{Z}_q\|+\|\mathbb{G}_1\|+'3+2l)\|\mathbb{G}_2\|+\|\mathbb{G}_3\|$ [b]	$\gamma_G+(2+l)\mathrm{E}_{\mathbb{G}_2}$	6	247	478
PIMA.`UserKeyGen`	Device 1 ($\mathcal{IP}$)	($\mathcal{IP}$-$\mathcal{U}$): $2\|\mathbb{Z}_q\|+2\|\mathbb{G}_2\|$	($\mathcal{IP}$/$\mathcal{U}$): $2\|\mathbb{Z}_q\|+2\|\mathbb{G}_2\|$	$2\mathrm{E}_{\mathbb{G}_2}$	6.3	6.3	6.3
PIMA.`Sign`	Device 1 ($\mathcal{IP}$)	($\mathcal{IP}$-$\mathcal{U}$): $(l+1)\|\mathbb{Z}_q\|+3\|\mathbb{G}_1\|$	($\mathcal{IP}$/$\mathcal{U}$): $(l+1)\|\mathbb{Z}_q\|+3\|\mathbb{G}_1\|$	$3\mathrm{E}_{\mathbb{G}_1}$	2.6	3.3	4
PIMA.`Modify`	Device 2 ($\mathcal{U}$)	($\mathcal{U}$-$\mathcal{SP}$): $(l-s+1)\|\mathbb{Z}_q\|+5\|\mathbb{G}_1\|+2\|\mathbb{G}_2\|$	($\mathcal{SP}$): $(l-s+1)\|\mathbb{Z}_q\|+5\|\mathbb{G}_1\|+2\|\mathbb{G}_2\|$	$(3s+8)\mathrm{E}_{\mathbb{G}_1}$	15	33	57.7
PIMA.`Verify`	Device 1 ($\mathcal{SP}$)	–	–	$(l\text{-}s)\mathrm{E}_{\mathbb{G}_2}+\mathrm{E}_{\mathbb{G}_3}+6\mathrm{P}$	13.6	64.4	130

NOTE: $|\mathbb{G}_1|$ (resp. $|\mathbb{G}_2|$, $|\mathbb{G}_3|$ and $|\mathbb{Z}_q|$) indicates the size of an element in $\mathbb{G}_1$ (resp. $\mathbb{G}_2$, $\mathbb{G}_3$ and $\mathbb{Z}_q$); E and P stand for group exponentiations and pairing costs respectively; γ_G is the cost of the cyclic group generation; [a] indicates that the communication cost is between $\mathcal{IP}$ and each of the system's other entities; [b] indicates that the storage cost concerns all the system entities; s is the number of blocks removed.

3.9.1 Test-bed and Methodology

All the algorithms[6] have been implemented and lead to several performance measurements relying on two devices which hardware features are given in Table 3.3.

Table 3.3: Hardware Features of The Running Devices

	Type	OS	Processor	RAM
Device 1	Laptop	*Ubuntu* 20.04.1 LTS (64 bits)	*Intel* Core i7 @1.30 GHz - 8 cores	16 GB
Device 2	Smartphone	*Android 12*	Snapdragon 730 Octa-Core	8 GB

Device 1 is used to test algorithms run by $\mathcal{IP}$ and $\mathcal{SP}$, while Device 2 runs algorithms performed by $\mathcal{U}$. The two devices run *Python* with the associated cryptographic library, supporting bilinear pairings, *bplib* [7]. Note that these two devices are considered as examples of test beds. PIMA algorithms could be also deployed on iOS smartphones. The implementation relies on a bilinear elliptic curve group of 616-bits group order, which corresponds approximately to a 308-bit security level. For accurate measurements of the processing time, each algorithm is run 100 times, while considering a standard deviation of an order 10^{-2}.

In our experiments, we consider two types of tests. In the first one, we consider l-blocks messages where l is equal to 10, 50 and 100, respectively. Each block is composed of 30 characters.
In the second test, we vary the the number of message blocks from 4 to 100 blocks to study the impact of variation on the computation time.
All the experiments refer to an amount of admissible blocks equal to 50% and an amount of modified blocks equal to 25% of all blocks (i.e., 50% of admissible blocks).

3.9.2 Communication and Storage Costs

This section discusses the communication and storage costs of PIMA. As depicted in Table 3.2, both costs are evaluated according to the size of group elements $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ and $\mathbb{Z}_q$.

Table 3.2 shows that the SETUP phase presents communication and storage costs of $3|\mathbb{Z}_q|+|\mathbb{G}_1|+(5+2P)|\mathbb{G}_2|+|\mathbb{G}_3|$ to share and store the system public parameters and the entities' keys.
The IDENTITY_ISSUE, including only the PIMA.`Sign` algorithm, introduces communication and storage costs that vary w.r.t. to the number of message blocks. The identity provider is required to have a storage capacity of $(l+1)|\mathbb{Z}_q|+3|\mathbb{G}_1|$ per user.
The REQUEST_SERVICE phase, including the PIMA.`Modify` and PIMA.`Verify` algorithms, offers acceptable communication and storage costs. Indeed, each session,

[6]The source code is available at https://github.com/soumasmoudi/malleable_unlinkable_sig
[7]https://pypi.org/project/bplib/

performed between the user and the service provider, introduces a communication overhead of $(l - s + 1)|\mathbb{Z}_q|+5|\mathbb{G}_1|+2|\mathbb{G}_2|$. The service provider is required to have a storage capacity of the same size. This storage memory is discharged once the PIMA.`Verify` algorithm is performed.

3.9.3 Computation Overhead

This section discusses the theoretical and experimental processing costs of PIMA's algorithms. Note that the computation cost of group element multiplication is ignored as it is considered as negligible compared with the exponentiation and pairing computation costs.

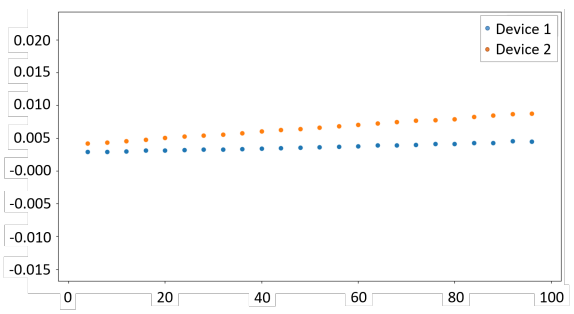
From Table 3.2, it is worth mentioning that, except from PIMA.`UserKeyGen` and PIMA.`Sign` algorithms, the communication cost varies w.r.t. to the message length l (i.e., for the PIMA.`Setup` algorithm), the number of blocks removed (i.e., for the PIMA.`Modify` algorithm) or both (i.e., for the PIMA.`Verify` algorithm). The PIMA.`Sign` algorithm algorithm is very efficient as the computation cost does not depend on the message length l, i.e., only 3 exponentiations are required. The PIMA.`Verify` algorithm has a constant computational cost in terms of pairing operations (i.e., 6 pairing operations) as all the attributes are certified in a single signature.

Table 3.2 shows that the tests' results are fully in line with the theoretical computation costs. Indeed, the generation of the system public parameters and the $\mathcal{IP}$'s keys requires a computation time of $6ms$ (resp. $247ms$ and $478ms$) for a number of blocks $l = 10$ (resp. $l = 50$ and $l = 100$), on Device 1. The computation time varies w.r.t. the number of weights (attributes' types) supported by $\mathcal{IP}$. Note that in our experiments, we consider that each message has a different type, thus the number of weights is equal to l.
The generation time of the user's key pair remains constant, regardless of the number of blocks. It reaches $6.3ms$ on Device 1.

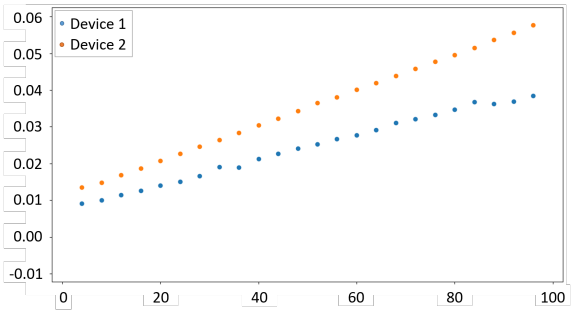
Figure 3.3 shows the impact of varying the message length from 4 to 100 blocks on the computation time of PIMA.`Sign`, PIMA.`Modify` and PIMA.`Verify` algorithms. This impact is studied on the two devices.

From Figure 3.3, it is worth stating that the computation times of PIMA.`Sign`, PIMA.`Modify` and PIMA.`Verify` algorithms are linear functions of the blocks' number, with different slopes. Figure 3.3a shows that the time for generating a signature (i.e., by $\mathcal{IP}$) varies from $2ms$ to $4ms$ on Device 1 and from $4ms$ to $8.7ms$ on Device 2, which is low and can satisfy many practical use cases. The obtained results vary with the number of blocks since the signing algorithm, performed on an l blocks message, involves only 3 modular exponentiations and l multiplications. With consideration to the keys' generation and the message signature times, we deduce that PIMA is efficient and practical at identity providers' side.

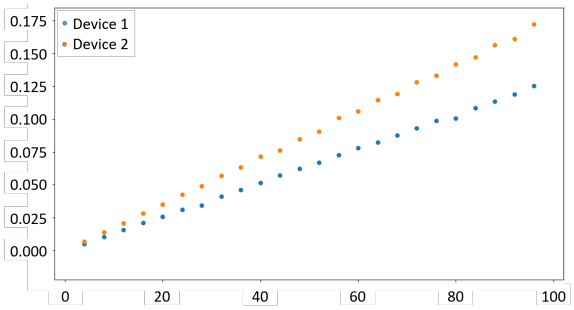
Figure 3.3b shows that the computation time for modifying 25% of a message is increasing significantly, varying from $9ms$ to $40ms$ on Device 1 and from $13.5ms$ to



(a) Pima Signature Processing Time



(b) Pima Modification Processing Time



(c) Pima Verification Processing Time

Figure 3.3: Computation Time (in ms) with a Number of Blocks Varying from 4 to 100

57.7ms on Device 2. Indeed, as reported in Table 3.1, the computation of the new signature requires $3s + 8$ modular exponentiations, where s is the number of modified blocks (i.e., 25% of l). Despite the increasing computation time, a very good efficiency for Pima can be observed on the `Modify` operation, even when running on a smartphone, thus confirming the utility of Pima.

Finally, in Figure 3.3c, the processing time to verify the validity of a signature over an $(l - s)$-blocks message increases with a slope steeper than the Pima.`Modify`'s one, which confirms the computational costs given in Table 3.2). This results in a computation time varying from 5ms to 130ms on Device 1 and from 6ms to 172ms on Device 2, due to $(l - s)$ modular exponentiations and 6 pairings.

3.10 Conclusion

To protect users from loosing control over their identities, and preventing service providers from building a precise profile by linking their transactions, this chapter proposes Pima, a privacy-preserving and user centric identity management system based on a new unlinkable malleable signature [94]. The proposed system enables a user to select and hide the attributes he does not want to reveal, while proving to a service provider that the disclosed information is certified by a trusted entity in each pseudonymous session. Thus, service providers are able to authenticate the origin of the data in order to authorize users to access to services. Additionally, Pima addresses a critical privacy concern, i.e., unlinkability, thanks to our proposed variants

of the PS scheme which supports randomness of both the signature and the user's keys. Experimental results show the practical usability of our solution through processing times that do not exceed few milliseconds for a message of 100 attributes, on an Android-12 smartphone.

However, in some cases, proving the possession of certified attributes is not sufficient to access services and resources (e.g., attendance management at companies, financial transactions, etc.). Service providers need to verify that attributes belong to the physical person who is performing the transaction, which refers to peer-entity authentication. Thus, in the next chapter, we will address users' authentication concerns in identity management systems, with a great interest to biometric identities. For this purpose, we propose a novel privacy-preserving biometric authentication scheme based on malleable signatures.

Chapter 4

A Privacy-preserving and User centric Biometric Authentication Protocol through Malleable Signatures

There are no such words like "over-dreaming" or dreaming without "biometric verification". You can dream over and over again! You don't need a certificate to dream big!

– Israelmore Ayivor

With regard to the travelling experience illustrated in the Introduction, we present, in this chapter, a solution that allows Alice to prove, to the airport or the airline company agents, the validity of her vaccination certificate without disclosing identifiable information like name and date of birth. Indeed, the proposed solution enables Alice to get vaccination certificate that is associated to her biometric identity (e.g., her fingerprint). The vaccination certificate only proves that the certificate's owner is fully vaccinated. Before her trip, Alice should enroll separately at the airport and the airline company using randomized versions of her vaccination certificate and the associated biometric identity. Then, at the airport, Alice only needs to present her fingerprint on the biometric sensors of the airport (resp. the airline company) that compares it with the registered one. As such, Alice proves the ownership of a valid vaccination certificate without harming her privacy, i.e., Alice remain anonymous during the whole process and unlinkable between the airport and the airline company.

4.1 Introduction

Biometric-based authentication represents a significant and emerging field of research that ensures both security and usability in different real-world applications. Indeed, biometric identities (e.g., fingerprint, iris, voice, face) are used for controlling physical access to secured areas, unlocking smartphones and installed applications, multi-factor authentication for smartphones' transactions, facilitating the user experience at the airport (e.g., check-in, luggage screening and boarding), etc. The widespread use of biometrics is motivated by their ability to prove the link to physical persons, which is required by some strong authentication methods. However, biometric traits are very sensitive personal data, strongly linked to a physical person and which are non revocable. Thus, they are subject to specific regulations, e.g., California Consumer Privacy Act (CCPA) [86] and California Privacy Rights Act (CPRA) [87] in the U.S and General Data Protection Regulation (GDPR) in the E.U. [57]. From a security perspective, in case of a compromised biometric dataset (i.e., biometric traits are stored in a clear way), it is not possible to revoke users' biometric data, which may lead to long-lasting impersonation attacks against legitimate users.

To deal with the aforementioned issues, different solutions have been proposed namely biometric templates. These templates are represented as vectors or matrices involving numerical data extracted from users' biometric data. Afterwards, different cryptographic tools have been applied to protect the privacy of biometric templates referred to as biometric template protection techniques. These tools are classified into three categories namely cancellable biometrics [13, 39, 118, 110], biometric cryptosystems [38, 65, 117] and biometrics in the encrypted domain [90, 63, 115, 150, 69, 146]. The first two techniques have a critical drawback as the verification phase requires some auxiliary sensitive data, referred to as helper data generated during the enrollment phase. Helper data allows to extract keys used for verification, but can be also used to reconstruct original biometric data [117]. This data can be also retrieved if secret keys used in cancellable biometric schemes are compromised [110]. Thus, biometrics in the encrypted domain is considered as an alternative to these two techniques. For instance, allowing to perform computation in the encrypted domain, homomorphic encryption schemes are mainly used for protecting databases' confidentiality [91], in particular for biometric templates databases. Several homomorphic encryption-based biometric authentication systems have been proposed in the literature that can be categorized into two types, namely the server-centric and the user-centric models.
In the server-centric model, the service provider first stores an encrypted form of the biometric template. When providing a fresh template, the user is responsible for retrieving the encrypted template from the service provider and for homomorphically computing an encrypted matching score between the two templates, referred to as a matching distance. Note that, the service provider's key is used for the encryption such that it is able to decrypt the distance and to decide whether to authenticate the user or not. In the server-centric, the users' privacy is not preserved. Indeed, the service provider has the full control over users' biometric templates and he is able to cross-check with other service providers this information to trace users.
The user-centric model suggests that the biometric template is encrypted by the user's

key. The service provider is responsible for homomorphically computing a distance between two encrypted templates and only the user is able to decrypt the distance. While this approach allows to protect users' privacy, additional requirements arise including (i) the need to verify that the user correctly decrypts the distance, i.e., considering the user as non-trusted aims at bypassing the authentication process and (ii) the enhancement of users' privacy while preventing the linkability between their transactions (i.e., biometric identities) among several service providers.

To address these issues, we present our second contribution [95], named PUBA. With respect to the taxonomy presented in Chapter 2 Section 2.2.3.5, this contribution is considered as a pseudonymous centralized user-centric identity management solution, supporting authentication, and belonging to the verifiable credentials category (cf. Figure 4.1).

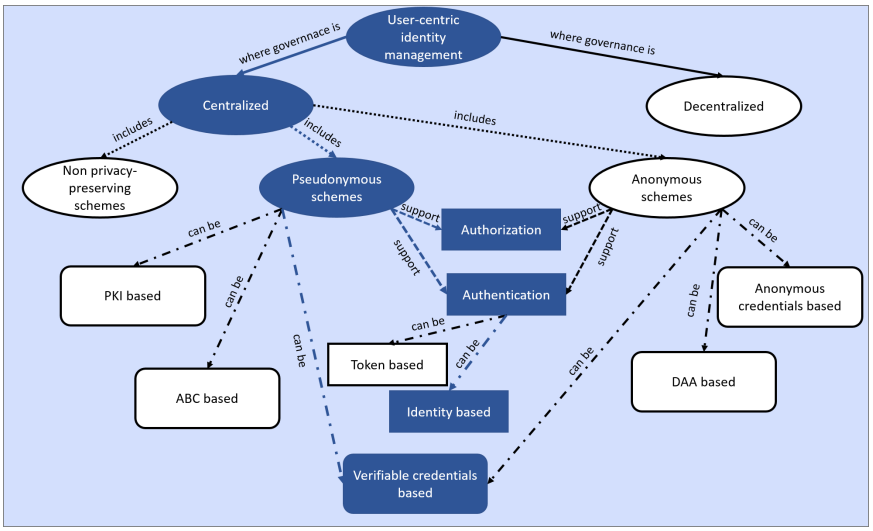


Figure 4.1: PUBA Features w.r.t. The Taxonomy of User-centric Identity Management Systems

Indeed, we propose a privacy-preserving biometric authentication system designed for user-centric identity management. While being physically present at an identity provider, the user, first, receives an encrypted and certified biometric template over his biometric traits, referred to as credential. This credential is then used to remotely enroll at service providers. The user is able to modify the encrypted template and the certificate such that he obtains a different credential for each service provider. To access services, the user provides a fresh biometric template, at the service provider's desk (i.e., biometric sensors). This template is homomorphically compared to the encrypted one by the service provider, with the help of the user. After verifying the correctness of the computation performed by the user, the service provider decides whether to authenticate the user or not. Note that the biometric templates extraction step is not considered in this chapter. We assume that the biometric template is defined as a fixed-length vector that contains integers.

The contributions of this chapter are summarized as follows:

- we design a biometric authentication scheme that enables to verify users' computations in the encrypted domain, i.e., verifying both the validity of users' credentials and the correctness of computation carried out by users on matching scores.

- we detail the concrete construction of the main building blocks and the different algorithms and procedures of the PUBA protocol, relying on the El Gamal homomorphic encryption scheme and a malleable signature scheme built upon Pointcheval-Sanders signature scheme. Through a detailed security analysis, the proposed construction of PUBA is proven to guarantee privacy properties namely *(i)* the unlinkability of users' transactions (i.e., enrollment sessions) among several service providers (i.e., users' modified credentials cannot be linked to each other), and *(ii)* the anonymity of users towards colluding system entities (i.e., identity provider and service providers are not able to link users' modified credentials to the original version).

- we evaluate the performances of PUBA through the implementation of different phases and algorithms. The conducted experiments have demonstrated acceptable computation times.

The remainder of this chapter is organized as follows. Section 4.2 presents a real-world use case and enumerates the design goals. Section 4.3 compares PUBA to closely-related work. Section 4.4 gives an overview of PUBA and presents the identified threat model. After introducing the main building blocks in Section 4.5, Section 4.6 provides a full construction of PUBA through detailed algorithms. A formal security analysis is given in Section 4.7. A detailed discussion of PUBA conducted experiments is provided in Section 4.8 before concluding in Section 4.9.

4.2 Motivation Through a Delivery Use Case

In this section, we present a practical use case scenario as a complement to the vaccination certificate verification scenario described in the preliminaries of the chapter. The scenario refers to a delivery use case.

Indeed, with the development of food delivery services, several platforms, like Uber Eats, Deliveroo and Stuart, have been created to enable people to order their favorite meals without moving. These platforms hire people to make speedy delivery for consumers. These people need to register at delivery platforms in order to become riders and deliver customers' orders. For this purpose, they are requested to prove, for instance, a legal residence. Once registered, the verification of the rider's identity is challenging. Indeed, a proof about the physical identity is required to double-check whether it is the right rider who will make the delivery. Recently, a fraudulent business has taken advantage of this lack of control to rent out accounts to illegal immigrants and minors for a fee (30% to 50% of revenues[1]). To mitigate risks, it is of utmost importance to verify the rider's true identity. Indeed, some platforms like Deliveroo

[1]https://www.nytimes.com/2019/06/16/business/uber-eats-deliveroo-glovo-migrants.html

have set up, in partnership with Onfido, a facial recognition system to check in real time the identity of the person who is making the delivery[2]. Indeed, to create a rider account, the individual scans his identity document with his smartphone and takes a reference selfie. Based on Machine Learning (ML) algorithms, a third party service (i.e., Onfido) evaluates the matching score between the photo included in the identity document and the real-time selfie. The account is validated once the verification holds and a numerical reference template of the biometric identity (i.e., the face capture) is stored at the service provider (i.e., Deliveroo and/or Onfido) with context information (e.g., geolocation and time).

The face recognition solution adopted by Deliveroo helps dealing with frauds but it raises various security and privacy issues. For instance, clear biometric identities and personally identifiable information are insecurely exchanged between entities (i.e., the communication channel cannot be trusted). They are also collected and stored at the service providers' side, making them vulnerable to data breach attacks. Finally, people's movements are tracked and linked to each other through several service providers, which harms their privacy.

4.3 Related Work

This section introduces biometric authentication schemes carried out in the encrypted domain and gives a detailed comparison between PUBA and related work.

Several works have been proposed in the literature to ensure the privacy of biometric templates using homomorphic encryption [140, 129, 3, 63, 146, 84, 115]. For instance, Gomez et al. [63] designed a protection scheme for multi-biometric templates using Paillier cryptosystem. In the proposed solution, the template encryption is carried out by the service provider, meaning that the clear template is exposed to the service provider. During the authentication, the matching distance is computed on the user side, which implies important processing capacities by end-users. Yang et al. [146] proposed a biometric authentication scheme where the service provider computes the matching distance between an encrypted reference template and a clear fresh one. To enhance privacy, Kumar et al. [84] suggested that the matching distance is computed over two encrypted templates. Both [146] and [84] schemes neglect the fact that a malicious user sends a fake matching distance, i.e., the user is considered as trusted to decrypt the matching distance. Pradel et al. [115] proposed an biometric authentication protocol where a remote service provider operates on encrypted templates to compute the matching distance. Ibarrondo et al. [69] proposed a face identification scheme based on functional encryption. Indeed, during the enrollment, a functional secret key is generated over the biometric template and shared with the service provider. For authenticating, the user encrypts the fresh template using a master secret key. An inner product is then computed, by the service provider, between the reference and the fresh templates. The inner product between the two templates is obtained through the decryption by the corresponding functional secret key. Hence, it is known to the service

[2]https://riders.deliveroo.co.uk/en/news/any-questions-take-a-look-at-our-frequently-asked

provider. For privacy concerns, Zhou and Ren [150] proposed a biometric authentication solution, named PassBio, based on a threshold predicate encryption scheme (TPE). Indeed, TPE represents an instance of functional encryption where the decryption returns a function over the plaintext instead of outputting the plaintext itself.

All the aforementioned works are built upon trust on the capture modules of biometric traits and the generation of biometric templates on the user side. However, this is not the case in reality. Indeed, malicious users may attempt to provide fake biometric data while enrolling and authenticating, e.g., by replaying previously generated templates.
Furthermore, in the literature, the entity that decrypts the matching distance, i.e., the user or the service provider, is always trusted to correctly carry out the decryption. Hence, the integrity of the matching distance is not ensured. Kumar et al. [99] proposed a solution to verify the integrity of the matching result relying on a trusted entity, called public auditor. The proposed scheme introduces communication overhead and excludes the case of collusion between the public auditor and the service provider. In [115], authors suggest to randomize the encrypted matching distance such that its decryption, by the user, returns a fixed value only known by the service provider. For this purpose, the service provider should know the user' public key that uniquely identifies the user.

Table 4.1 presents a comparison between PUBA and biometric authentication schemes carried out in the encrypted domain in terms of fulfilled security and privacy properties, w.r.t. IDM systems' properties defined in Chapter 2, Sections 2.2.2.1 and 2.2.2.2, namely,

- *(i)* **unforgeability** of biometric information given to enroll at service providers, which refers to the data integrity **(S2)** and authenticity **(S3)** properties,
- *(ii)* **soundness** of authentication when access services and resources (i.e., is not able to impersonate a legitimate user and to successfully authenticate), which refers to the peer-entity authentication property **(S4)**,
- *(iii)* **unlinkability** of users' enrollment sessions (i.e., implies that protected biometric templates of the same user cannot be linked to each other), which refers to the multi-show unlinkability of property **(P4)**, and
- *(iv)* **anonymity** of users towards colluding identity and service providers, which refers to anonymity property **(P1)**.

From a security perspective, Table 4.1 shows that, on the one hand, all the reviewed schemes do not satisfy the unforgeability property. They strongly rely on the security of the device extracting the templates, although biometric systems are vulnerable to several attacks [127]. Fake templates can be then generated and used to enroll at service providers while the forgery cannot be detected. As such, service providers are compelled to have important storage capacities, otherwise their storage servers will be congested with fake templates.
On the other hand, when considering malicious adversaries attempting to impersonate

legitimate users at the verification phase, most of the works do not ensure the authentication soundness. Contrary to [115], [99], other works trust the entity responsible for decrypting the matching distance (i.e., the user or the service provider). Nevertheless, to ensure soundness, [99] introduces a trusted intermediary entity, which weakens the system's security. In [115], the proposed solution compromises users' privacy for the sake of security since users should provide their public keys to service providers.

From a privacy perspective, it is noticeable from Table 4.1 that all the reviewed schemes satisfy only partial unlinkability. That is, they focus only on the unlinkability between encrypted reference biometric templates stored at service providers. The partial unlinkability property is satisfied thanks to the randomness of the implemented encryption scheme, i.e., the same message, encrypted twice, gives two different ciphertexts. However, biometric templates are often associated with other information, like unique identifiers or public keys. Thus, service providers are able to link several enrollment sessions to the same user. For the same reason, these works do not ensure users' anonymity even without considering colluding curious entities.

We deduce that none of the reviewed ensures privacy-reserving biometric authentication of users with no risk of using fake biometric template and bypassing the authentication process. Thus, in PUBA, we propose a novel biometric authentication scheme for identity management systems, where users' privacy is protected and the authentication soundness is ensured.

In the following, we will describe the new design of PUBA. We will formally define the desired properties and prove their fulfillment based on a concrete construction and standard assumptions. The efficiency of PUBA will be also proven through a detailed performance analysis.

Table 4.1: Comparison between PUBA and Related Work

		PUBA	[63]	[146]	[84]	[115]	[150]	[69]	[99]
Security properties	Unforgeability	✓	✗	✗	✗	✗	✗	✗	✗
	Soundness	✓	✗	✗	✗	✓	✗	✗	✓
Privacy properties	Unlinkability	✓	✓[a]	✓[a]	✓[a]	✓[a]	✓[a]	✓[a]	✓[a]
	Anonymity	✓	✗	✗	✗	✗	✗	✗	✗

NOTE: [a] indicates that only partial unlinkability is satisfied, i.e., the unlinkability between encrypted biometric templates.

4.4 System and Threat Models

In this section, we give a high level description of PUBA while presenting the involved entities and the different phases of the solution. Then, we give formal definitions of the security and privacy properties.

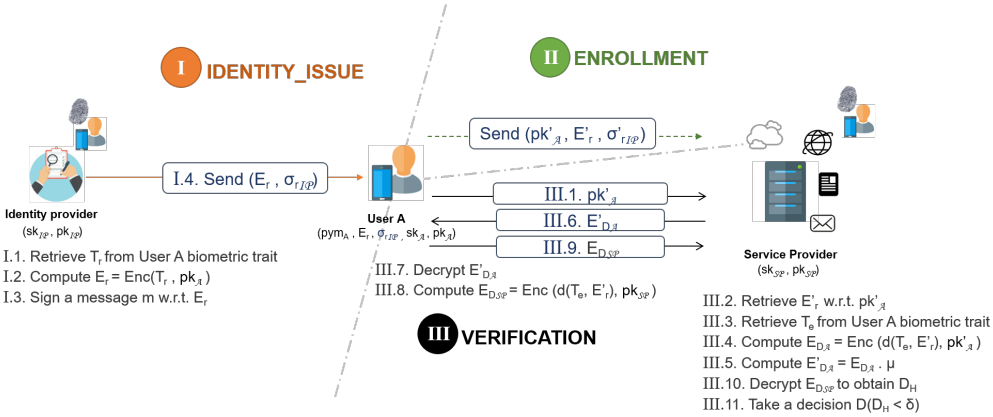


Figure 4.2: Overview of PUBA

NOTE: E_r (resp. E'_r) is encrypted with $\mathtt{pk}_{\mathcal{A}}$ (resp. $\mathtt{pk}'_{\mathcal{A}}$); $E_{D_{\mathcal{A}}}$ and $E'_{D_{\mathcal{A}}}$ are encrypted with $\mathtt{pk}'_{\mathcal{A}}$; $E_{D_{\mathcal{SP}}}$ is encrypted with $\mathtt{pk}_{\mathcal{SP}}$; User A is physically present at the identity provider (resp. the service provider) during the IDENTITY_ISSUE phase (resp. the VERIFICATION phase).

4.4.1 System Overview

PUBA is a privacy-preserving biometric authentication solution that involves three main entities namely the user ($\mathcal{U}$), the identity provider ($\mathcal{IP}$) and the service provider ($\mathcal{SP}$), as depicted in Figure 4.2. $\mathcal{U}$ receives, from $\mathcal{IP}$, an encrypted reference template E_r of her numerical biometric template T_r associated with a signed credential $\sigma_{r_{\mathcal{IP}}}$, during the IDENTITY_ISSUE phase. $\mathcal{U}$ locally stores the couple $(E_r, \sigma_{r_{\mathcal{IP}}})$ received from $\mathcal{IP}$. Note that T_r is encrypted using the user's public key. For high assurance authentication, we assume that the IDENTITY_ISSUE phase is performed at the $\mathcal{IP}$. Indeed, the user is required to be present at $\mathcal{IP}$'s office and prove his identity through a legal document (e.g., passport), in order to obtain his biometric identity.

Afterwards, $\mathcal{U}$ is able to derive several credentials from the couple $(E_r, \sigma_{r_{\mathcal{IP}}})$ in order to remain unlinkable when enrolling at different $\mathcal{SP}$s. Indeed, during the ENROLLMENT phase, $\mathcal{U}$ sends a randomized tuple $(E'_r, \sigma'_{r_{\mathcal{IP}}})$ to $\mathcal{SP}$. A randomized version of the user's encryption key is also shared with $\mathcal{SP}$ considered as a user's pseudonym at $\mathcal{SP}$. $\mathcal{SP}$ verifies the signature and stores the given credentials and key.

During the VERIFICATION phase, the user is invited to be present at the $\mathcal{SP}$'s desk to provide a fresh template, called extracted template T_e. Note that T_e is associated to the user's public key in order to allow $\mathcal{SP}$ to retrieve the registered reference template. Then, $\mathcal{SP}$ homomorphically computes the distance between the registered template and the fresh one. $\mathcal{SP}$ provides a randomized form of the calculated encrypted to $\mathcal{U}$ that blindly re-encrypts it with the $\mathcal{SP}$'s key. If $\mathcal{SP}$ verifies that $\mathcal{U}$ has correctly re-encrypted the matching distance, he takes the decision whether to accept the user as successfully authenticated while comparing the distance to a predefined threshold.

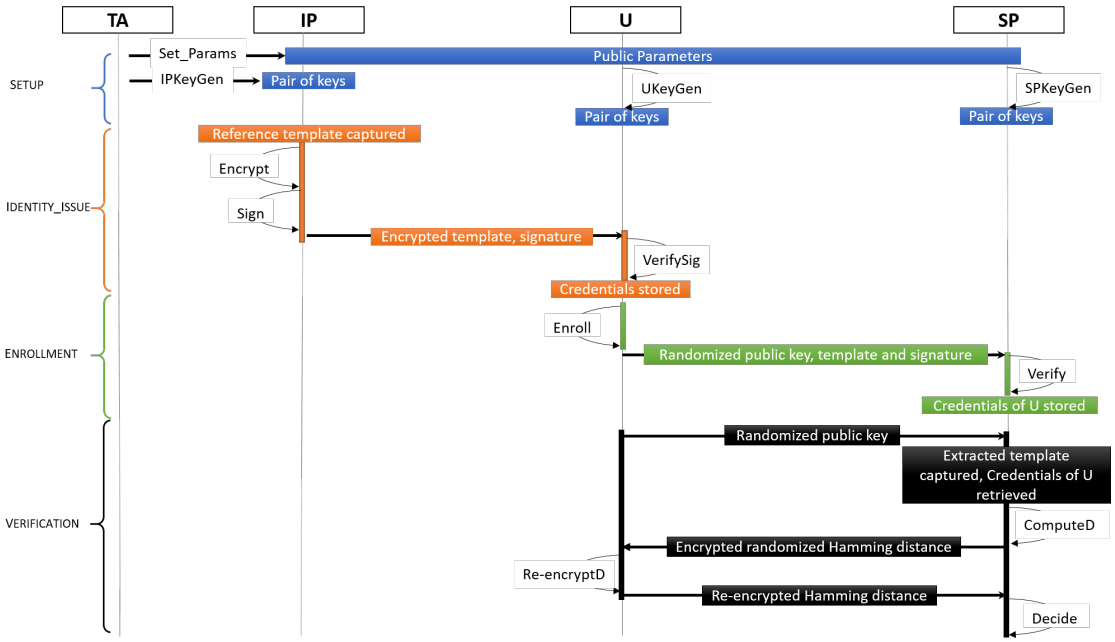


Figure 4.3: Workflow of PUBA

4.4.2 System Phases

PUBA involves four phases, referred to as SETUP, IDENTITY_ISSUE, ENROLLMENT and VERIFICATION. The four phases include twelve algorithms that are chronologically presented in Figure 4.3.

SETUP – This phase consists of initializing the system, setting up the global parameters and generating the keys of the involved entities. It relies on four algorithms referred to as $\mathsf{Set\_Params}_{\mathcal{TA}}$, $\mathsf{IPKeyGen}_{\mathcal{TA}}$, $\mathsf{UKeyGen}_{\mathcal{U}}$ and $\mathsf{SPKeyGen}_{\mathcal{SP}}$ and defined as follows.

- $\mathsf{Set\_Params}_{\mathcal{TA}}(\lambda) \rightarrow pp$ – performed by a trusted authority ($\mathcal{TA}$). Relying on the security parameter λ, this algorithm returns the system global public parameters pp. Note that pp will be considered as a default input for all PUBA's algorithms.
- $\mathsf{IPKeyGen}_{\mathcal{TA}}() \rightarrow (\mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}})$ – run by $\mathcal{TA}$. This algorithm returns the key pair $(\mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}})$ of the identity provider.
- $\mathsf{UKeyGen}_{\mathcal{U}}() \rightarrow (\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$ – performed by $\mathcal{U}$. This algorithm returns the key pair $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$ of the user.
- $\mathsf{SPKeyGen}_{\mathcal{SP}}() \rightarrow (\mathtt{sk}_{\mathcal{SP}}, \mathtt{pk}_{\mathcal{SP}})$ – run by $\mathcal{SP}$. This algorithm outputs the pair of keys $(\mathtt{sk}_{\mathcal{SP}}, \mathtt{pk}_{\mathcal{SP}})$ of the service provider.

IDENTITY_ISSUE – This phase consists of issuing a certified identity to $\mathcal{U}$, when being physically present at $\mathcal{IP}$, relying on three algorithms referred to as $\mathsf{Encrypt}_{\mathcal{IP}}$, $\mathsf{Sign}_{\mathcal{IP}}$ and $\mathsf{ComputeSig}_{\mathcal{U}}$ detailed hereafter.

- $\mathsf{Encrypt}_{\mathcal{IP}}(T_r, \mathtt{pk}_{\mathcal{U}}) \rightarrow (E_r, (\rho_1, ..., \rho_n))$ – run by $\mathcal{IP}$ to encrypt the user's biometric reference template T_r[3] given as input with the user' public key $\mathtt{pk}_{\mathcal{U}}$. The $\mathsf{Encrypt}_{\mathcal{IP}}$ algorithm returns an encrypted reference template E_r and a list of randoms $(\rho_1, ..., \rho_n)$ used during the encryption, where n denotes the length of the template T_r.

- $\mathsf{Sign}_{\mathcal{IP}}(m, \mathtt{sk}_{\mathcal{IP}}, \mathtt{A}, t_r, (\rho_1, ..., \rho_n)) \rightarrow (\sigma_{r_{\mathcal{IP}}}, \mathcal{G}, \mathcal{H}_{\mathcal{U}})$ – performed by $\mathcal{IP}$. It takes as input the public message m retrieved from the system global public parameters pp, $\mathcal{IP}$'s secret key $\mathtt{sk}_{\mathcal{IP}}$, a part of the user's secret key $\mathtt{A}$, the initial reference template t_r and the list of randoms $(\rho_1, ..., \rho_n)$ used in the $\mathsf{Encrypt}_{\mathcal{IP}}$ algorithm. The $\mathsf{Sign}_{\mathcal{IP}}$ algorithm returns a signature $\sigma_{r_{\mathcal{IP}}}$ over the message m w.r.t. the reference template T_r and two editing-keys $\mathcal{G}$ and $\mathcal{H}_{\mathcal{U}}$. Note that $\mathcal{G}$ and $\mathcal{H}_{\mathcal{U}}$ will be used for randomizing users' transactions towards several service providers.

- $\mathsf{VerifySig}_{\mathcal{U}}(\mathtt{pk}_{\mathcal{IP}}, E_r, \sigma_{r_{\mathcal{IP}}}) \rightarrow b$ – optionally performed by the user. This algorithm checks the validity of the signature $\sigma_{r_{\mathcal{IP}}}$ w.r.t. the public key $\mathtt{pk}_{\mathcal{IP}}$ of $\mathcal{IP}$ and the encrypted template E_r. It returns $b \in \{0, 1\}$.

Note that at the end of the IDENTITY_ISSUE phase, the user locally stores the signature $\sigma_{r_{\mathcal{IP}}}$, the encrypted reference template E_r and the couple of editing-keys $(\mathcal{G}, \mathcal{H}_{\mathcal{U}})$.

ENROLLMENT – This phase occurs when a user wants to register at a service provider by providing a randomized certified biometric template. It includes two algorithms referred to as $\mathsf{Enroll}_{\mathcal{U}}$ and $\mathsf{Verify}_{\mathcal{SP}}$ defined as follows.

- $\mathsf{Enroll}_{\mathcal{U}}(E_r, \sigma_{r_{\mathcal{IP}}}, \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}}, \mathcal{G}, \mathcal{H}_{\mathcal{U}}, v_1, V_2) \rightarrow (E'_r, \sigma'_{r_{\mathcal{IP}}}, \mathtt{sk}'_{\mathcal{U}}, \mathtt{pk}'_{\mathcal{U}}, \mathcal{G}')$ – performed by $\mathcal{U}$. It takes as input the user's encrypted reference template E_r, the associated signature $\sigma_{r_{\mathcal{IP}}}$, the user's pair of keys $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$, the editing-keys $\mathcal{G}$ and $\mathcal{H}_{\mathcal{U}}$ and two session keys v_1 and V_2. The $\mathsf{Enroll}_{\mathcal{U}}$ algorithm returns a transformed version of the reference template E_r denoted by E'_r (i.e., the transformation consists in re-randomizing and re-keying) and the associated signature $\sigma'_{r_{\mathcal{IP}}}$. It also returns the new randomized pair of the user's keys $(\mathtt{sk}'_{\mathcal{U}}, \mathtt{pk}'_{\mathcal{U}})$ and a randomized editing-key $\mathcal{G}'$. Note that the modified credentials are specific to each $\mathcal{SP}$.

- $\mathsf{Verify}_{\mathcal{SP}}(\mathtt{pk}_{\mathcal{IP}}, E'_r, \sigma'_{r_{\mathcal{IP}}}, v_2) \rightarrow b$ – run by $\mathcal{SP}$ to verify the credentials sent by the user. This algorithm takes as input the $\mathcal{IP}$'s public key $\mathtt{pk}_{\mathcal{IP}}$, the transformed encrypted template E'_r, the signature $\sigma'_{r_{\mathcal{IP}}}$ and a session verifying-key v_2. It returns a bit $b \in \{0, 1\}$ stating whether the signature is valid or not.

Note that if the verification holds, $\mathcal{SP}$ stores the encrypted template E'_r and the associated signature $\sigma'_{r_{\mathcal{IP}}}$ under the user's randomized public key $\mathtt{pk}'_{\mathcal{U}}$.

VERIFICATION – This phase occurs when a user wants to authenticate physically with $\mathcal{SP}$ in order to access to $\mathcal{SP}$'s services. Based on an extracted template T_e

[3]The reference template T_r is computed by $\mathcal{IP}$ w.r.t. an initial reference template t_r of the captured biometric identity.

captured on the $\mathcal{SP}$'s sensors, a matching distance is computed through an interaction between $\mathcal{SP}$ and $\mathcal{U}$. The VERIFICATION phase includes three algorithms, referred to as $\mathsf{ComputeD}_{\mathcal{SP}}$, $\mathsf{ReEncryptD}_{\mathcal{U}}$ and $\mathsf{Decide}_{\mathcal{SP}}$ defined as follows:

- $\mathsf{ComputeD}_{\mathcal{SP}}(E'_r, T_e, \mathtt{pk}'_{\mathcal{U}}, \mathtt{pk}_{\mathcal{SP}}) \rightarrow (E_{D_{\mathcal{U}}}, E'_{D_{\mathcal{U}}}, E_{k_{\mathcal{SP}}}, \gamma, z_1, G_{z_1}, \{\gamma_i\}_{i=1..n})$ – run by $\mathcal{SP}$. It takes as input the encrypted reference template E'_r, the extracted fresh template T_e, the user's randomized public key $\mathtt{pk}'_{\mathcal{U}}$ and $\mathcal{SP}$'s public key $\mathtt{pk}_{\mathcal{SP}}$. This algorithm computes an encrypted Hamming distance $E_{D_{\mathcal{U}}}$ between the two templates and a randomized one denoted by $E'_{D_{\mathcal{U}}}$. The $\mathsf{ComputeD}_{\mathcal{SP}}$ algorithm also returns an encrypted session key $E_{k_{\mathcal{SP}}}$ and a set of elements $(\gamma, z_1, G_{z_1}, \{\gamma_i\}_{i=1..n})$ randomly selected during the execution of the algorithm (i.e., γ and $\{\gamma_i\}_{i=1..n}$ refer to the randoms used for El-Gamal encryption while z_1 and G_{z_1} are randoms characterizing a particular session between $\mathcal{U}$ and $\mathcal{SP}$).

- $\mathsf{ReEncryptD}_{\mathcal{U}}(E'_{D_{\mathcal{U}}}, \mathtt{sk}'_{\mathcal{U}}, E_{k_{\mathcal{SP}}}, \mathcal{G}', G_{z_1}) \rightarrow (E_{D_{\mathcal{SP}}}, \mathcal{G}", G_{z_2})$ – run by $\mathcal{U}$. It takes as input the encrypted and randomized Hamming distance $E'_{D_{\mathcal{U}}}$, the decryption key $\mathtt{sk}'_U$, the encrypted session key $E_{k_{\mathcal{SP}}}$, the list of editing-keys $\mathcal{G}'$ and the session random G_{z_1}. This algorithm returns the encrypted Hamming distance $E_{D_{\mathcal{SP}}}$ between the reference and extracted templates (i.e., the distance is encrypted with $\mathcal{SP}$'s public key), a randomized list of editing-keys $\mathcal{G}$" and a verifying key G_{z_2}.

- $\mathsf{Decide}_{\mathcal{SP}}(E_{D_{\mathcal{SP}}}, E_{D_{\mathcal{U}}}, \mathtt{sk}_{\mathcal{SP}}, \mathtt{pk}_{\mathcal{SP}}, \mathtt{pk}'_{\mathcal{U}}, \mathcal{G}", G_{z_2}, \gamma, z_1, \{\gamma_i\}_{i=1..n}, T_e, \delta) \rightarrow b$ – performed by $\mathcal{SP}$. After verifying the correctness of the distance decryption by $\mathcal{U}$, this algorithm returns a bit $b \in \{0, 1\}$ when comparing the Hamming distance to a predefined threshold δ.

4.4.3 Threat Model

This section identifies the adversaries considered in the proposed authentication solution and gives the formal definitions of the different security and privacy properties.

Two main security and privacy adversaries are identified as follows:

- **A malicious adversary**: attempts to authenticate to a service provider using forged credentials. Malicious adversaries include malicious users and outsiders.

- **A honest but curious adversary**: tries to identify users and link their multiple transactions in order to collect more data about them. Honest but curious adversaries include curious identity and service providers.

Malicious adversaries are considered against security properties, namely unforgeability and soundness, while honest but curious adversaries are considered against privacy requirements, i.e., unlinkability and anonymity.

4.4.3.1 Unforgeability

In PUBA, unforgeability means that it is not possible to generate a valid signature over a biometric template unless the private key of the $\mathcal{IP}$ is known. This can be formally defined in a game $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ where an adversary $\mathcal{A}$, playing the role of a malicious user, has access to Encrypt and Sign oracles. Note that, for each session i, $\mathcal{A}$ gets the encrypted template $E_r^{(i)}$ from the Encrypt oracle and the signature $\sigma_r^{(i)}$ from the Sign oracle over a reference template $T_r^{(i)}$ of his choice. Then, $\mathcal{A}$ chooses a template T_r^* that has neither given before nor obtained by modifying given templates. $\mathcal{A}$ succeeds if it outputs a valid encrypted template/signature pair (E_r^*, σ_r^*) such that the VerifySig verification holds.

Definition 9. Unforgeability – We say that PUBA satisfies the unforgeability property, if for every *PPT* adversary $\mathcal{A}$, there exists a negligible function κ such that:

$$Pr[\mathbf{Exp}_{\mathcal{A}}^{unforg}(1^{\lambda}) = 1] \leq \kappa(\lambda)$$

where $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ is the security experiment against the unforgeability property given below.

$\boldsymbol{Exp}_{\mathcal{A}}^{unforg}(\lambda)$
$pp \leftarrow \mathsf{Set\_Params}(\lambda)$
$(\mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}}) \leftarrow \mathsf{IPKeyGen}()$
$(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}}) \leftarrow \mathsf{UKeyGen}()$
$\mathcal{O} \leftarrow \{\mathsf{Encrypt}(\cdot,\cdot), \mathsf{Sign}(\cdot, \mathtt{sk}_{\mathcal{IP}}, \cdot)\}$
$(E_r^*, \sigma_r^*) \leftarrow \mathcal{A}^{\mathcal{O}}(pp, \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{IP}}, T_r^*)$
letting $T_r^{(i)}$ *denote the queries to* Encrypt *and* Sign
oracles, $E_r^{(i)}$ *and* $\sigma_r^{(i)}$ *the answers from the two*
oracles resp., and T_r^* *was not used before*
If $\mathsf{VerifySig}(\mathtt{pk}_{\mathcal{IP}}, E_r^*, \sigma_r^*) = 1$
return 1
Else return 0

4.4.3.2 Soundness

The soundness property ensures that a non legitimate adversary is not able to authenticate at $\mathcal{SP}$ as a legitimate user. This property is formally defined in a game $\mathbf{Exp}_{\mathcal{A}}^{sound}$ where an adversary $\mathcal{A}$ acting as a malicious outsider, captures the credentials $(E'_r, \sigma_{r_{\mathcal{IP}}}, \mathtt{pk}'_{\mathcal{U}})$ of a user $(\mathcal{U})$ when enrolling at a service provider $(\mathcal{SP})$. $\mathcal{A}$ has also access to ComputeD, ReEncryptD and Decide oracles over extracted templates T_e^i. Note that the same tuple of randoms and verifying keys $(\mu, \gamma, z_1, G_{z_1}, \{\gamma_i\}_{i=1..n}, \mathcal{G}", G_{z_2})$ is used during the whole game. The challenger $\mathcal{C}$, then, chooses an extracted template T_e^* that has not been used before and computes the encrypted distance $E_{D_{\mathcal{U}}}^*$ over E'_r and T_e^*. Given the randomized encrypted Hamming distance $E_{D_{\mathcal{U}}}^{'*}$ and the encrypted

session key $E_{k_{\mathcal{SP}}}$, $\mathcal{A}$ is asked to re-encrypt the Hamming distance and return a valid $E^*_{D_{\mathcal{SP}}}$, such that, when being verified, decrypted by $\mathcal{SP}$ and compared to a predefined threshold, the Decide decision is positive.

Definition 10. Soundness – We say that PUBA satisfies the soundness property, if for every *PPT* adversary $\mathcal{A}$, there exists a negligible function κ such that:

$$Pr[\mathbf{Exp}^{sound}_{\mathcal{A}}(1^\lambda) = 1] \leq \kappa(\lambda)$$

where $\mathbf{Exp}^{sound}_{\mathcal{A}}$ is the security experiment against the soundness property detailed hereafter.

$\boldsymbol{Exp}^{sound}_{\mathcal{A}}(\lambda)$
$pp \leftarrow \mathsf{Set\_Params}(\lambda)$
$(\mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}}) \leftarrow \mathsf{IPKeyGen}()$
$(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}}) \leftarrow \mathsf{UKeyGen}()$
$(\mathtt{sk}_{\mathcal{SP}}, \mathtt{pk}_{\mathcal{SP}}) \leftarrow \mathsf{SPKeyGen}()$
$(E_r, (\rho_1, ..., \rho_n)) \leftarrow \mathsf{Encrypt}(T_r, \mathtt{pk}_{\mathcal{U}})$
$(\sigma_r, \mathcal{G}, \mathcal{H}_{\mathcal{U}}) \leftarrow \mathsf{Sign}(m, \mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{U}}, t_r, (\rho_1, ..., \rho_n))$
$(E'_r, \sigma'_{r_{\mathcal{IP}}}, \mathtt{sk}'_{\mathcal{U}}, \mathtt{pk}'_{\mathcal{U}}, \mathcal{G}') \leftarrow \mathsf{Enroll}(E_r, \sigma_{r_{\mathcal{IP}}}, \mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}}, \mathcal{G}, \mathcal{H}_{\mathcal{U}}, v_1, V_2)$
$\mathcal{O} \leftarrow \{\mathsf{ComputeD}(\cdot,\cdot), \mathsf{ReEncryptD}(\cdot, \mathtt{sk}'_{\mathcal{U}}, \cdot, \mathcal{G}', \cdot), \mathsf{Decide}(\cdot, \delta)\}$
$E^*_{D_{\mathcal{SP}}} \leftarrow \mathcal{A}^{\mathcal{O}}$ $(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{pk}'_{\mathcal{U}}, \mathtt{pk}_{\mathcal{SP}}, E'_r, \sigma'_{r_{\mathcal{IP}}}, E_{k_{\mathcal{SP}}}, G_{z_1}, E'^*_{D_{\mathcal{U}}})$
letting T^i_e *denote the queries to the* ComputeD *oracle,*
T^*_e *was not used before*
and $E'^*_{D_{\mathcal{U}}}$ *is the randomized encrypted distance between* E'_r *and* T^*_e
If $\mathsf{Decide}(E^*_{D_{\mathcal{SP}}}, E^*_{D_{\mathcal{U}}}, \mathtt{sk}_{\mathcal{SP}}, \mathtt{pk}_{\mathcal{SP}}, \mathtt{pk}'_{\mathcal{U}}, \mathcal{G}", G_{z_2}, \gamma, z_1, \{\gamma_i\}_{i=1..n}, T^*_e, \delta) = 1$
return 1
Else return 0

4.4.3.3 Unlinkability

The unlinkability property ensures that two or several service providers are not able to collude and link several enrollment sessions belonging to the same user. Formally, this is defined in a game $\mathbf{Exp}^{unlink}_{\mathcal{A}}$ where an adversary $\mathcal{A}$, acting as two colluding curious service providers $\mathcal{SP}_0$ and $\mathcal{SP}_1$, has access to an Enroll oracle. The adversary may query this oracle for two users $\mathcal{U}_0$ and $\mathcal{U}_1$ having the tuples $(E_{r_{\mathcal{U}_0}}, \sigma_{r_{\mathcal{IP}_0}}, \mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{\mathcal{U}_0}, \mathcal{G}_{\mathcal{U}_0}, \mathcal{H}_{\mathcal{U}_0})$ and $(E_{r_{\mathcal{U}_1}}, \sigma_{r_{\mathcal{IP}_1}}, \mathtt{sk}_{\mathcal{U}_1}, \mathtt{pk}_{\mathcal{U}_1}, \mathcal{G}_{\mathcal{U}_1}, \mathcal{H}_{\mathcal{U}_1})$, respectively. A left-or-right oracle LoRErl is initialized with a secret random bit b and tuples $(E_{r_{\mathcal{U}_0}}, \sigma_{r_{\mathcal{IP}_0}}, \mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{\mathcal{U}_0}, \mathcal{G}_{\mathcal{U}_0}, \mathcal{H}_{\mathcal{U}_0})$ for user $\mathcal{U}_0$ and $(E_{r_{\mathcal{U}_1}}, \sigma_{r_{\mathcal{IP}_1}}, \mathtt{sk}_{\mathcal{U}_1}, \mathtt{pk}_{\mathcal{U}_1}, \mathcal{G}_{\mathcal{U}_1}, \mathcal{H}_{\mathcal{U}_1})$ for user $\mathcal{U}_1$. Note that session keys (i.e., v_{11}, V_{21}, v_{12} and V_{22}) are pre-computed by the adversary. The LoRErl oracle returns to $\mathcal{A}$ two modified credentials $(E'_{r_{\mathcal{U}_0}}, \sigma'_{r_{\mathcal{IP}_0}}, \mathtt{pk}'_{\mathcal{U}_0})$ and $(E'_{r_{\mathcal{U}_b}}, \sigma'_{r_{\mathcal{IP}_b}}, \mathtt{pk}'_{\mathcal{U}_b})$ for users $\mathcal{U}_0$ and $\mathcal{U}_b$, respectively. The adversary wins the game if he successfully predicts the bit b with a guessing probability greater than $\frac{1}{2}$.

Definition 11. Unlinkability – We say that PUBA satisfies the unlinkability property, if for every *PPT* adversary $\mathcal{A}$, there exists a negligible function κ such that:

$$Pr[\mathbf{Exp}_{\mathcal{A}}^{unlink}(1^{\lambda}) = 1] = \frac{1}{2} \pm \kappa(\lambda)$$

where $\mathbf{Exp}_{\mathcal{A}}^{unlink}$ is the security experiment against the unlinkability property, given hereafter.

$\boldsymbol{Exp}_{\mathcal{A}}^{unlink}(\lambda)$
$pp \leftarrow \mathsf{Set\_Params}(\lambda)$
$(\mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}}) \leftarrow \mathsf{IPKeyGen}()$
$(\mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{\mathcal{U}_0}) \leftarrow \mathsf{UKeyGen}()$
$(\mathtt{sk}_{\mathcal{U}_1}, \mathtt{pk}_{\mathcal{U}_1}) \leftarrow \mathsf{UKeyGen}()$
$(\mathtt{sk}_{\mathcal{SP}_0}, \mathtt{pk}_{\mathcal{SP}_0}) \leftarrow \mathsf{SPKeyGen}()$
$(\mathtt{sk}_{\mathcal{SP}_1}, \mathtt{pk}_{\mathcal{SP}_1}) \leftarrow \mathsf{SPKeyGen}()$
$(E_{r_{\mathcal{U}_i}}, (\rho_{1_{\mathcal{U}_i}}, ..., \rho_{n_{\mathcal{U}_i}})) \leftarrow \mathsf{Encrypt}(T_{r_{\mathcal{U}_i}}, \mathtt{pk}_{\mathcal{U}_i}),\ i \in \{0,1\}$
$(\sigma_{r_{\mathcal{IP}_i}}, \mathcal{G}_{\mathcal{U}_i}, \mathcal{H}_{\mathcal{U}_i}) \leftarrow \mathsf{Sign}(m, \mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{U}_i}, t_{r_{\mathcal{U}_i}}, (\rho_{1_{\mathcal{U}_i}}, ..., \rho_{n_{\mathcal{U}_i}})),\ i \in \{0,1\}$
$b \leftarrow \{0,1\}$
$\mathcal{O} \leftarrow \{\mathsf{Enroll}(\cdot, \cdot, \mathtt{sk}_{\mathcal{U}_i}, \mathcal{G}_{\mathcal{U}_i}, \mathcal{H}_{\mathcal{U}_i}, v_{1_{\mathcal{U}_i}}, \cdot), \mathsf{LoREnl}(\cdot,\cdot,b)\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}(pp, \mathtt{pk}_{\mathcal{IP}}, \mathtt{sk}_{\mathcal{SP}_0}, \mathtt{pk}_{\mathcal{SP}_0}, \mathtt{sk}_{\mathcal{SP}_1}, \mathtt{pk}_{\mathcal{SP}_1})$
If $b = b'$
  return 1
Else return 0

$\mathsf{LoREnl}((E_{r_{\mathcal{U}_0}}, \sigma_{r_{\mathcal{IP}_0}}, \mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{U_0}, \mathcal{G}_{\mathcal{U}_0}, \mathcal{H}_{\mathcal{U}_0}), (E_{r_{\mathcal{U}_1}}, \sigma_{r_{\mathcal{IP}_1}}, \mathtt{sk}_{\mathcal{U}_1}, \mathtt{pk}_{\mathcal{U}_1}, \mathcal{G}_{\mathcal{U}_1}, \mathcal{H}_{\mathcal{U}_1}), v_{11}, V_{21}, v_{12}, V_{22}, b)$

$(E'_{r_{\mathcal{U}_0}}, \sigma'_{r_{\mathcal{IP}_0}}, \mathtt{sk}'_{\mathcal{U}_0}, \mathtt{pk}'_{\mathcal{U}_0}, \mathcal{G}'_{\mathcal{U}_1}) \leftarrow \mathsf{Enroll}(E_{r_{\mathcal{U}_0}}, \sigma_{r_{\mathcal{IP}_0}}, \mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{U_0}, \mathcal{G}_{\mathcal{U}_0}, \mathcal{H}_{\mathcal{U}_0}, v_{11}, V_{21})$
$(E'_{r_{\mathcal{U}_b}}, \sigma'_{r_{\mathcal{IP}_b}}, \mathtt{sk}'_{\mathcal{U}_b}, \mathtt{pk}'_{\mathcal{U}_b}, \mathcal{G}'_{\mathcal{U}_b}) \leftarrow \mathsf{Enroll}(E_{r_{\mathcal{U}_b}}, \sigma_{r_{\mathcal{IP}_b}}, \mathtt{sk}_{\mathcal{U}_b}, \mathtt{pk}_{\mathcal{U}_b}, \mathcal{G}_{\mathcal{U}_b}, \mathcal{H}_{\mathcal{U}_b}, v_{12}, V_{22})$
return $((E'_{r_{\mathcal{U}_0}}, \sigma'_{r_{\mathcal{IP}_0}}, \mathtt{pk}'_{\mathcal{U}_0}), (E'_{r_{\mathcal{U}_b}}, \sigma'_{r_{\mathcal{IP}_b}}, \mathtt{pk}'_{\mathcal{U}_b}))$

4.4.3.4 Anonymity

The anonymity property states that colluding curious entities are able not to determine the real identity of a registered user. In other words, curious entities are not able to link modified credentials to their origin even if this latter is known. Formally, this is defined in a game $\mathbf{Exp}_{\mathcal{A}}^{anon}$, where an adversary $\mathcal{A}$ acting as colluding curious identity and service providers, has access to the Enroll oracle for two users $\mathcal{U}_0$ and $\mathcal{U}_1$ having the tuples $(E_{r_{\mathcal{U}_0}}, \sigma_{r_{\mathcal{IP}_0}}, \mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{U_0}, \mathcal{G}_{\mathcal{U}_0}, \mathcal{H}_{\mathcal{U}_0})$ and $(E_{r_{\mathcal{U}_1}}, \sigma_{r_{\mathcal{IP}_1}}, \mathtt{sk}_{\mathcal{U}_1}, \mathtt{pk}_{\mathcal{U}_1}, \mathcal{G}_{\mathcal{U}_1}, \mathcal{H}_{\mathcal{U}_1})$, respectively. $\mathcal{A}$ has also access to a left-or-right oracle LoRAN initialized with a secret random bit $b \in \{0,1\}$. $\mathcal{A}$ is given back modified credentials of the tuple $(E_{r_{\mathcal{U}_b}}, \sigma_{r_{\mathcal{IP}_b}}, \mathtt{sk}_{\mathcal{U}_b}, \mathtt{pk}_{\mathcal{U}_b}, \mathcal{G}_{\mathcal{U}_b}, \mathcal{H}_{\mathcal{U}_b})$ for user $\mathcal{U}_b$. The adversary wins the game if he successfully predicts the bit b, i.e., $\mathcal{A}$ links the randomized credentials to their origin with a probability greater than $\frac{1}{2}$.

Definition 12. Anonymity – We say that PUBA satisfies the anonymity property, if for every PPT adversary $\mathcal{A}$, there exists a negligible function κ such that:

$$Pr[\mathbf{Exp}_{\mathcal{A}}^{anon}(1^{\lambda}) = 1] = \frac{1}{2} \pm \kappa(\lambda)$$

where $\mathbf{Exp}_{\mathcal{A}}^{anon}$ is the security experiment against the anonymity property, given hereafter.

$\boldsymbol{Exp}_{\mathcal{A}}^{anon}(\lambda)$
$pp \leftarrow \mathsf{Set\_Params}(\lambda)$
$(\mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}}) \leftarrow \mathsf{IPKeyGen}()$
$(\mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{\mathcal{U}_0}) \leftarrow \mathsf{UKeyGen}()$
$(\mathtt{sk}_{\mathcal{U}_1}, \mathtt{pk}_{\mathcal{U}_1}) \leftarrow \mathsf{UKeyGen}()$
$(\mathtt{sk}_{\mathcal{SP}}, \mathtt{pk}_{\mathcal{SP}}) \leftarrow \mathsf{SPKeyGen}()$
$(E_{r_{\mathcal{U}_i}}, (\rho_{1_{\mathcal{U}_i}}, ..., \rho_{n_{\mathcal{U}_i}})) \leftarrow$
$\mathsf{Encrypt}(T_{r_{\mathcal{U}_i}}, \mathtt{pk}_{\mathcal{U}_i})$, $i \in \{0,1\}$
$(\sigma_{r_{\mathcal{IP}_i}}, \mathcal{G}_{\mathcal{U}_i}, \mathcal{H}_{\mathcal{U}_i}) \leftarrow \mathsf{Sign}(m, \mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{U}_i}, t_{r_{\mathcal{U}_i}}, (\rho_{1_{\mathcal{U}_i}}, ..., \rho_{n_{\mathcal{U}_i}}))$, $i \in \{0,1\}$
$b \leftarrow \{0,1\}$
$\mathcal{O} \leftarrow \{\mathsf{Enroll}(\cdot,\cdot,\mathtt{sk}_{\mathcal{U}_i},\cdot), \mathsf{LoRAN}(\cdot,\cdot,b)\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathtt{pk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{SP}}\ \mathtt{pk}_{\mathcal{U}_i}, T_{r_{\mathcal{U}_i}}, E_{r_{\mathcal{U}_i}}, \sigma_{r_{\mathcal{IP}_i}}, pp)$
If $b = b'$
  return 1
Else return 0

$\mathsf{LoRAN}((E_{r_{\mathcal{U}_0}}, \sigma_{r_{\mathcal{IP}_0}}, \mathtt{sk}_{\mathcal{U}_0}, \mathtt{pk}_{U_0}, \mathcal{G}_{\mathcal{U}_0}, \mathcal{H}_{\mathcal{U}_0}), (E_{r_{\mathcal{U}_1}}, \sigma_{r_{\mathcal{IP}_1}}, \mathtt{sk}_{\mathcal{U}_1}, \mathtt{pk}_{\mathcal{U}_1}, \mathcal{G}_{\mathcal{U}_1}, \mathcal{H}_{\mathcal{U}_1}), v_1, V_2, b)$

$(E'_{r_{\mathcal{U}_b}}, \sigma'_{r_{\mathcal{IP}_b}}, \mathtt{sk}'_{\mathcal{U}_b}, \mathtt{pk}'_{\mathcal{U}_b}, \mathcal{G}'_{\mathcal{U}_b}) \leftarrow \mathsf{Enroll}(E_{r_{\mathcal{U}_b}}, \sigma_{r_{\mathcal{IP}_b}}, \mathtt{sk}_{\mathcal{U}_b}, \mathtt{pk}_{\mathcal{U}_b}, \mathcal{G}_{\mathcal{U}_b}, \mathcal{H}_{\mathcal{U}_b}, v_1, V_2)$
return $(E'_{r_{\mathcal{U}_b}}, \sigma'_{r_{\mathcal{U}_b}}, \mathtt{pk}'_{\mathcal{U}_b})$

4.5 Building Blocks

The construction of PUBA relies on two main building blocks, namely the PS signature scheme and the El Gamal encryption scheme. As the PS signature scheme has been already presented in Chapter 3, Section 3.6, this section first introduces the El Gamal encryption scheme in Section 4.5.1. Second, it shows, in 4.5.2, how to use this encryption scheme to compute an encrypted Hamming distance.

4.5.1 El Gamal Encryption Scheme

El Gamal encryption is an asymmetric encryption scheme introduced by El Gamal in 1984 [55]. The scheme relies on Diffie-Hellman assumption and involves three main primitives referred to as KEY GENERATION, ENCRYPTION and DECRYPTION.
KEY GENERATION – Considering a cyclic group $\mathbb{G}$ of order q and a generator g of $\mathbb{G}$, the secret key is set as $sk = x$, where $x \in \mathbb{Z}^*_q$ and the public key is $pk = g^x$. Note that pk and sk refer to the encryption and decryption keys, respectively.
ENCRYPTION – Relying on the encryption key $\mathtt{pk}$, a message $m \in \mathbb{G}$ is encrypted as

follows: select $r \in \mathbb{Z}_q^*$ and compute $c_1 = g^r$ and $c_2 = m.\mathtt{pk}^r$. The ciphertext is then $C = E(m, r) = (c_1, c_2)$.
DECRYPTION – Relying on the decryption key $\mathtt{sk}$, the message is decrypted as follows: compute $\frac{c_2}{c_1^{\mathtt{sk}}} = m$.

The El Gamal cryptosystem supports several properties listed below :

- **Homomorphism:** $E(m_1, r_1) * E(m_2, r_2) = E(m_1 m_2,\ r_1 + r_2)$, where $m_1, m_2 \in \mathbb{G}$ and $r_1, r_2 \in \mathbb{Z}_q^*$.
- **Inverse:** $(E(m, r))^{-1} = E(m^{-1}, -r)$, where $m \in \mathbb{G}$ and $r \in \mathbb{Z}_q^*$
- **Juxtaposition:** $m_2 \cdot E(m_1, r) = E(m_1 m_2, r)$, where $m_1, m_2 \in \mathbb{G}$ and $r \in \mathbb{Z}_q^*$.

In the following, we denote by $\mathcal{E}_{\mathtt{pk}}(m) = E(m, r)$ the El Gamal encryption of the message m using the encryption key $\mathtt{pk}$. For ease of presentation, the random r, used for the encryption, is omitted.

4.5.2 Encrypted Hamming Distance

Given two templates $T_r = (r_1, ..., r_n)$ and $T_e = (e_1, ..., e_n)$ of length n, where $r_i, e_i \in \mathbb{Z}_q^*$ for $i \in \{1..n\}$. Each component of the two templates can be encrypted and expressed as follows:

$$\begin{cases} \mathcal{E}_{\mathtt{pk}}(g^{r_i}) = E(g^{r_i}, \alpha_i) = (g^{\alpha_i}, g^{r_i} \cdot \mathtt{pk}^{\alpha_i}), & \forall i \\ \mathcal{E}_{\mathtt{pk}}(g^{e_i}) = E(g^{e_i}, \beta_i) = (g^{\beta_i}, g^{e_i} \cdot \mathtt{pk}^{\beta_i}), & \forall i \end{cases}$$

where $\alpha_i, \beta_i \in \mathbb{Z}_q^*$ for $i \in \{1..n\}$.

Relying on the El Gamal cryptosystem, the encrypted Hamming distance $\mathcal{E}_{\mathtt{pk}}(D_H)$ between T_r and T_e can be computed as

$$\begin{aligned} \mathcal{E}_{\mathtt{pk}}(D_H) &= \textstyle\prod_{i=1}^n \mathcal{E}_{\mathtt{pk}}(g^{r_i})^{(1-2e_i)} * \mathcal{E}_{\mathtt{pk}}(g^{e_i}) \\ &= \textstyle\prod_{i=1}^n E(g^{r_i}, \alpha_i)^{(1-2e_i)} * E(g^{e_i}, \beta_i) \\ &= \textstyle\prod_{i=1}^n E(g^{r_i(1-2e_i)}, \alpha_i(1-2e_i)) * E(g^{e_i}, \beta_i) \\ &= \textstyle\prod_{i=1}^n E(g^{r_i(1-2e_i)} g^{e_i}, \alpha_i(1-2e_i) + \beta_i) \\ &= \textstyle\prod_{i=1}^n E(g^{r_i - 2r_i e_i + e_i}, \alpha_i(1-2e_i) + \beta_i) \\ &= E(g^{\sum_i (r_i - 2r_i e_i + e_i)}, \textstyle\sum_i (\alpha_i(1-2e_i) + \beta_i)) \\ &= E(D_H, \textstyle\sum_i (\alpha_i(1-2e_i) + \beta_i)) \\ &= (g^{\sum_i \alpha_i(1-2e_i) + \beta_i}, D_H \cdot \mathtt{pk}^{\sum_i \alpha_i(1-2e_i) + \beta_i}) \end{aligned}$$

4.6 PUBA Algorithms

This section presents the detailed construction of PUBA, including the twelve algorithms, as presented in Section 4.4.2. The concrete construction is based on the polymorphic

features of the El Gamal encryption scheme and the Pointcheval-Sanders signature scheme introduced in Section 4.5.

4.6.1 Setup Phase

This phase includes four algorithms referred to as $\mathsf{Set\_Params}_{\mathcal{TA}}$, $\mathsf{IPKeyGen}_{\mathcal{TA}}$, $\mathsf{UKeyGen}_{\mathcal{U}}$ and $\mathsf{SPKeyGen}_{\mathcal{SP}}$.

- $\mathsf{Set\_Params}_{\mathcal{TA}}$ – Relying on the security parameter λ, $\mathcal{TA}$ generates an asymmetric bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, g_1, g_2, e)$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are two cyclic groups of prime order q, g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and e is a bilinear map such that $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$. $\mathcal{TA}$ also sets a global public message $m = (m_1, ..., m_n) \in \mathbb{Z}_q^n$. Thus, let the tuple $(n, q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, g_1, g_2, e, m)$ denoted by pp be the output of the $\mathsf{Set\_Params}_{\mathcal{TA}}$ algorithm. It represents the system global public parameters that are known by all the system entities.

- $\mathsf{IPKeyGen}_{\mathcal{TA}}$ – $\mathcal{TA}$ generates the pair of secret and public keys $(\mathtt{sk}_{\mathcal{IP}}, \mathtt{pk}_{\mathcal{IP}})$ of $\mathcal{IP}$, relying on two selected randoms $x, y \in \mathbb{Z}_q^*$, as follows.

$$\mathtt{sk}_{\mathcal{IP}} = (x, y) \quad ; \quad \mathtt{pk}_{\mathcal{IP}} = (X, Y) = (g_2^x, g_2^y)$$

- $\mathsf{UKeyGen}_{\mathcal{U}}$ – the user generates his pair of secret and public keys $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$ that he shares with $\mathcal{TA}$. $\mathcal{U}$ selects a random $\alpha \in \mathbb{Z}_q^*$ and sets the pair of keys as follows.

$$\mathtt{sk}_{\mathcal{U}} = (\alpha, \mathtt{A}) = (\alpha, g_1^\alpha) \quad ; \quad \mathtt{pk}_{\mathcal{U}} = g_2^\alpha$$

  Note that $\mathcal{U}$ shares only a part from his secret key with the $\mathcal{IP}$, i.e., the component $\mathtt{A}$.

- $\mathsf{SPKeyGen}_{\mathcal{SP}}$ – the service provider sets up his pair of secret and public keys $(\mathtt{sk}_{\mathcal{SP}}, \mathtt{pk}_{\mathcal{SP}})$ to be shared with $\mathcal{TA}$. $\mathcal{SP}$ picks at random $\beta \in \mathbb{Z}_q^*$ and computes the pair of keys as follows.

$$\mathtt{sk}_{\mathcal{SP}} = \beta \quad ; \quad \mathtt{pk}_{\mathcal{SP}} = g_2^\beta$$

4.6.2 Identity_Issue Phase

The IDENTITY_ISSUE phase assumes that the user is physically present at the $\mathcal{IP}$'s desk. It starts by capturing a biometric identity of $\mathcal{U}$. $\mathcal{IP}$ extracts a template $t_r = (r_1, ..., r_n)$ and computes the associated user's reference template $T_r = (R_1, ..., R_n) = (g_2^{r_1}, ..., g_2^{r_n})$. Then, the IDENTITY_ISSUE phase, as depicted in Table 4.2, relies on the $\mathsf{Encrypt}_{\mathcal{IP}}$, $\mathsf{Sign}_{\mathcal{IP}}$ and $\mathsf{VerifySig}_{\mathcal{U}}$ algorithms to generate a certified identity for the user.

- $\mathsf{Encrypt}_{\mathcal{IP}}$ – The $\mathcal{IP}$ relies on the El Gamal cryptosystem to encrypt the reference template T_r with the user's public key $\mathtt{pk}_{\mathcal{U}}$. Indeed $\mathcal{IP}$ picks at random $\rho_1, ..., \rho_n \in$

Table 4.2: Description of the IDENTITY_ISSUE Phase

	Identity Provider		**User**
	knows $t_r = (r_1, ..., r_n)$, $T_r = (R_1, ..., R_n)$, $\mathtt{sk}_{\mathcal{IP}} = (x, y)$, $\mathtt{pk}_{\mathcal{IP}} = (X, Y)$, $\mathtt{A}$ and $\mathtt{pk}_{\mathcal{U}} = g_2^{\alpha}$		knows $\mathtt{sk}_{\mathcal{U}} = (\alpha, \mathtt{A})$, $\mathtt{pk}_{\mathcal{U}} = g_2^{\alpha}$ and $\mathtt{pk}_{\mathcal{IP}} = (X, Y)$
$\mathsf{Encrypt}_{\mathcal{IP}}$	pick at random $\rho_1, ..., \rho_n \in \mathbb{Z}_q^*$ compute $E_r = ((c_1^{(1)}, c_2^{(1)})..., (c_1^{(n)}, c_2^{(n)}))$ $= ((g_2^{\rho_1}, R_1 \mathtt{pk}_{\mathcal{U}}{}^{\rho_1}), ..., (g_2^{\rho_n}, R_n \mathtt{pk}_{\mathcal{U}}{}^{\rho_n}))$		
$\mathsf{Sign}_{\mathcal{IP}}$	pick at random $t_{\mathcal{U}} \in \mathbb{Z}_q^*$ compute $h_{\mathcal{U}} = g_1^{t_{\mathcal{U}}}$ set $a_u = h_{\mathcal{U}}$ compute $b_u = h_{\mathcal{U}}^{x + \sum_i \rho_i + (y + r_i) m_i} \cdot \prod_i \mathtt{A}^{t_{\mathcal{U}} \rho_i m_i}$ set $\sigma_{r_{\mathcal{IP}}} = (a_u, b_u)$ compute $\mathcal{G} = \{g_1^{\rho_i}\}_{i=1..n}$ and $\mathcal{H}_{\mathcal{U}} = \{h_{\mathcal{U}}{}^{\rho_i}\}_{i=1..n}$		
		$\xrightarrow{(E_r, \sigma_{r_{\mathcal{IP}}}, \mathcal{G}, \mathcal{H}_{\mathcal{U}})}$	
$\mathsf{VerifySig}_{\mathcal{U}}$			**If** $e(a_u, X \cdot \prod_{i=1}^n c_1^{(i)} (Y \cdot c_2^{(i)})^{m_i}) = e(b_u, g_2)$ **then** $b = 1$ **Else** $b = 0$

$\mathbb{Z}_q^*$ and computes E_r the encrypted reference template as follows.

$$\begin{aligned} E_r &= (\mathcal{E}_{\mathtt{pk}_{\mathcal{U}}}(R_1), ..., \mathcal{E}_{\mathtt{pk}_{\mathcal{U}}}(R_n)) \\ &= ((c_1^{(1)}, c_2^{(1)})..., (c_1^{(n)}, c_2^{(n)})) \\ &= ((g_2^{\rho_1}, R_1 \mathtt{pk}_{\mathcal{U}}{}^{\rho_1}), ..., (g_2^{\rho_n}, R_n \mathtt{pk}_{\mathcal{U}}{}^{\rho_n})) \end{aligned}$$

The $\mathsf{Encrypt}_{\mathcal{IP}}$ algorithm returns the encrypted template E_r and the tuple $(\rho_1, ..., \rho_n)$.

- $\mathsf{Sign}_{\mathcal{IP}}$ – The $\mathcal{IP}$ signs the message m w.r.t. the initial reference template t_r, the associated ciphertext E_r and the tuple $(\rho_1, ..., \rho_n)$ as shown in Algorithm 6. The tuple $(\sigma_{r_{\mathcal{IP}}}, \mathcal{G}, \mathcal{H}_{\mathcal{U}})$ is sent to $\mathcal{U}$.

Algorithm 6 $\mathsf{Sign}_{\mathcal{IP}}$ Algorithm

1: **Input:** the public message m, the $\mathcal{IP}$'s secret key $\mathtt{sk}_{\mathcal{IP}}$, the part $\mathtt{A}$ of the user's secret key, the template t_r and the tuple $(\rho_1, ..., \rho_n)$
2: **Output:** a signature $\sigma_{r_{\mathcal{IP}}}$ over the message m w.r.t. the reference template T_r and two lists of editing-keys $\mathcal{G}$ and $\mathcal{H}_{\mathcal{U}}$
3: pick at random $t_{\mathcal{U}} \in \mathbb{Z}_q^*$;
4: compute $h_{\mathcal{U}} = g_1^{t_{\mathcal{U}}}$;
5: set $a_u = h_{\mathcal{U}}$;
6: compute $b_u = h_{\mathcal{U}}^{x + \sum_i \rho_i + (y + r_i) m_i} \cdot \prod_i \mathtt{A}^{t_{\mathcal{U}} \rho_i m_i}$;
7: set $\sigma_{r_{\mathcal{IP}}} = (a_u, b_u)$;
8: compute $\mathcal{G} = \{g_1^{\rho_i}\}_{i=1..n}$ and $\mathcal{H}_{\mathcal{U}} = \{h_{\mathcal{U}}{}^{\rho_i}\}_{i=1..n}$;
9: **return** $(\sigma_{r_{\mathcal{IP}}}, \mathcal{G}, \mathcal{H}_{\mathcal{U}})$

- $\mathsf{VerifySig}_{\mathcal{U}}$ – The user checks the correctness of the signature $\sigma_{r_{\mathcal{IP}}}$ generated by the $\mathcal{IP}$ w.r.t. the encrypted template E_r and the $\mathcal{IP}$'s public key $\mathtt{pk}_{\mathcal{IP}}$. For this

purpose, $\mathcal{U}$ checks if the following equation holds.

$$e(a_u, X \cdot \prod_{i=1}^{n} c_1^{(i)} \cdot (Y \cdot c_2^{(i)})^{m_i}) = e(b_u, g_2) \tag{4.1}$$

Correctness of Equation 4.1

$$\begin{aligned} & e(a_u, X \cdot \textstyle\prod_{i=1}^{n} c_1^{(i)} \cdot (Y \cdot c_2^{(i)})^{m_i}) \\ = \; & e(h_{\mathcal{U}}, {g_2}^{x+\sum_i[\rho_i+(y+r_i)m_i+\alpha\rho_i m_i]}) \\ = \; & e(h_{\mathcal{U}}^{x+\sum_i[\rho_i+(y+r_i)m_i+\alpha\rho_i m_i]}, g_2) \\ = \; & e(b_u, g_2) \end{aligned}$$

At the end of the IDENTITY_ISSUE phase, $\mathcal{U}$ locally stores the tuple $(E_r, \sigma_{r_{\mathcal{IP}}}, \mathcal{G}, \mathcal{H}_{\mathcal{U}})$ in order to be used in the next phases.

4.6.3 Enrollment Phase

This phase is an interactive protocol between the user and a service provider (cf. Table 4.3). It involves two algorithms, namely $\mathsf{Enroll}_{\mathcal{U}}$ and $\mathsf{Verify}_{\mathcal{SP}}$. Indeed, when aiming at enrolling to $\mathcal{SP}$, $\mathcal{U}$ sends an enrollment request containing $h_{\mathcal{U}}^{v_1}$, where $v_1 \in \mathbb{Z}_q^*$. In turn, $\mathcal{SP}$ selects a random $v_2 \in \mathbb{Z}_q^*$ and gives back $h_{\mathcal{U}}^{v_1 v_2}$ to $\mathcal{U}$. The two algorithms $\mathsf{Enroll}_{\mathcal{U}}$ and $\mathsf{Verify}_{\mathcal{SP}}$ are then run as follows.

Table 4.3: Description of the ENROLLMENT Phase

	User		**Service Provider**
	knows $E_r = ((c_1^{(1)}, c_2^{(1)})..., (c_1^{(n)}, c_2^{(n)}))$, $\sigma_{r_{\mathcal{IP}}} = (a_u, b_u)$, $\mathcal{G}$, $\mathcal{H}_{\mathcal{U}}$, $\mathtt{sk}_{\mathcal{U}} = (\alpha, \mathtt{A})$, $\mathtt{pk}_{\mathcal{U}} = g_2^{\alpha}$ and $\mathtt{pk}_{\mathcal{SP}} = g_2^{\beta}$		knows $\mathtt{sk}_{\mathcal{SP}} = \beta$, $\mathtt{pk}_{\mathcal{SP}} = g_2^{\beta}$, and $\mathtt{pk}_{\mathcal{IP}} = (X, Y)$
	pick at random $v_1 \in \mathbb{Z}_q^*$ compute $h_{\mathcal{U}}^{v_1}$		
		$\xrightarrow{h_{\mathcal{U}}^{v_1}}$	
			pick at random $v_2 \in \mathbb{Z}_q^*$ compute $h_{\mathcal{U}}^{v_1 v_2}$
		$\xleftarrow{h_{\mathcal{U}}^{v_1 v_2}}$	
$\mathsf{Enroll}_{\mathcal{U}}$	pick at random $k \in \mathbb{Z}_q^*$ set $\mathtt{pk}'_{\mathcal{U}} = {\mathtt{pk}_{\mathcal{U}}}^k$ and $\mathtt{sk}'_{\mathcal{U}} = (\alpha', \mathtt{A}') = (k\alpha, \mathtt{A}^k)$ pick at random $\rho'_1, ..., \rho'_n \in \mathbb{Z}_q^*$ compute $E'_r = ((c_1^{'(1)}, c_2^{'(1)})..., (c_1^{'(n)}, c_2^{'(n)}))$ $= (((c_1^{(1)} \cdot g_2^{\rho'_1})^{\frac{1}{k}}, c_2^{(1)} \cdot \mathtt{pk}_{\mathcal{U}}{}^{\rho'_1})..., ((c_1^{(n)} \cdot g_2^{\rho'_n})^{\frac{1}{k}}, c_2^{(n)} \cdot \mathtt{pk}_{\mathcal{U}}{}^{\rho'_n}))$ pick at random $t \in \mathbb{Z}_q^*$ compute $a'_u = (h_{\mathcal{U}}^{v_1 v_2})^{\frac{t}{v_1}}$ compute $b'_u = \left(b_u \cdot {h_{\mathcal{U}}}^{\frac{1}{k}\sum_i \rho'_i} \cdot {h_{\mathcal{U}}}^{\alpha \sum_i \rho'_i m_i} \cdot \prod_i \mathcal{H}_{\mathcal{U}}[i]^{\frac{1}{k}-1}\right)^t$ set $\sigma'_{r_{\mathcal{IP}}} = (a'_u, b'_u)$ compute $\mathcal{G}' = \left\{\left(\mathcal{G}[i] \cdot {g_1}^{\rho'_i}\right)^{\frac{1}{k}}\right\}_{i=1..n}$		
		$\xrightarrow{(\mathtt{pk}'_{\mathcal{U}}, E'_r, \sigma'_{r_{\mathcal{IP}}})}$	
$\mathsf{Verify}_{\mathcal{SP}}$			**If** $e(a'_u, X \cdot \prod_{i=1}^{n} c_1^{'(i)} (Y \cdot c_2^{'(i)})^{m_i}) = e(b'_u, g_2)^{v_2}$ **then** $b = 1$ **Else** $b = 0$

- $\mathsf{Enroll}_{\mathcal{U}}$ – The user transforms the encrypted template E_r and accordingly the signature $\sigma_{r_{\mathcal{IP}}}$, the user's pair of keys $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$ and the list $\mathcal{G}$ of editing-keys,

as detailed in Algorithm 7. $\mathcal{SP}$ receives from $\mathcal{U}$ the tuple $(E'_r, \sigma'_{r_{\mathcal{IP}}}, \mathtt{pk}'_{\mathcal{U}})$. We assume that the randomized encrypted template E'_r is an El Gamal encryption of T_r with the randomized public key $\mathtt{pk}'_{\mathcal{U}}$, thus E'_r can be written as $E'_r = (\mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_1), ..., \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_n))$. $\mathcal{G}'$ and $\mathtt{sk}'_{\mathcal{U}}$ are kept secret and locally stored at $\mathcal{U}$.

Algorithm 7 $\mathsf{Enroll}_{\mathcal{U}}$ Algorithm

Input: the encrypted reference template E_r, the signature $\sigma_{r_{\mathcal{IP}}}$, the user's pair of keys $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$, the lists of editing-keys $\mathcal{G}$ and $\mathcal{H}_{\mathcal{U}}$ and the session keys v_1 and $h_{\mathcal{U}}^{v_1 v_2}$
Output: a transformed template E'_r, a signature $\sigma'_{r_{\mathcal{U}}}$, a randomized pair of encryption/decryption keys $(\mathtt{pk}'_{\mathcal{U}}, \mathtt{sk}'_{\mathcal{U}})$ and a randomized editing-key $\mathcal{G}'$
// **The next is executed to randomize the pair of encryption/decryption keys**
parse $\mathtt{sk}_{\mathcal{U}}$ as $(\alpha, \mathtt{A})$;
pick at random $k \in \mathbb{Z}_q^*$;
set $\mathtt{pk}'_{\mathcal{U}} = {\mathtt{pk}_{\mathcal{U}}}^k$ and $\mathtt{sk}'_{\mathcal{U}} = (\alpha', \mathtt{A}') = (k\alpha, \mathtt{A}^k)$;
// **The next is executed to transform encrypted reference template** E_r
parse E_r as $((c_1^{(1)}, c_2^{(1)})..., (c_1^{(n)}, c_2^{(n)}))$;
pick at random $\rho'_1, ..., \rho'_n \in \mathbb{Z}_q^*$;
compute $E'_r = ((c_1^{'(1)}, c_2^{'(1)})..., (c_1^{'(n)}, c_2^{'(n)})) = (((c_1^{(1)} \cdot {g_2^{\rho'_1}})^{\frac{1}{k}}, c_2^{(1)} \cdot {\mathtt{pk}_{\mathcal{U}}}^{\rho'_1})..., ((c_1^{(n)} \cdot g_2^{\rho'_n})^{\frac{1}{k}}, c_2^{(n)} \cdot {\mathtt{pk}_{\mathcal{U}}}^{\rho'_n}))$;
// **The next is executed to randomize the signature** $\sigma_{r_{\mathcal{IP}}}$
parse $\sigma_{r_{\mathcal{IP}}}$ as (a_u, b_u) ;
pick at random $t \in \mathbb{Z}_q^*$;
compute $a'_u = (h_{\mathcal{U}}^{v_1 v_2})^{\frac{t}{v_1}}$;
compute $b'_u = \left(b_u \cdot {h_{\mathcal{U}}}^{\frac{1}{k}\sum_i \rho'_i} \cdot {h_{\mathcal{U}}}^{\alpha \sum_i \rho'_i m_i} \cdot \prod_i \mathcal{H}_{\mathcal{U}}[i]^{\frac{1}{k}-1}\right)^t$;
set $\sigma'_{r_{\mathcal{IP}}} = (a'_u, b'_u)$;
// **The next is executed to randomize the list of editing-keys** $\mathcal{G}$
compute $\mathcal{G}' = \left\{\left(\mathcal{G}[i] \cdot {g_1}^{\rho'_i}\right)^{\frac{1}{k}}\right\}_{i=1..n}$;
return $(E'_r, \sigma'_{r_{\mathcal{IP}}}, \mathtt{sk}'_{\mathcal{U}}, \mathtt{pk}'_{\mathcal{U}}, \mathcal{G}')$

- $\mathsf{Verify}_{\mathcal{SP}}$ – The $\mathcal{SP}$ verifies the freshness of the enrollment session initialized by $\mathcal{U}$, and the validity of the signature $\sigma'_{r_{\mathcal{IP}}}$ w.r.t the template E'_r. The details of the $\mathsf{Verify}_{\mathcal{SP}}$ algorithm are depicted in Algorithm 8. If it returns 1, then $\mathcal{SP}$ stores the tuple $(\mathtt{pk}'_{\mathcal{U}}, E'_r, \sigma'_{r_{\mathcal{IP}}})$, otherwise, he rejects the enrollment request.

Correctness of equation in Algorithm 8

$$\begin{aligned} e(a'_u, X \cdot \prod_{i=1}^n c_1^{'(i)} (Y \cdot c_2^{'(i)})^{m_i}) &= e(h_{\mathcal{U}}^{v_2 t}, g_2^{x + \sum_i \frac{\rho_i + \rho'_i}{k}[(y+r_i)m_i + (\rho_i+\rho'_i)\alpha m_i]}) \\ &= e(h_{\mathcal{U}}^{t\left(x + \sum_i \frac{\rho_i+\rho'_i}{k}[(y+r_i)m_i+(\rho_i+\rho'_i)\alpha m_i]\right)}, g_2)^{v_2} \\ &= e(b'_u, g_2)^{v_2} \end{aligned}$$

Algorithm 8 $\mathsf{Verify}_{\mathcal{SP}}$ Algorithm

Input: the public message m, the template E'_r, the signature $\sigma'_{r_{\mathcal{IP}}}$, $\mathcal{IP}$'s public key $\mathtt{pk}_{\mathcal{IP}}$ and the session keys v_2 and $h_{\mathcal{U}}^{v_1 v_2}$
Output: a bit $b \in \{0,1\}$
parse $\sigma'_{r_{\mathcal{IP}}}$ as (a'_u, b'_u) and E'_r as $((c_1^{'(1)}, c_2^{'(1)})..., (c_1^{'(n)}, c_2^{'(n)}))$;
if $e(a'_u, X \cdot \prod_{i=1}^{n} c_1^{'(i)} (Y \cdot c_2^{'(i)})^{m_i}) = e(b'_u, g_2)^{v_2}$ **then**
    $b = 1$
else
    $b = 0$
end if
return b

4.6.4 Verification Phase

The Verification phase is an interactive protocol between a user and a service provider (cf. Table 4.4). This protocol is initialized by the reception of the user's randomized public key $\mathtt{pk}'_{\mathcal{U}}$ and the extraction of a fresh template $T_e = (e_1, ..., e_n) \in \mathbb{Z}_q^n$ on the $\mathcal{SP}$'s sensors. The Verification phase is characterized by the computation of an homomorphically encrypted Hamming distance between two templates based on El Gamal cryptosystem. This phase is run between $\mathcal{U}$ and $\mathcal{SP}$ through three algorithms, namely $\mathsf{ComputeD}_{\mathcal{SP}}$, $\mathsf{ReEncryptD}_{\mathcal{U}}$ and $\mathsf{Decide}_{\mathcal{SP}}$.

Table 4.4: Description of the Verification Phase

	User		**Service Provider**	
	knows $\mathtt{sk}'_{\mathcal{U}}$, $\mathtt{pk}'_{\mathcal{U}}$ and $\mathtt{pk}_{\mathcal{SP}}$		knows $\mathtt{sk}_{\mathcal{SP}}$, $\mathtt{pk}_{\mathcal{SP}}$, $\mathtt{pk}'_{\mathcal{U}}$, $E'_r = (\mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_1), ..., \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_n))$, $T_e = (e_1, ..., e_n)$ and δ	
		$\xrightarrow{\mathtt{pk}'_{\mathcal{U}}}$		
			Forall $i \in \{1, ..., n\}$ pick at random $\gamma_i \in \mathbb{Z}_q^*$ compute $\mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(g_2^{e_i}) = (g_2^{\gamma_i}, g_2^{e_i} \mathtt{pk}'^{\gamma_i}_{\mathcal{U}})$ compute $D_i = \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_i)^{(1-2e_i)} * \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(g_2^{e_i})$ compute $E_{D_{\mathcal{U}}} = \prod_{i=1}^{n} D_i = \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(D_H)$ pick at random $\mu \in \mathbb{G}_2^*$ compute $E'_{D_{\mathcal{U}}} = \mu \cdot E_{D\mathcal{U}} = \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(D_H \mu)$ pick at random $\gamma \in \mathbb{Z}_q^*$ compute $E_{k_{\mathcal{SP}}} = \mathcal{E}_{\mathtt{pk}_{\mathcal{SP}}}(\mu) = (g_2^{\gamma}, \mathtt{pk}_{\mathcal{SP}}{}^{\gamma} \mu)$ pick at random $z_1 \in \mathbb{Z}_q^*$ compute $G_{z_1} = g_1^{z_1}$	$\mathsf{ComputeD}_{\mathcal{SP}}$
		$\xleftarrow{(E'_{D_{\mathcal{U}}}, E_{k_{\mathcal{SP}}}, G_{z_1})}$		
$\mathsf{ReEncryptD}_{\mathcal{U}}$	parse $E'_{D_{\mathcal{U}}}$ as (c_{d1}, c_{d2}) compute $D'_H = \frac{c_{d2}}{(c_{d1})^{\mathtt{sk}'_{\mathcal{U}}}} = D_H \mu$ compute $E_{D_{\mathcal{SP}}} = D'_H \cdot (E_{k_{\mathcal{SP}}})^{-1} = \mathcal{E}_{\mathtt{pk}_{\mathcal{SP}}}(D_H)$ pick at random $z_2 \in \mathbb{Z}_q^*$ compute $\mathcal{G}'' = \{\mathcal{G}'[i]^{z_2}\}_{i=1..n}$ compute $G_{z_2} = G_{z_1}{}^{z_2}$			
		$\xrightarrow{(E_{D_{\mathcal{SP}}}, \mathcal{G}'', G_{z_2})}$		
			parse $E_{D_{\mathcal{SP}}}$ as $(\phi_1, \theta_1) = (g_2{}^{-\gamma}, D_H \ \mathtt{pk}_{\mathcal{SP}}^{-\gamma})$ and $E_{D_{\mathcal{U}}}$ as $(\phi_2, \theta_2) = (g_2{}^{\sum_i \frac{\rho_i + \rho'_i}{k}(1-2e_i)+\gamma_i}, D_H \ \mathtt{pk}'^{\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i)+\gamma_i}_{\mathcal{U}})$ compute $\zeta_1 = \mathtt{pk}'^{-\gamma}_{\mathcal{U}}$ and $\zeta_2 = \phi_2{}^{\mathtt{sk}_{\mathcal{SP}}} = \mathtt{pk}_{\mathcal{SP}}{}^{\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i)+\gamma_i}$ **If** $e(G_{z_2}, \frac{\theta_1}{\theta_2} \cdot \frac{\zeta_2}{\zeta_1}) = e(G_{z_2}{}^{-\gamma + \sum_i \gamma_i} \cdot \prod_i \mathcal{G}''[i]^{(1-2e_i)z_1}, \mathtt{pk}_{\mathcal{SP}} / \mathtt{pk}'_{\mathcal{U}})$ **then** compute $D_H = \frac{\theta_1}{(\phi_1)^{\mathtt{sk}_{\mathcal{SP}}}}$ **Else** Exit **If** $D_H < \delta$ **then** $b = 1$ **Else** $b = 0$	$\mathsf{Decide}_{\mathcal{SP}}$

- $\mathsf{ComputeD}_{\mathcal{SP}}$ – $\mathcal{SP}$ homomorphically computes an encrypted Hamming distance $E_{D\mathcal{U}}$ between the two templates T_r and T_e as depicted in Algorithm 9. $E_{D\mathcal{U}}$ is the

El Gamal encryption of the Hamming distance D_H with the user's randomized public key $\mathtt{pk}'_{\mathcal{U}}$. $E_{D_{\mathcal{U}}}$ and the elements $(\mu, \gamma, z_1, \{\gamma_i\}_{i=1..n})$ are kept secret at the $\mathcal{SP}$'s side. $\mathcal{SP}$ shares only the randomized distance $E'_{D_{\mathcal{U}}}$, the encrypted session key $E_{k_{\mathcal{SP}}}$ and the element G_{z_1} with $\mathcal{U}$.

Algorithm 9 $\mathsf{ComputeD}_{\mathcal{SP}}$ Algorithm

Input: the randomized encrypted template E'_r, the extracted template T_e, the user's randomized public key $\mathtt{pk}'_{\mathcal{U}}$ and the $\mathcal{SP}$'s public key $\mathtt{pk}_{\mathcal{SP}}$.
Output: an encrypted Hamming distance $E_{D_{\mathcal{U}}}$, a randomized distance $E'_{D_{\mathcal{U}}}$, an encrypted session key $E_{k_{\mathcal{SP}}}$ and the elements $(\gamma, z_1, G_{z_1}, \{\gamma_i\}_{i=1..n})$
parse E'_r as $(\mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_1), ..., \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_n))$ and T_e as $(e_1, ..., e_n)$;
for all $i \in \{1, \cdots, n\}$ **do**
    $\gamma_i \leftarrow \mathbb{Z}_q^*$;
    $\mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}({g_2}^{e_i}) = ({g_2}^{\gamma_i}, {g_2}^{e_i}{\mathtt{pk}'_{\mathcal{U}}}^{\gamma_i})$;
    $D_i = \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(R_i)^{(1-2e_i)} * \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}({g_2}^{e_i})$; // $*$ denotes the homomorphism property (cf. Section 4.5)
end for
compute $E_{D_{\mathcal{U}}} = \prod_{i=1}^{n} D_i = \mathcal{E}_{\mathtt{pk}'_{\mathcal{U}}}(D_H)$; // $\prod_{i=1}^{n} D_i = D_1 * D_2 * ... * D_n$
pick at random $\mu \in \mathbb{G}_2^*$;
compute $E'_{D_{\mathcal{U}}} = \mu \cdot E_{D_{\mathcal{U}}}$; // $\cdot$ denotes the juxtaposition property (cf. Section 4.5)
pick at random $\gamma \in \mathbb{Z}_q^*$ and compute $E_{k_{\mathcal{SP}}}$=$\mathcal{E}_{\mathtt{pk}_{\mathcal{SP}}}(\mu)$=$({g_2}^{\gamma}, {\mathtt{pk}_{\mathcal{SP}}}^{\gamma}\mu)$;
pick at random $z_1 \in \mathbb{Z}_q^*$;
compute $G_{z_1} = g_1^{z_1}$;
return $(E_{D_{\mathcal{U}}}, E'_{D_{\mathcal{U}}}, E_{k_{\mathcal{SP}}}, \gamma, z_1, G_{z_1}, \{\gamma_i\}_{i=1..n})$

- $\mathsf{ReEncryptD}_{\mathcal{U}}$ – The user decrypts the randomized distance $E'_{D_{\mathcal{U}}}$ and blindly re-encrypts the Hamming distance D_H with the $\mathcal{SP}$'s public key as shown in Algorithm 10. $\mathcal{U}$ also computes the randomized list of editing-keys $\mathcal{G}$" and a verifying key G_{z_2} that he sends with the re-encrypted distance $E_{D_{\mathcal{SP}}}$ to $\mathcal{SP}$.

Algorithm 10 $\mathsf{ReEncryptD}_{\mathcal{U}}$ Algorithm

Input: the randomized distance $E'_{D_{\mathcal{U}}}$, the user's randomized secret key $\mathtt{sk}'_{\mathcal{U}}$, the encrypted session key $E_{k_{\mathcal{SP}}}$, the list of editing-keys $\mathcal{G}'$ and the element G_{z_1}.
Output: an encrypted Hamming distance $E_{D_{\mathcal{SP}}}$ and a randomized list of editing-keys $\mathcal{G}$" and a verifying key G_{z_2}
parse $E'_{D_{\mathcal{U}}}$ as (c_{d1}, c_{d2})
compute $D'_H = \frac{c_{d2}}{(c_{d1})^{\mathtt{sk}'_{\mathcal{U}}}} = D_H\mu$;
compute $E_{D_{\mathcal{SP}}} = D'_H \cdot (E_{k_{\mathcal{SP}}})^{-1} = \mathcal{E}_{\mathtt{pk}_{\mathcal{SP}}}(D_H)$; // inverse and juxtaposition properties (cf. Section 4.5)
pick at random $z_2 \in \mathbb{Z}_q^*$;
compute $\mathcal{G}" = \{\mathcal{G}'[i]^{z_2}\}_{i=1..n}$;
compute $G_{z_2} = {G_{z_1}}^{z_2}$;
return $(E_{D_{\mathcal{SP}}}, \mathcal{G}", G_{z_2})$

- $\mathsf{Decide}_{\mathcal{SP}}$ – $\mathcal{SP}$ verifies that the Hamming distance has been correctly re-encrypted by $\mathcal{U}$. If so, he decrypts the distance and takes a decision whether to authenticate the user or not as illustrated in Algorithm 11.

Algorithm 11 $\mathsf{Decide}_{\mathcal{SP}}$ Algorithm

Input: the two encrypted distances $E_{D_{\mathcal{SP}}}$ and $E_{D_{\mathcal{U}}}$, $\mathcal{SP}$'s pair of keys ($\mathtt{sk}_{\mathcal{SP}}$,$\mathtt{pk}_{\mathcal{SP}}$), the user's randomized encryption key $\mathtt{pk}'_{\mathcal{U}}$, the list of editing-keys $\mathcal{G}$", the verifying key G_{z_2}, the elements $(\gamma, z_1, \{\gamma_i\}_{i=1..n})$, the extracted template T_e and a threshold δ
Output: a bit $b \in \{0,1\}$
parse $E_{D_{\mathcal{SP}}}$ as (ϕ_1,θ_1)=$({g_2}^{-\gamma}, D_H\ \mathtt{pk}_{\mathcal{SP}}^{-\gamma})$ and $E_{D_{\mathcal{U}}}$ as (ϕ_2,θ_2)=$({g_2}^{\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i)+\gamma_i}, D_H\ {\mathtt{pk}'_{\mathcal{U}}}^{\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i)+\gamma_i})$;
compute $\zeta_1 = {\mathtt{pk}'_{\mathcal{U}}}^{-\gamma}$ and $\zeta_2 = {\phi_2}^{\mathtt{sk}_{\mathcal{SP}}} = {\mathtt{pk}_{\mathcal{SP}}}^{\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i)+\gamma_i}$;
if $e(G_{z_2}, \frac{\theta_1}{\theta_2} \cdot \frac{\zeta_2}{\zeta_1}) = e({G_{z_2}}^{-\gamma+\sum_i \gamma_i} \cdot \prod_i \mathcal{G}"[i]^{(1-2e_i)z_1}, \mathtt{pk}_{\mathcal{SP}}/\mathtt{pk}'_{\mathcal{U}})$ **then**
    compute $D_H = \frac{\theta_1}{(\phi_1)^{\mathtt{sk}_{\mathcal{SP}}}}$
else
    Exit;
end if
if $D_H < \delta$ **then**
    b = 1
else
    b = 0
end if
return b

Correctness of equation in Algorithm 11

$$
\begin{aligned}
& e(G_{z_2}, \tfrac{\theta_1}{\theta_2} \cdot \tfrac{\zeta_2}{\zeta_1}) \\
= \ & e(g_1^{z_1 z_2}, \frac{D_H\ \mathtt{pk}_{\mathcal{SP}}^{-\gamma}}{D_H\ {\mathtt{pk}'_{\mathcal{U}}}^{\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i)+\gamma_i}} \cdot \frac{{\mathtt{pk}_{\mathcal{SP}}}^{\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i)+\gamma_i}}{{\mathtt{pk}'_{\mathcal{U}}}^{-\gamma}}) \\
= \ & e(g_1^{z_1 z_2(-\gamma+\sum_i \gamma_i)} \cdot g_1^{z_1 z_2(\sum_i \frac{\rho_i+\rho'_i}{k}(1-2e_i))}, \tfrac{\mathtt{pk}_{\mathcal{SP}}}{\mathtt{pk}'_{\mathcal{U}}}) \\
= \ & e({G_{z_2}}^{-\gamma+\sum_i \gamma_i} \cdot \textstyle\prod_i \mathcal{G}"[i]^{(1-2e_i)z_1}, \tfrac{\mathtt{pk}_{\mathcal{SP}}}{\mathtt{pk}'_{\mathcal{U}}})
\end{aligned}
$$

4.7 Security Analysis

This section shows that PUBA satisfies security and privacy requirements w.r.t. the threat model defined in Section 4.4.3.

For ease of presentation, we suppose, for all proofs, that the length of a template is set to $n = 1$.

4.7.1 Unforgeability

Theorem 4 (Unforgeability)**.** If a probabilistic-polynomial time (PPT) adversary $\mathcal{A}$ wins the $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ as defined in Section 4.4.3 with a non-negligible advantage ϵ, then

a PPT simulator $\mathcal{B}$ can be constructed to break the Existential Unforgeability under Chosen Message Attacks (EUF-CMA) of the Pointcheval-Sanders signature scheme, with non-negligible advantage ϵ.

Proof. Let us consider an adversary $\mathcal{A}$ that is allowed to query, as many times as he wants, the Encrypt and Sign oracles over a reference template $T_r^{(i)}$. $\mathcal{A}$ is then asked to produce a valid encrypted template/signature pair over a new reference template T_r^*. In this proof, we show that a simulator $\mathcal{B}$ can be built with the help of an adversary $\mathcal{A}$ having advantage ϵ against PUBA. The message m is a one-block message as $n = 1$.

The EUF-CMA challenger $\mathcal{C}$ of the Pointcheval-Sanders signature scheme sends to $\mathcal{B}$ the tuple $(g_2, X = g_2^x, Y = g_2^y)$, where $x, y \in \mathbb{Z}_q^*$ are the signer's secret keys and (X, Y) are the corresponding public keys. $\mathcal{C}$ asks $\mathcal{B}$ to produce a valid couple (h, h^{x+ym}) over a message m and a random $h \in \mathbb{G}_1^*$ of his choice, where m has not been used before and $h \neq 1_{\mathbb{G}_1}$. Then, $\mathcal{B}$ generates the public parameters of PUBA, including a one-block message m. $\mathcal{B}$ also considers that (X, Y) are the identity provider public key $\mathtt{pk}_{\mathcal{IP}}$. It randomly selects $r^* \in \mathbb{Z}_q^*$ and sends the biometric template $T_r^* = g^{r^*}$ to $\mathcal{A}$, along with the public key $\mathtt{pk}_{\mathcal{IP}}$ and the public parameters of the system. $\mathcal{A}$ has access to the user's pair of keys $\mathtt{sk}_{\mathcal{U}} = (\alpha, g_1^\alpha)$ and $\mathtt{pk}_{\mathcal{U}} = g_2^\alpha$.

During the challenge phase, $\mathcal{A}$ selects $\rho^*, t^* \in \mathbb{Z}_q^*$ and sets $h_{\mathcal{U}} = g_1^{t^*}$.
$\mathcal{A}$ forges PUBA with advantage ϵ. That is, it generates the encrypted template E_r^* and the signature σ_r^* over the message m, w.r.t. the template T_r^*: $E_r^* = (g_2^{\rho^*}, g_2^{\alpha\rho^{*}+r^*})$ and $\sigma_r^* = (a_u^*, b_u^*) = (h_{\mathcal{U}}, h_{\mathcal{U}}^{x+\rho^*+(y+r^*+\alpha\rho^*)m})$. $\mathcal{A}$ also computes $\mathcal{G}^* = g_1^{\rho^*}$, $\mathcal{H}_{\mathcal{U}}^* = h_{\mathcal{U}}^{\rho^*}$ w.r.t. the Sign algorithm. To help $\mathcal{B}$ to succeed the challenge against the EUF-CMA of the PS signature scheme, $\mathcal{A}$ computes an extra element $h_{\mathcal{U}}^{\alpha\rho^*}$. The tuple $(m, E_r^*, \sigma_r^*, \mathcal{G}^*, \mathcal{H}_{\mathcal{U}}^*, h_{\mathcal{U}}^{\alpha\rho^*})$ is sent back to $\mathcal{B}$.
Knowing the value of r^*, $\mathcal{B}$ can compute the couple $(h_{\mathcal{U}}, h_{\mathcal{U}}^{x+ym})$. It then sends the result to $\mathcal{C}$. As such, $\mathcal{B}$ succeeds the forgery against the EUF-CMA of the Pointcheval-Sanders signature scheme with advantage ϵ.
According to [113], the advantage ϵ of forging the EUF-CMA of the Pointcheval-Sanders signature scheme is negligible, thus, PUBA satisfies the unforgeability requirement. □

4.7.2 Soundness

Theorem 5 (Soundness)**.** PUBA satisfies the soundness requirement, with respect to the security of the El Gamal cryptosystem.

Proof. Let us consider an adversary $\mathcal{A}$ that is allowed to query, as many times as he wants, the ComputeD, ReEncryptD and Decide oracles over an extracted template T_e^i. We suppose that the tuple of randoms and verifying keys $(\mu, \gamma, z_1, G_{z_1}, \gamma_1, \mathcal{G}", G_{z_2})$ is set for the whole game. Then, during the challenge phase, $\mathcal{A}$ is asked to produce a valid

encrypted distance $E^*_{D_{\mathcal{SP}}}$ w.r.t. a new extracted template $T^*_e = g_2^{e^*}$ that was not queried before, and the associated randomized encrypted distance $E'^*_{D_{\mathcal{U}}} = (g_2^{\frac{\rho+\rho'}{k}(1-2e^*)+\gamma_1}, \mu D^*_H \cdot \mathtt{pk}'^{\frac{\rho+\rho'}{k}(1-2e^*)+\gamma_1}_{\mathcal{U}})$. That is, $\mathcal{A}$ needs to compute $E^*_{D_{\mathcal{SP}}} = (g_2^{-\gamma}, D^*_H \cdot \mathtt{pk}^{-\gamma}_{\mathcal{SP}})$, where $D^*_H = g_2^{r+e^*-2re^*}$.

For this purpose, $\mathcal{A}$ should correctly decrypt $E'^*_{D_{\mathcal{U}}}$ without having access to the user' secret key. Let us consider an adversary $\mathcal{B}$ against the security of the El Gamal cryptosystem[4]. Thus, the advantage of $\mathcal{A}$ to break the soundness of PUBA is expressed as follows:

$$\begin{aligned}\mathbf{Adv}^{sound}_{\mathcal{A}}(1^\lambda) &= Pr[\mathbf{Exp}^{sound}_{\mathcal{A}}(1^\lambda) = 1] \\ &= \mathbf{Adv}^{sec}_{\mathsf{ElGamal},\mathcal{B}}(1^\lambda)\end{aligned}$$

According to [139], the $\mathbf{Adv}^{sec}_{\mathsf{ElGamal},\mathcal{B}}(1^\lambda)$ function is negligible, proving that PUBA satisfies the soundness property.

□

4.7.3 Unlinkability

Theorem 6 (Unlinkability)**.** PUBA satisfies the unlinkability requirement, with respect to $\mathbf{Exp}^{unlink}_{\mathcal{A}}$ experiment.

Proof. We suppose, in this proof, that the same session key v_2 is used.
First, $\mathcal{A}$ is allowed to query, as many times as he wants, the Enroll oracle for two users $\mathcal{U}_0$ and $\mathcal{U}_1$. Thus, for each session i, the adversary $\mathcal{A}$ has access to the following tuples:
For user $\mathcal{U}_0$ owning a template $T_{r_{\mathcal{U}_0}} = g_2^{r_0}$

$$\begin{cases} E'^{(i)}_{r_{\mathcal{U}_0}} = (g_2^{E_0^{(i)}}, g_2^{F_0^{(i)}}) \\ \sigma'^{(i)}_{r_{\mathcal{IP}_0}} = (h_{\mathcal{U}_0}^{t_0^{(i)} v_2}, h_{\mathcal{U}_0}^{t_0^{(i)}(x+E_0^{(i)}+(y+F_0^{(i)})m)}) \\ \mathtt{pk}'^{(i)}_{\mathcal{U}_0} = g_2^{k_0^{(i)}\alpha_0} \end{cases}$$

where $E_0^{(i)} = \frac{\rho_0+\rho_0'^{(i)}}{k_0^{(i)}}$, $F_0^{(i)} = r_0 + \alpha_0(\rho_0 + \rho_0'^{(i)})$ and $h_{\mathcal{U}_0} = g_1^{t_{\mathcal{U}_0}}$.
For user $\mathcal{U}_1$ owning a template $T_{r_{\mathcal{U}_1}} = g_2^{r_1}$

$$\begin{cases} E'^{(i)}_{r_{\mathcal{U}_1}} = (g_2^{E_1^{(i)}}, g_2^{F_1^{(i)}}) \\ \sigma'^{(i)}_{r_{\mathcal{IP}_1}} = (h_{\mathcal{U}_1}^{t_1^{(i)} v_2}, h_{\mathcal{U}_1}^{t_1^{(i)}(x+E_1^{(i)}+(y+F_1^{(i)})m)}) \\ \mathtt{pk}'^{(i)}_{\mathcal{U}_1} = g_2^{k_1^{(i)}\alpha_1} \end{cases}$$

[4]The security of El Gamal refers to the impossibility to decrypt a ciphertext unless the appropriate secret keys are known.

where $E_1^{(i)} = \frac{\rho_1 + \rho_1^{'(i)}}{k_1^{(i)}}$, $F_1^{(i)} = r_1 + \alpha_1(\rho_1 + \rho_1^{'(i)})$ and $h_{\mathcal{U}_1} = g_1^{t_{\mathcal{U}_1}}$.

On the one hand, all the components of the reference template are El Gamal ciphertexts. According to [139], the El Gamal cryptosystem is proven to have indistinguishable encryptions under chosen plaintext attacks. Thus, $\mathcal{A}$ is not able to link different ciphertexts issued over the same plaintext biometric template, i.e., even randomized versions of the same ciphertext cannot be linked to each other.

On the other hand, we suppose that $\mathcal{A}$ is extremely strong, such that he is able to know the exponents of the signature's elements. Then, $\mathcal{A}$ has access to the following linear system of $4i$ equations and $6i + 10$ variables.

$$A_{0i} = t_{\mathcal{U}_0} t_0^{(i)} v_2, \forall i \tag{4.2}$$
$$B_{0i} = t_{\mathcal{U}_0} t_0^{(i)} (x + E_0^{(i)} + (y + F_0^{(i)})m), \forall i \tag{4.3}$$
$$A_{1i} = t_{\mathcal{U}_1} t_1^{(i)} v_2, \forall i \tag{4.4}$$
$$B_{1i} = t_{\mathcal{U}_1} t_1^{(i)} (x + E_1^{(i)} + (y + F_1^{(i)})m), \forall i \tag{4.5}$$

$\mathcal{A}$ knows the value of v_2, so the linear system can be simplified below to obtain the following system.

$$A'_{0i} = A_{0i}/v_2 = t_{\mathcal{U}_0} t_0^{(i)}, \forall i \tag{4.6}$$
$$B_{0i} = A'_{0i}(x + E_0^{(i)} + (y + F_0^{(i)})m), \forall i \tag{4.7}$$
$$A'_{1i} = A_{1i}/v_2 = t_{\mathcal{U}_1} t_1^{(i)}, \forall i \tag{4.8}$$
$$B_{1i} = A'_{1i}(x + E_1^{(i)} + (y + F_1^{(i)})m), \forall i \tag{4.9}$$

Afterwards, $\mathcal{A}$ has access to a left-or-right oracle LoREr that returns modified credentials for users $\mathcal{U}_0$ and $\mathcal{U}_b$, where $b \in \{0, 1\}$. Thus, $\mathcal{A}$ receives two signatures that can be respectively parsed as follows as $\sigma'_{r_{\mathcal{IP}_0}} = (h_{\mathcal{U}_0}^{t_0 v_2}, h_{\mathcal{U}_0}^{t_0(x+E_0+(y+F_0)m)})$ and $\sigma'_{r_{\mathcal{IP}_b}} = (h_{\mathcal{U}_b} t_b v_2, h_{\mathcal{U}_b}^{t_b(x+E_b+(y+F_b)m)})$, where $E_0 = \frac{\rho_0 + \rho_0^{'}}{k_0}$, $F_0 = r_0 + \alpha_0(\rho_0 + \rho_0^{'})$, $E_b = \frac{\rho_b + \rho_b^{'}}{k_b}$ and $F_b = r_b + \alpha_b(\rho_b + \rho_b^{'})$.

To break the unlinkability of Puba, $\mathcal{A}$ should compare the values of either α_0 and α_b, r_0 and r_b, or $t_{\mathcal{U}_0}$ and $t_{\mathcal{U}_b}$. Being a strong adversary, $\mathcal{A}$ establishes the following linear system from the exponents of signatures $\sigma'_{r_{\mathcal{IP}_0}}$ and $\sigma'_{r_{\mathcal{IP}_b}}$. Note that a simplification is made on the system since the value of v_2 is known by $\mathcal{A}$.

$$A_0 = t_{\mathcal{U}_0} t_0 \tag{4.10}$$
$$B_0 = x + E_0 + (y + F_0)m \tag{4.11}$$
$$A_1 = t_{\mathcal{U}_b} t_b \tag{4.12}$$
$$B_1 = x + E_1 + (y + F_1)m \tag{4.13}$$

Combining the two linear systems, $\mathcal{A}$ tries to solve a linear system with $4i + 4$ equations and $6i + 20$ unknown variables, i.e., x, y, α_0, α_1, α_b, r_0, r_1, r_b, ρ_0, ρ_1, ρ_b, ρ'_0, ρ'_b, k_0, k_b, $t_{\mathcal{U}_0}$, $t_{\mathcal{U}_1}$, $t_{\mathcal{U}_b}$, t_0, t_b, $t_0^{(i)}$, $t_1^{(i)}$, $\rho_0^{'(i)}$, $\rho_1^{'(i)}$, $k_0^{(i)}$ and $k_1^{(i)}$ $\forall i$, which is unfeasible. Hence, PUBA satisfies the unlinkability property. □

4.7.4 Anonymity

Theorem 7 (Anonymity)**.** PUBA satisfies the anonymity requirement, with respect to $\mathbf{Exp}_{\mathcal{A}}^{anon}$ experiment.

Proof. For this proof, we also consider that the same session key v_2 is used.
Let us consider an adversary $\mathcal{A}$ that is allowed to query, as many times as he wants, the Enroll oracle for two users $\mathcal{U}_0$ and $\mathcal{U}_1$ owning the templates $T_{r_{\mathcal{U}_0}} = g_2^{r_0}$ and $T_{r_{\mathcal{U}_1}} = g_2^{r_1}$, respectively. Thus, for each session i, $\mathcal{A}$ has access to the tuples below.

$$\begin{cases} E'^{(i)}_{r_{\mathcal{U}_0}} = (g_2^{E_0^{(i)}}, g_2^{F_0^{(i)}}) \\ \sigma'^{(i)}_{r_{\mathcal{IP}_0}} = (h_{\mathcal{U}_0}^{t_0^{(i)} v_2}, h_{\mathcal{U}_0}^{t_0^{(i)}(x + E_0^{(i)} + (y + F_0^{(i)})m)}) \\ \mathtt{pk}'^{(i)}_{\mathcal{U}_0} = g_2^{k_0^{(i)} \alpha_0} \end{cases}$$

$$\begin{cases} E'^{(i)}_{r_{\mathcal{U}_1}} = (g_2^{E_1^{(i)}}, g_2^{F_1^{(i)}}) \\ \sigma'^{(i)}_{r_{\mathcal{IP}_1}} = (h_{\mathcal{U}_1}^{t_1^{(i)} v_2}, h_{\mathcal{U}_1}^{t_1^{(i)}(x + E_1^{(i)} + (y + F_1^{(i)})m)}) \\ \mathtt{pk}'^{(i)}_{\mathcal{U}_1} = g_2^{k_1^{(i)} \alpha_1} \end{cases}$$

where $E_0^{(i)} = \frac{\rho_0 + \rho_0^{'(i)}}{k_0^{(i)}}$, $F_0^{(i)} = r_0 + \alpha_0(\rho_0 + \rho_0^{'(i)})$, $h_{\mathcal{U}_0} = g_1^{t_{\mathcal{U}_0}}$, $E_1^{(i)} = \frac{\rho_1 + \rho_1^{'(i)}}{k_1^{(i)}}$, $F_1^{(i)} = r_1 + \alpha_1(\rho_1 + \rho_1^{'(i)})$ and $h_{\mathcal{U}_1} = g_1^{t_{\mathcal{U}_1}}$.

We consider $\mathcal{A}$ as an extremely strong adversary that is able to know the exponents of the encrypted templates and the signatures' elements. Then, $\mathcal{A}$ has access to the following linear system.

$$E_0^{(i)} = \frac{\rho_0 + \rho_0^{'(i)}}{k_0^{(i)}}, \forall i \tag{4.14}$$

$$F_0^{(i)} = r_0 + \alpha_0(\rho_0 + \rho_0^{'(i)}), \forall i \tag{4.15}$$

$$A_{0i} = t_{\mathcal{U}_0} t_0^{(i)} v_2, \forall i \tag{4.16}$$

$$B_{0i} = t_{\mathcal{U}_0} t_0^{(i)} (x + E_0^{(i)} + (y + F_0^{(i)})m), \forall i \tag{4.17}$$

$$E_1^{(i)} = \frac{\rho_1 + \rho_1^{'(i)}}{k_1^{(i)}}, \forall i \tag{4.18}$$

$$F_1^{(i)} = r_1 + \alpha_1(\rho_1 + \rho_1^{'(i)}), \forall i \tag{4.19}$$

$$A_{1i} = t_{\mathcal{U}_1} t_1^{(i)} v_2, \forall i \tag{4.20}$$

$$B_{1i} = t_{\mathcal{U}_1} t_1^{(i)} (x + E_1^{(i)} + (y + F_1^{(i)})m), \forall i \tag{4.21}$$

$\mathcal{A}$ knows the values of x, y, v_2, $t_{\mathcal{U}_0}$, $t_{\mathcal{U}_1}$, r_0, r_1, ρ_0 and ρ_1. Thus, it is able to deduce the values of $t_0^{(i)}$ and $t_1^{(i)}$ and the linear system is simplified as follows:

$$C_0^{(i)} = \frac{F_0^{(i)} - r_0}{E_0^{(i)}} = \alpha_0 k_0^{(i)}, \forall i \tag{4.22}$$

$$C_1^{(i)} = \frac{F_1^{(i)} - r_1}{E_1^{(i)}} = \alpha_1 k_1^{(i)}, \forall i \tag{4.23}$$

During the challenge phase, $\mathcal{A}$ has access to left-or-right oracle LoRAN that returns modified credentials for user $\mathcal{U}_b$, where $b \in \{0, 1\}$. Thus, $\mathcal{A}$ obtains a new tuple that can be parsed as $E'_{r_b} = (g_2^{E_b}, g_2^{F_b})$, $\sigma'_{r_{\mathcal{IP}_b}} = (h_{\mathcal{U}_b}^{t_b v_2}, h_{\mathcal{U}_b}^{t_b(x+E_b+(y+F_b)m)})$ and $\mathtt{pk}'_b = g_2^{k_b \alpha_b}$, where $E_b = \frac{\rho_b + \rho_b^{'}}{k_b}$ and $F_b = r_b + \alpha_b(\rho_b + \rho_b^{'})$. As a strong adversary, $\mathcal{A}$ has access to the exponents of the ciphertext and the signature. Thus, after simplification with the variables it already knows, $\mathcal{A}$ gets the following system:

$$E_b = \frac{\rho_b + \rho'_b}{k_b} \tag{4.24}$$

$$F_b = r_b + \alpha_b(\rho_b + \rho_b^{'}) \tag{4.25}$$

$$A_b = t_{\mathcal{U}_b} t_b \tag{4.26}$$

To break the anonymity property, $\mathcal{A}$ should guess whether the value of α_b (resp. r_b and $t_{\mathcal{U}_b}$) is equal to α_0 (resp. r_0 and $t_{\mathcal{U}_0}$) or to α_1 (resp. r_1 and $t_{\mathcal{U}_1}$). Thus, when combining with the two previous linear systems, $\mathcal{A}$ attempts to solve a system with $2i + 3$ equations and $2i + 9$ unknown variables, i.e., r_b, α_b, ρ_b, ρ'_b, k_b, $t_{\mathcal{U}_b}$, t_b, α_0, α_1, $k_0^{(i)}$ and $k_1^{(i)}$,$\forall i$, which is impossible. Thus, PUBA satisfies the anonymity property. □

4.8 Performance Analysis

This section presents a proof of concept of PUBA including a full implementation of the different algorithms. It, first, introduces PUBA test-bed and then, discusses the theoretical and experimental results for the twelve algorithms, as depicted in Table 4.5.

Table 4.5: Communication, Storage and Computation Costs of PUBA's Algorithms

Algorithm	Running Device	Communication cost	Storage cost	Computation cost	Computation time in ms			
					$n = 300$	$n = 600$	$n = 900$	$n = 1200$
Set_Params	Device 1 ($\mathcal{TA}$)	$(2+n)\|\mathbb{Z}_q\|+\|\mathbb{G}_1\|+\|\mathbb{G}_2\|+\|\mathbb{G}_3\|$ [a]	$(2+n)\|\mathbb{Z}_q\|+\|\mathbb{G}_1\|+\|\mathbb{G}_2\|+\|\mathbb{G}_3\|$ [b]	γ_G	7	9	12	14
IPKeyGen	Device 1 ($\mathcal{TA}$)	($\mathcal{TA}$-$\mathcal{IP}$): $2\|\mathbb{G}_2\|$	($\mathcal{IP}$): $2\|\mathbb{Z}_q\|+2\|\mathbb{G}_2\|$	$2\mathrm{E}_{\mathbb{G}_2}$	3	3	3	3
UKeyGen	Device 2 ($\mathcal{U}$)	($\mathcal{U}$-$\mathcal{TA}$): $\|\mathbb{G}_2\|$	($\mathcal{U}$): $\|\mathbb{Z}_q\|+\|\mathbb{G}_1\|+\|\mathbb{G}_2\|$	$1\mathrm{E}_{\mathbb{G}_1}+1\mathrm{E}_{\mathbb{G}_2}$	3	3	3	3
SPKeyGen	Device 1 ($\mathcal{SP}$)	($\mathcal{SP}$-$\mathcal{TA}$): $\|\mathbb{G}_2\|$	($\mathcal{SP}$): $\|\mathbb{Z}_q\|+\|\mathbb{G}_2\|$	$1\mathrm{E}_{\mathbb{G}_2}$	2	2	2	2
Encrypt	Device 1 ($\mathcal{IP}$)	($\mathcal{IP}$-$\mathcal{U}$): $2n\|\mathbb{G}_2\|$	($\mathcal{IP}$): $n\|\mathbb{Z}_q\|+2n\|\mathbb{G}_2\|$	$2n\mathrm{E}_{\mathbb{G}_2}$	1464	2937	4443	5908
Sign	Device 1 ($\mathcal{IP}$)	($\mathcal{IP}$-$\mathcal{U}$): $(2n+2)\|\mathbb{G}_1\|$	($\mathcal{IP}$): $(2n+2)\|\mathbb{G}_1\|$	$(3n+2)\mathrm{E}_{\mathbb{G}_1}$	552	1103	1659	2229
VerifySig	Device 2 ($\mathcal{U}$)	–	($\mathcal{U}$): $(2n+2)\|\mathbb{G}_1\|+2n\|\mathbb{G}_2\|$	$n\mathrm{E}_{\mathbb{G}_2}+2\mathrm{P}$	721	1426	2127	2964
Enroll	Device 2 ($\mathcal{U}$)	($\mathcal{U}$-$\mathcal{SP}$): $2\|\mathbb{G}_1\|+(2n+1)\|\mathbb{G}_2\|$	($\mathcal{U}$): $(n+2)\|\mathbb{Z}_q\|+n\|\mathbb{G}_1\|+\|\mathbb{G}_2\|$	$(3n+4)\mathrm{E}_{\mathbb{G}_1}+(3n+1)\mathrm{E}_{\mathbb{G}_2}$	2984	5918	8877	12147
Verify	Device 1 ($\mathcal{SP}$)	–	($\mathcal{SP}$): $(2n+1)\|\mathbb{G}_2\|$	$n\mathrm{E}_{\mathbb{G}_2}+\mathrm{E}_{\mathbb{G}_3}+2\mathrm{P}$	516	1020	1522	2025
ComputeD	Device 1 ($\mathcal{SP}$)	($\mathcal{SP}$-$\mathcal{U}$): $\|\mathbb{G}_1\|+4\|\mathbb{G}_2\|$	($\mathcal{SP}$): $(n+2)\|\mathbb{G}_1\|+3\|\mathbb{G}_2\|$	$\mathrm{E}_{\mathbb{G}_1}+(4n+4)\mathrm{E}_{\mathbb{G}_2}$	2478	4960	7442	10001
ReEncryptD	Device 2 ($\mathcal{U}$)	($\mathcal{U}$-$\mathcal{SP}$): $(n+1)\|\mathbb{G}_1\|+2\|\mathbb{G}_2\|$	–	$(n+1)\mathrm{E}_{\mathbb{G}_1}+2\mathrm{E}_{\mathbb{G}_2}$	276	547	818	1102
Decide	Device 1 ($\mathcal{SP}$)	–	–	$(n+1)\mathrm{E}_{\mathbb{G}_1}+3\mathrm{E}_{\mathbb{G}_2}+2\mathrm{P}$	202	390	580	767

NOTE: $|\mathbb{G}_1|$ (resp. $|\mathbb{G}_2|$, $|\mathbb{G}_3|$ and $|\mathbb{Z}_q|$) indicates the size of an element in $\mathbb{G}_1$ (resp. $\mathbb{G}_2$, $\mathbb{G}_3$ and $\mathbb{Z}_q$); E and P stand for group exponentiations and pairing costs respectively; γ_G is the cost of the cyclic group generation; [a] indicates that the communication cost is between $\mathcal{TA}$ and each of the system's other entities; [b] indicates that the storage cost concerns all the system entities.

4.8.1 Test-bed and Methodology

For our experiments, we developed a prototype of PUBA that implements the four phases SETUP, IDENTITY_ISSUE, ENROLLMENT and VERIFICATION including the twelve algorithms[5]. The tests are made on the same test-beds presented in Chapter 3, Table 3.3 with the same programming language *Python*. The same measurements' conditions are also considered.

In our experiments, Device 1 (resp. Device 2) is used to test algorithms run by $\mathcal{TA}$, $\mathcal{IP}$, and $\mathcal{SP}$ (resp. $\mathcal{U}$). We also consider biometric templates of lengths; 300, 600, 900 and 1200, respectively. Indeed, each type of biometric trait is characterized by a specific number of features, thus a different length of the template vector. For instance, a fingerprint template could be of length 300 or 600, i.e., a template of length 600 offers more accuracy [146].

4.8.2 Communication and Storage Costs

This section discusses the communication and storage costs of PUBA. As depicted in Table 4.5, both costs are evaluated according to the size of group elements $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ and $\mathbb{Z}_q$.

From Table 4.5, it is worth noticing that the IDENTITY_ISSUE phase is the most consuming in terms of bandwidth and storage. Indeed, it includes the Encrypt and

[5]The source code is available at https://github.com/soumasmoudi/PUBA

Sign algorithms that introduce communication overheads of $2n|\mathbb{G}_2|$ and $(2n+2)|\mathbb{G}_1|$, respectively. These two algorithms are performed once for each user. At the end of the IDENTITY_ISSUE phase, the user stores his credentials of size $(2n+2)|\mathbb{G}_1|+2n|\mathbb{G}_2|$.
The ENROLLMENT phase offers acceptable communication overhead as it is performed once per user and per service provider. In terms of storage, the user is required to have a storage capacity of $(n+2)|\mathbb{Z}_q|+n|\mathbb{G}_1|+|\mathbb{G}_2|$, per enrollment session. While, the service provider is only required to store the user's credentials of size $(2n+1)|\mathbb{G}_2|$, which refer to the size of the encrypted template and the randomized public key.
The VERIFICATION phase, including the ComputeD, ReEncryptD and Decide algorithms repeatedly performed between the user and the service provider, has an acceptable communication overhead w.r.t. to the security it ensures (i.e., verification of computations performed by the user). Per verification session, only the service provider is required to have a storage capacity of $(n+2)|\mathbb{G}_1|+3|\mathbb{G}_2|$. This storage memory is discharged at the end of the session.

4.8.3 Computation Overhead

This section presents the theoretical and experimental computation costs of PUBA's algorithms.

Table 4.5 shows that the computation cost of all PUBA's algorithms, except those of the SETUP phase, varies w.r.t. the length n of the template. All algorithms are consuming in terms of exponentiation, while the number of pairing operations is constant for the VerifySig, Verify and Decide algorithms (i.e., two pairing operations are needed). Indeed, to generate a user's credentials, using Device 1, $5n+2$ exponentiations are needed. For the ENROLLMENT phase, the Enroll algorithm is the most consuming one as it requires $6n+5$ exponentiations, on Device 2. For the VERIFICATION phase, $5n+9$ exponentiations are required on Device 1, and $n+3$ exponentiations are needed to perform the ReEncryptD algorithm on Device 2.

Table 4.5 also depicts the computation times of the different algorithms on Device 1 and Device 2, w.r.t. the tests' conditions described in Section 4.8.1. Figures 4.4 and 4.5 illustrate the impact of the template length on the processing times. Indeed, it is worth noticing that the processing times depend on the length of biometric templates. For all algorithms, the processing time is a rising affine/linear function of the template length, which is consistent with the costs expressed in Table 4.5.

On Device 1, Encrypt and ComputeD are the most consuming algorithms. For instance, the encryption of a template, by the $\mathcal{IP}$, requires a processing time varying from approximately $1,5s$ to $5s$, when the template length varies from $n=300$ to $n=1200$. These results are acceptable as the Encrypt algorithm is performed once for each user. The Sign algorithm is also run once by the $\mathcal{IP}$ for each user, but it is less consuming than the Encrypt algorithm. From these two algorithms, we deduce that the generation of a user's credentials requires approximately $8s$ when considering a reference template of length $n=1200$. The computation time of the ComputeD algorithm has the highest slope w.r.t. the template length, i.e., it varies from $2,5s$

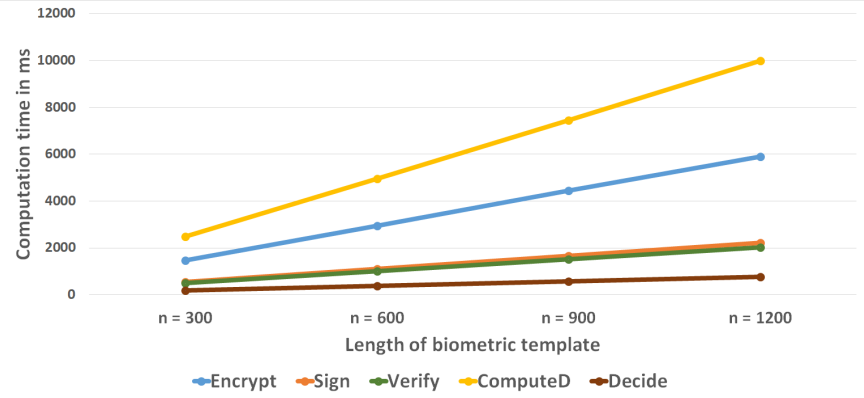


Figure 4.4: Processing Times on Device 1

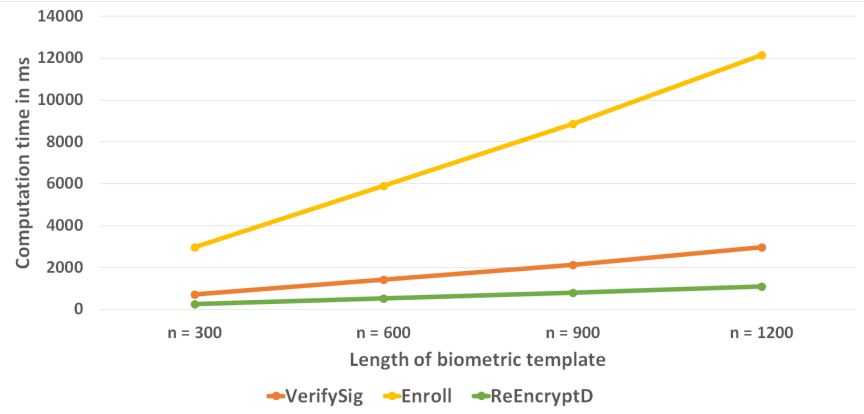


Figure 4.5: Processing Times on Device 2

to $10s$ when the template length varies from $n = 300$ to $n = 1200$. These results are plausible as the ComputeD algorithm includes an encryption of an n-length template and an homomorphic evaluation over two templates. However, they must be put into perspective and improved relying on advanced hardware features and paralleled calculations at the $\mathcal{SP}$ side. The Verify and Decide algorithms demonstrate reasonable computation times that could be also improved using advanced hardware features.

On Device 2, the Enroll algorithm is the most consuming one as it includes a large number of exponentiations. The processing time varies from approximately $3s$ to $12s$, when the template length n varies from 300 to 1200. These results are acceptable as the Enroll algorithm is performed once for multiple authentication sessions. The VerifySig algorithm is performed only once by $\mathcal{U}$ at the reception of his credentials, and offers acceptable computation times. Indeed, to verify the credentials associated to a template of length $n = 300$ (resp. $n = 600$, $n = 900$ and $n = 1200$), a user needs $0,5s$ (resp. $1s$, $1,5s$ and $2s$). The ReEncryptD algorithm is the less consuming one. For instance, even for a template of length $n = 1200$, the computation time does not exceed one second. This result demonstrates the practical usability of PUBA for continuous authentication.

Finally, from theoretical and experimental costs depicted in Table 4.5, it is worth mentioning that an important computation time is consumed to randomly generate

elements in $\mathbb{Z}_q$, mainly for the most consuming algorithms, i.e., Encrypt, Enroll and ComputeD.

4.9 Conclusion

In this chapter, a novel biometric authentication scheme for user-centric identity management, PUBA, is introduced. Thanks to malleable signatures and polymorphic encryption, PUBA enables users to derive several unlinkable biometric credentials from a certified one to enroll at different service providers. PUBA also offers, to service providers, security guarantees against both the use of fake biometric templates during enrollment, and the uncontrolled computation by users during authentication.
Detailed algorithms of PUBA are provided based on the El Gamal cryptosystem and the Pointcheval-Sanders signature scheme. The proposed construction is proven to be secure and to satisfy strong privacy requirements under standards assumptions. A proof of concept of PUBA, through the implementation of the different algorithms, demonstrates the feasibility of the proposed authentication scheme and its practical usability for continuous authentication. Computation time required by the user during the verification phase, does not exceed 1 second for a biometric template of 600 samples. Thus, with the development of European and international legislation, in terms of privacy, privacy-preserving biometric authentication schemes, like PUBA, should be deployed in real world systems.

While PUBA is designed for controlling physical access to services and resources, identity management systems can be also designed to enable users share data among different parties. Thus, in the next chapter, we will consider identity management systems from a data sharing perspective. A particular type of identities will be considered referred to as ephemeral identities. For this purpose, we present a novel privacy-preserving proximity tracing protocol for e-healthcare systems that allows to securely exchange and share correct contact information.

Chapter 5

A Secure and Privacy-preserving Proximity-tracing Protocol for E-healthcare Systems

Be safe, be smart, be kind!

– Tedros Adhanom Ghebreyesus

In this chapter, we provide an alternative to the proximity-tracing application that the traveler Alice uses to be notified in case of contact with infected persons. Relying on the novel proximity-tracing application, Alice is able to share contact information with people in proximity independently from the centralized authority. In turn, Alice is ensured to receive only correct alerts in case of being at risk of contamination. Moreover, in case of infection, Alice shares her contact information with dedicated authorities, namely health authorities, that are not able to identify the persons with whom she has been in contact. Thus, no social graph can be built and users' privacy is preserved.

5.1 Introduction

IDENTITY management systems have been intensively used in healthcare to ensure the protection of sensitive medical data and to help users share this data among different parties. The recent health crisis has boosted the use of IDM systems for data sharing, mainly when relying on proximity-tracing solutions to control the contamination chain among the population. These solutions are aimed at sharing valuable data while preserving users' privacy. However, there are still privacy threats and robustness challenges as long as users are required to disclose and share correct sensitive and personal information with different third parties with various levels of trust.

In this context, most of the proposed solutions rely on the Bluetooth technology, namely Bluetooth Low Energy (BLE), to exchange contact information, thanks to its efficiency in active communications [120]. Among the governmental solutions, the TraceTogether application [66] has been launched by Singapore. TraceTogether enables to collect, via the Bluetooth technology, temporary IDs (generated by a central trusted server) of users in close proximity. Collected IDs are stored in an encrypted form using the server's public key, at users' devices, and in case of infection, they are shared with the server. The COVIDSafe application [12] from the Australian government is another Bluetooth-based solution. It also logs encrypted users' contact information, and share them once an infection is detected. The server is required to alert users at risk without revealing the identity of the infected user. Both TraceTogether and COVIDSafe applications are set upon a centralized architecture. Many other applications like Stop COVID-19 (Croatia) [135], CA Notify (California) [28] rely on the Google and Apple Exposure Notification (GAEN) service [7], which is set upon a decentralized architecture. Although contact tracing applications have helped governments to alleviate the widespread of the pandemic by automating the manual contact tracing done by health authorities, they raise critical privacy concerns, namely users tracking and identification [88].

Academic solutions have been also proposed to support both centralized [70] and decentralized [36, 37, 124, 74, 112] architectures. However, each architecture has its merits and limits in terms of security and privacy. Indeed, as discussed in Chapter 2, Section 2.2.3.4, using centralized solutions, users guarantee the reception of correct alerts as long as the generation of users' contact tokens and the verification in case of infection are performed by a centralized server. This guarantee is compensated with threats to users' privacy, i.e., users are exposed to tracking and identification of their contact lists by the centralized server. Decentralized solutions have been proposed to mitigate these privacy threats. Users are responsible for generating their contact tokens in order to ensure their privacy and anonymity, but, they are not prevented from forging contact information, which results in high level of false alerts. To get the best of both architectures, hybrid architecture based solutions [46, 67] have been proposed. However, there are still security and privacy concerns that have not been yet addressed, like ensuring both the correctness of contact information and the anonymity of contacted users.

In this chapter, we present SPOT, a novel hybrid Secure and Privacy-preserving

prOximiTy-based protocol for e-healthcare systems. It combines a decentralized proxy front-end architecture, ensuring both users' anonymity and contact information integrity, and a centralized back-end computing server and guaranteeing a real time verification of contact information integrity. Spot assumes that two users in close proximity rely on their Ephemeral Bluetooth IDentifiers (EBID) to compute a common contact message. This message is relayed to a central server through a group of proxies. With the help of the computing server and relying on a proof-based group signature, Spot prevents users from forging their contact lists. The signed contact messages are given to the user to be locally stored. In case of a detected infection, the user consents to share his contact list, i.e., a set of signed contact messages, with the health authority. This latter checks the correctness of the received list and shared it back with all the involved users, if the verification holds. The originality of this third contribution is manifold:

- we design a proximity protocol for e-health services that prevents the injection of false positives, i.e., alert users to be at risk when they are not. Spot enforces the verification of the correctness and the integrity of users' contact information by health authorities, thanks to the support of both a computing server and a group of proxies.

- we guarantee strong privacy properties namely the anonymity of users being in contact with infected people, and the unlinkability of users' transactions when relying on random EBIDs that can neither be linked to each other nor be linked to their issuer.

- we propose a novel group signature that offers an efficient, aggregated and batch verification over multiple Groth-Sahai Non-Interactive Witness-Indistinguishable proofs [64]. Relying on these proofs, proxies able to prove to the health authority the integrity of contact information and proxies' keys without revealing the signing keys. For efficiency reasons, the health authority is given the capacity to proceed to the verification of multiple proofs at once. If the batch verification fails, then she proceeds to a *divide and conquer* verification. She splits the list of proofs into sub-lists and performs verification to each sub-list recursively until all invalid signatures are identified.

- we propose a concrete construction of the Spot protocol relying on a structure-preserving signature scheme [2] that supports securely signing group's elements, i.e., contacts' information, and the novel group signature scheme that supports batch and aggregated verification over multiple NIWI proofs.

- we evaluate the performances of Spot through the full implementation of different procedures and algorithms. The conducted experiments have shown acceptable computation times proving the practical usability of the proposed solution and its efficiency as the batch verification reaches a gain of up to 50% compared to the naive verification.

- we present a demonstrator of Spot to evaluate the feasibility of Spot in real use-cases.

This chapter is organized as follows. Section 5.2 introduces proximity-tracing and batch verification, and compares most closely-related proximity-tracing algorithms and solutions to Spot. Section 5.3 gives an overview of Spot. After introducing the underlying building blocks in Section 5.4, a novel group signature scheme is proposed in Section 5.4.5. The concrete construction of the proposed Spot protocol is presented in Section 5.5. Section 5.6 and Section 5.7 provide security and privacy properties and a detailed discussion of Spot's conducted experiments. We present, in Section 5.8, a demonstrator of Spot, before concluding in Section 5.9.

5.2 Related Work

This section, first, reviews existing proximity-tracing protocols and gives a detailed comparison with the Spot protocol. Second, it gives an overview about batch verification proposals over multiple signatures.

Proximity-tracing Recently, several industrial and research contact tracing solutions have been proposed for e-health applications [128, 120]. These solutions aim at ensuring security and privacy properties, namely:

- *(i)* **anti-replay** mitigating the multi-submission of the same contact information,
- *(ii)* **unforgeability** preventing malicious entities[1] from threatening data integrity (cf. property **(S2)**, Chapter 2, Section 2.2.2.1),
- *(iii)* **unlinkability** between users' different transactions (i.e., contact information), which refers to the multi-show unlinkability of property **(P4)**, and
- *(iv)* **anonymity** of end-users involved in contact with an infected person, which refers to anonymity property **(P1)**.

Note that a formal definition of security and privacy requirements is given in Section 5.3.4.

Researchers from Inria, France, and Fraunhofer, Germany proposed Robert [70], a contact tracing protocol that relies on a centralized architecture, where a central server delivers pseudonyms to users. Each user collects pseudonyms of users in close proximity and shares them with the server when being infected. In such centralized solution, users are sure that they receive correct alerts (i.e., collected pseudonyms are neither replayed nor falsified by malicious entities), however, their privacy is threatened as long as the server is able to identify users' contacts and to track them.
In [36], Troncoso *et al.* introduced the Decentralized Privacy-Preserving Proximity Tracing (DP-3T) solution which is one of the most popular contact-tracing protocols. DP-3T has been proved to mitigate the privacy threats of centralized solutions as there is no need for a central entity which collects users' contact information with the risk of

[1]Malicious entities involve either a single malicious adversary or colluding adversaries.

tracking them. However, it does not prevent relay and replay attacks and gives no mean to verify the correctness of contact information. Thus, users are exposed to false alerts from malicious entities either by creating falsified information or replaying information of previous sessions.
Afterwards, Castelluccia *et al.* proposed Desire [46], a proximity tracing protocol that leverages the advantages of the centralized and decentralized solutions. However, some security and privacy issues have not been considered in this solution. First malicious users are able to collude and merge their contact lists, which leads to false positive injection attacks. Second, the server requested to compute the exposure status and risk, is able to link users' requests, and to de-anonymize them.
Two very similar proposals named PACTs are also introduced. The east coast PACT [124] and the west coast PACT [37] are very close to DP-3T. The two solutions rely on random pseudonyms derived from a private seed, that are broadcasted to users in proximity via Bluetooth. The pseudonyms are generated using cryptographic pseudorandom generators and pseudorandom functions. Apart from the non-resistance against replay attacks, these two proposals give no mean to check the correctness of the contact information before being broadcasted.
Pietrzak [112] proposed a decentralized contact-tracing solution to mitigate replay attacks against DP-3T. However, privacy concerns are raised, namely tracking users, as geo-location and time of contacts are shared within the Bluetooth message.
In [67], Hoepman proposed two tracing protocols, the first one relies on an interactive session between two users in proximity to register contact information. If the interaction fails, the contact is not registered. Thus, the second protocol comes to mitigate this risk of failure and relies on an authority that relays information between users. In both protocols, the identities of users who have been in contact with an infected person, are revealed to a central entity, which contradicts the defined anonymity requirement.
Liu et al. [74] use zero-knowledge proofs and group signatures in order to preserve users' privacy. Zero-knowledge proofs are generated by users over the contact information they collected. Indeed, users prove the contacts to their doctors without revealing the information. Afterwards, doctors, being members of a group, sign the proofs and publish them in a public board. Then, relying on their secrets, users can check if they were in contact with infected people. As such, no entity can identify the contacts of an infected user. However, based on a long interactive protocol between two devices, the collection of contacts' information may result in a failed interaction, thus causing the non-registration of the contact. Furthermore, the authors only consider honest but curious adversaries, which leads to possible false alerts due to malicious colluding users. It is also worth noticing that, unlike Spot and other related work [46, 36, 112], [74] assume that all the computations (handshake, zero-knowledge, verification) are performed by the user's device, which leads to device's battery depletion. A new contact tracing strategy based on online social networking is proposed in [147] but does not provide privacy guarantees.

Table 5.1 provides a comparative summary between Spot and related work in terms of architecture settings and properties. As shown, the reviewed solutions do not prevent the injection of false positive alerts either while being vulnerable to either malicious users [74, 67, 36, 124, 37, 46, 112] or replay attacks [124, 37]. Additionally, apart from [46], other works do not ensure both unlinkability and anonymity properties. Thus, we

propose SPOT a novel proximity-tracing protocol which supports strong security and privacy requirements.

Table 5.1: Comparison between SPOT and Related Work

		SPOT	[74]	[67]	[36]	[124] [37]	[70]	[46]	[112]
Architecture	Centralized	-	-	-	-	-	✓	-	-
	Decentralized	-	✓	-	✓	✓	-	-	✓
	Semi-centralized	✓	-	✓	-	-	-	✓	-
Properties	Unforgeability [a]	✓	✓[b]	✓[b]	✗	✗	N.A.	✗	✗
	Anti-replay	✓	✓	✓	✗	✗	✓	✓	✓
	Unlinkability	✓	N.A.	✗	✓	✓	✗	✓	✗
	Anonymity	✓	✓	✗	✗	✗	✗	✓	✗

NOTE: N.A. is the abbreviation for Not Applicable; [a] indicates that the unforgeability implies the integrity of users' contact information and the prevention of false positives injection; [b] indicates that unforgeability is partially satisfied while not considering malicious colluding entities.

Batch Verification Giving consideration to the huge number of collected proofs, in many applications, and to the processing time needed to verify a single proof, there is a crucial need to optimize the verification algorithm by verifying multiple proofs at once. To this question, batch verification has been introduced by Naccache *et al.* [103] enabling the verification of multiple DSA-type signatures generated by different signers. Since then, several batch verification methods have been proposed for other digital signature schemes, namely for group signatures. Indeed, batch verification over group signatures was introduced by Ferrara *et. al* [60]. Wasef and Shen proposed to use batch verification for vehicular ad hoc networks [143]. In [82], authors exploited Ferrara *et. al* scheme to build a new batch vertification scheme that supports invalid signatures identification. They rely on the *divide-and-conquer* approach [109]. In [59], authors presented a group signature scheme with batch verification that allows to deal with the excessive need for signatures verification in pervasive social networking. The proposed scheme do not support bad signatures identification, i.e., if the batch verification fails, all the signatures are rejected. Recently, Alamer proposed a secure and privacy-preserving group signature scheme supporting batch verification. It aims at mitigating the increasing computation delay in IoT systems [4]. In [149], authors designed a novel group signature scheme with batch verification for IoT consortium blockchain. It suggests two types of verification, i.e., a naive verification for urgent transactions and batch verification for ordinary ones.
To the best of our knowledge, no batch verifier has been constructed over Proof of Knowledge (PoK) based group signature schemes, i.e., signatures that endorse the verification of the signer key.

5.3 System and Threat Models

This section first presents the involved entities and gives an overview of SPOT. Then, it details the system model with its procedures and algorithms and defines the threat

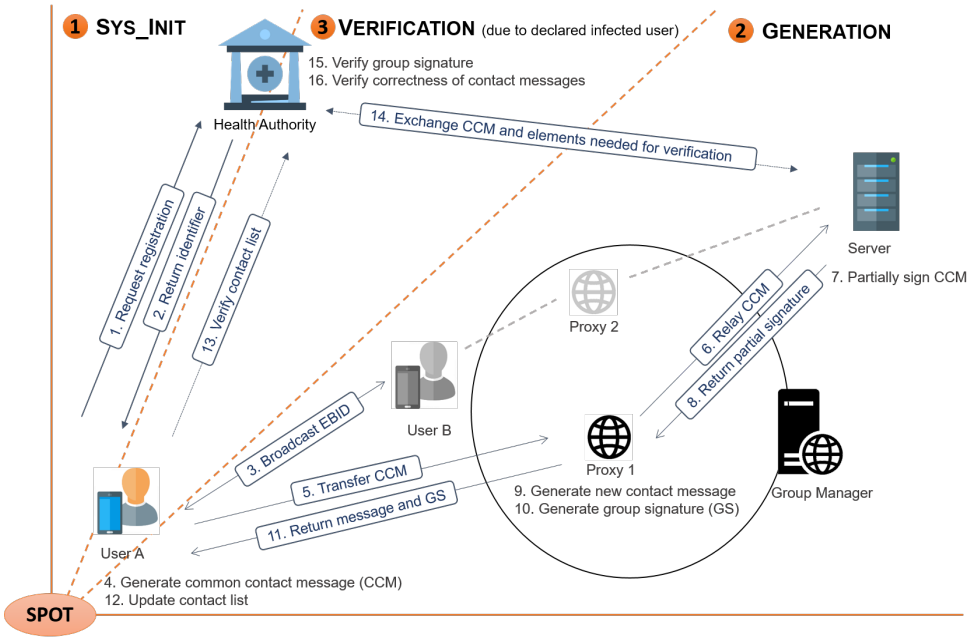


Figure 5.1: Overview of the SPOT Protocol

model through formal security games.

5.3.1 Entities

Figure 5.1 depicts the four main actors involved in SPOT with their interactions according to the different phases.

- The user ($\mathcal{U}$) represents the entity that owns the device where the proximity-tracing application is installed. During the GENERATION phase, $\mathcal{U}$ broadcasts his EBID (Ephemeral Bluetooth IDentifier), collects the EBIDs of other users in proximity, and computes a common contact message shared between each two users being in contact. $\mathcal{U}$ wants to receive alerts if he was in contact with confirmed cases.

- The Health Authority ($\mathcal{HA}$) is responsible for issuing users' identifiers during the SYS_INIT phase, and for checking the correctness of the contact messages provided by an infected user during the VERIFICATION phase.

- The Server ($\mathcal{S}$) is responsible for anonymously collecting and storing users' contact messages relayed by proxies during the GENERATION phase. $\mathcal{S}$ performs a real-time verification of the received contacts during the GENERATION phase, in order to help $\mathcal{HA}$ to verify the correctness and integrity of the contact messages.

- The Proxy ($\mathcal{P}$) is considered as a member of a group of proxies managed by the group manager ($\mathcal{GM}$). Proxies form an intermediate layer by relaying the common contact messages of users to $\mathcal{S}$ in order to ensure the anonymity of

involved users towards the server during the GENERATION phase. Proxies also play an important role in ensuring data integrity and user geolocation privacy thanks to group signatures.

Remark 2. The Server can be distributed by considering one or several servers per geographical area, each server participating in locally storing part of users' contact messages databases. All the parts are then collected on offline in a centralized server. Thus, for ease of presentation, we consider only one server.

Remark 3. Proxies are distributed over several geographical areas. We assume that a load-balancing is established between at least two proxies in the same geographical area to ensure the system availability in case of failure or overload. More precisely, proxies in a same geographical area are separated into two subsets - a primary and a secondary - and two users in a contact interaction must contact proxies belonging to different subsets in order to prevent a proxy from gaining too much knowledge about users' interactions.

5.3.2 Overview

SPOT is set upon an hybrid architecture that leverages the best of the centralized and decentralized settings in proximity-tracing protocols. It relies on a proxy-based solution to preserve users' privacy (i.e., users remain anonymous towards the server, thus preventing users' tracking) and is based on a semi-trusted computing server to ensure data consistency and integrity (i.e., users are ensured that the received alerts are correct). The architecture of the proposed protocol is depicted in Figure 5.1. SPOT involves three main phases: SYS_INIT, GENERATION and VERIFICATION presented hereafter.

The SYS_INIT phase consists of initializing the whole system. It relies on seven algorithms, referred to as $\mathsf{Set\_params}$, $\mathsf{HA\_keygen}$, $\mathsf{S\_keygen}$, $\mathsf{Setup\_ProxyGr}_{\mathcal{GM}}$ and $\mathsf{Join\_ProxyGr}_{\mathcal{P}/\mathcal{GM}}$, $\mathsf{Set\_UserID}_{\mathcal{HA}}$ and $\mathsf{Userkeygen}_{\mathcal{U}}$. During the SYS_INIT phase, a trusted authority[2] generates the system public parameters published to all involved entities and the pair of keys of both $\mathcal{HA}$ and $\mathcal{S}$, relying on $\mathsf{Set\_params}$, $\mathsf{HA\_keygen}$ and $\mathsf{S\_keygen}$ algorithms. During this phase, the group manager defines the group of proxies. It generates the group signature parameters using the $\mathsf{Setup\_ProxyGr}_{\mathcal{GM}}$ algorithm and it interacts with each group member to derive the associated keys relying on the $\mathsf{Join\_ProxyGr}_{\mathcal{P}/\mathcal{GM}}$ algorithm. The Health Authority is also involved in this phase to register a user when installing the proximity-tracing application. $\mathcal{HA}$ generates a specific secret value $t_{\mathcal{U}}$ (only known by $\mathcal{HA}$) and a unique identifier $\mathtt{ID}_{\mathcal{U}}$ for each user ($\mathcal{U}$), using the $\mathsf{Set\_UserID}_{\mathcal{HA}}$ algorithm. Finally, $\mathcal{U}$ uses his identifier to generate his pair of keys relying on the $\mathsf{Userkeygen}_{\mathcal{U}}$ algorithm. The user's identifier $\mathtt{ID}_{\mathcal{U}}$, secret value $t_{\mathcal{U}}$ and public key are stored in a database DB_{USER} owned by $\mathcal{HA}$. We note that the trusted authority, the group manager and proxies are involved only once in the

[2]For ease of presentation, the trusted authority is neither presented in Figure 5.1 nor in the system's model entities.

SYS_INIT phase, while the health authority must intervene every time a user wants to register.

The GENERATION phase occurs when two users $\mathcal{U}_A$ and $\mathcal{U}_B$ are in contact. It represents the process of generating contact messages and contact lists for users. Three main entities participate in this phase relying on three different algorithms, referred to as $\mathsf{Set\_CCM}_{\mathcal{U}}$, $\mathsf{S\_PSign}_{\mathcal{S}}$ and $\mathsf{P\_Sign}_{\mathcal{P}}$. At first, $\mathcal{U}_A$ and $\mathcal{U}_B$ execute the $\mathsf{Set\_CCM}_{\mathcal{U}}$ algorithm to generate a common contact message relying on their random $EBID$s (denoted by $\mathtt{D}_A^e$ and $\mathtt{D}_B^e$) for an epoch e[3]. $\mathcal{U}_A$ and $\mathcal{U}_B$ choose two different proxies to relay their common contact message to the server. For this purpose, they compare their $EBID$s, i.e., if $\mathtt{D}_A^e > \mathtt{D}_B^e$, $\mathcal{U}_A$ chooses the first proxy and $\mathcal{U}_B$ selects the second one, and vice versa. Each of the two proxies relays the common contact message to the server. $\mathcal{S}$ checks if the two copies are similar. If so, $\mathcal{S}$ executes the $\mathsf{S\_PSign}_{\mathcal{S}}$ algorithm to partially sign the common contact message, thus proving that the contact message correctly reached the server. Afterwards, given back only a correct message, each proxy executes the $\mathsf{P\_Sign}_{\mathcal{P}}$ algorithm. Indeed, each proxy extends the message, given by $\mathcal{S}$, with the corresponding user's identifier and it signs the resulting message on behalf of the group. He, finally, sends back the message and the corresponding group signature to the user and closes the communication session, while removing all the exchanged and generated contact information. The user adds the group signature, along with the common contact message, the date, time and duration of contact, in his contact list $CL_{\mathcal{U}}$. Note that each contact information is stored for Δ days.

The VERIFICATION phase is run by the health authority to check the correctness of the contact lists of infected users during a period of time t. To this end, $\mathcal{HA}$ performs three successive verifications. During the first verification, $\mathcal{HA}$ checks if, in his DB_{USER} database, $\mathcal{U}$ is infected[4]. For the two other verifications, we propose two options, (i) a naive verification to check the correctness of a single contact message and (ii) a batch verification to verify the correctness of multiple contact messages belonging either to the same user or to different users.

- *Naive Verification:* relies on two algorithms, referred to as $\mathsf{Sig\_Verify}_{\mathcal{HA}}$ and $\mathsf{CCM\_Verify}_{\mathcal{HA}}$. That is, $\mathcal{HA}$, first, checks the validity of a single group signature relying on the $\mathsf{Sig\_Verify}_{\mathcal{HA}}$ algorithm. Then, $\mathcal{HA}$ verifies that the contact messages have been correctly generated and have successfully reached $\mathcal{S}$, using the $\mathsf{CCM\_Verify}_{\mathcal{HA}}$ algorithm.

- *Batch Verification:* involves three algorithms, referred to as $\mathsf{Batch\_Sig\_Verify}_{\mathcal{HA}}$, $\mathsf{Agg\_Sig\_Verify}_{\mathcal{HA}}$ and $\mathsf{Batch\_CCM\_Verify}_{\mathcal{HA}}$. That is, $\mathcal{HA}$ carries out the $\mathsf{Batch\_Sig\_Verify}_{\mathcal{HA}}$ algorithm to verify multiple proofs over contact messages sent by multiple users, in a single verification. If the batch signature verification fails, $\mathcal{HA}$ should proceed progressively by dividing the list of contacts in sub-lists

[3]An epoch e denotes a period of time in which the Bluetooth identifier (EBID) remains unchanged.

[4]We suppose that the health status of a user is updated when being tested. Indeed, to be tested, $\mathcal{U}$ has to provide an encrypted form of his identifier $\mathtt{ID}_{\mathcal{U}}$ (i.e., $\mathtt{ID}_{\mathcal{U}}$ is encrypted meaning the $\mathcal{HA}$ public key). Afterwards, the analysis' result is sent with the encrypted identifier to $\mathcal{HA}$, that extracts the identifier and updates the user's health status in the DB_{USER} database.

and then by verifying the invalid sub-list message by message. To verify a single message, $\mathcal{HA}$ performs the $\mathsf{Agg\_Sig\_Verify}_{\mathcal{HA}}$. To check that all contact messages, contained in a user's contact list, have successfully reached $\mathcal{S}$, in a single transaction, $\mathcal{HA}$ runs the $\mathsf{Batch\_CCM\_Verify}_{\mathcal{HA}}$ algorithm.

It is worth mentioning that if one of the verifications given above fails, the contact message is rejected. Otherwise, $\mathcal{HA}$ collects all verified messages of all infected users in a set $S_{\texttt{CCM}}$ that she signs. Note that for each period of time t, $\mathcal{HA}$ removes users' contact lists after verifications. $S_{\texttt{CCM}}$ and the corresponding signature are sent to the server that shares them with all users. To compute the risk score, each user compares the set $S_{\texttt{CCM}}$ with his contact list, taking into account the number of infected users being contacted and the contact duration.

5.3.3 System Phases

Based on the three phases, Figure 5.2 presents the chronological sequence of fifteen PPT algorithms, defined below. For ease of presentation, we consider only one proxy in the sequence diagram and we illustrate the GENERATION only for user $\mathcal{U}_A$. That is, the two users $\mathcal{U}_A$ and $\mathcal{U}_B$ have been in contact and exchanged their EBIDs. For the VERIFICATION phase, we suppose that both users have received a negative analysis' result.

SYS_INIT phase

- $\mathsf{Set\_params}(\lambda) \rightarrow pp$ – run by a trusted authority. Given the security parameter λ, this algorithm generates the system public parameters pp that will be considered as a default input for all the following algorithms.
- $\mathsf{j\_keygen}() \rightarrow (\texttt{sk}_j, \texttt{pk}_j)$ – performed by a trusted authority. It returns the pair of keys $(\texttt{sk}_j, \texttt{pk}_j)$ of j where $j = \{\mathcal{HA}, \mathcal{S}\}$.
- $\mathsf{Setup\_ProxyGr}_{\mathcal{GM}}() \rightarrow (\texttt{sk}_g, \texttt{vk}_g)$ – this algorithm is performed by the group manager to set up the group signature. It returns the proxies' group verification key $\texttt{vk}_g$ represented as $(\texttt{pk}_g, \Sigma_{\mathsf{NIWI}})$, where $\texttt{pk}_g$ is the public key of the group manager and Σ_{NIWI} is the Common Reference String CRS of a NIWI proof [64]. The $\mathsf{Setup\_ProxyGr}_{\mathcal{GM}}$ algorithm also outputs the secret key $\texttt{sk}_g$ that is privately stored by $\mathcal{GM}$.
- $\mathsf{Join\_ProxyGr}_{\mathcal{P}/\mathcal{GM}}(\texttt{sk}_g) \rightarrow (\texttt{sk}_p, \texttt{pk}_p, \sigma_p)$ – this algorithm is performed through an interactive session between the proxy and the group manager. It takes as input the secret key $\texttt{sk}_g$, and outputs the pair of keys $(\texttt{sk}_p, \texttt{pk}_p)$ of $\mathcal{P}$ belonging to the group (i.e., $\mathcal{P}$ is responsible for generating his pair of keys), and a signature σ_p over $\mathcal{P}$'s public key $\texttt{pk}_p$ (i.e., σ_p is generated by $\mathcal{GM}$).
- $\mathsf{Set\_UserID}_{\mathcal{HA}}() \rightarrow (t_{\mathcal{U}}, \texttt{ID}_{\mathcal{U}})$ – this algorithm is run by $\mathcal{HA}$ and returns a secret value $t_{\mathcal{U}}$ specific for $\mathcal{U}$ and the identifier $\texttt{ID}_{\mathcal{U}}$ of $\mathcal{U}$.

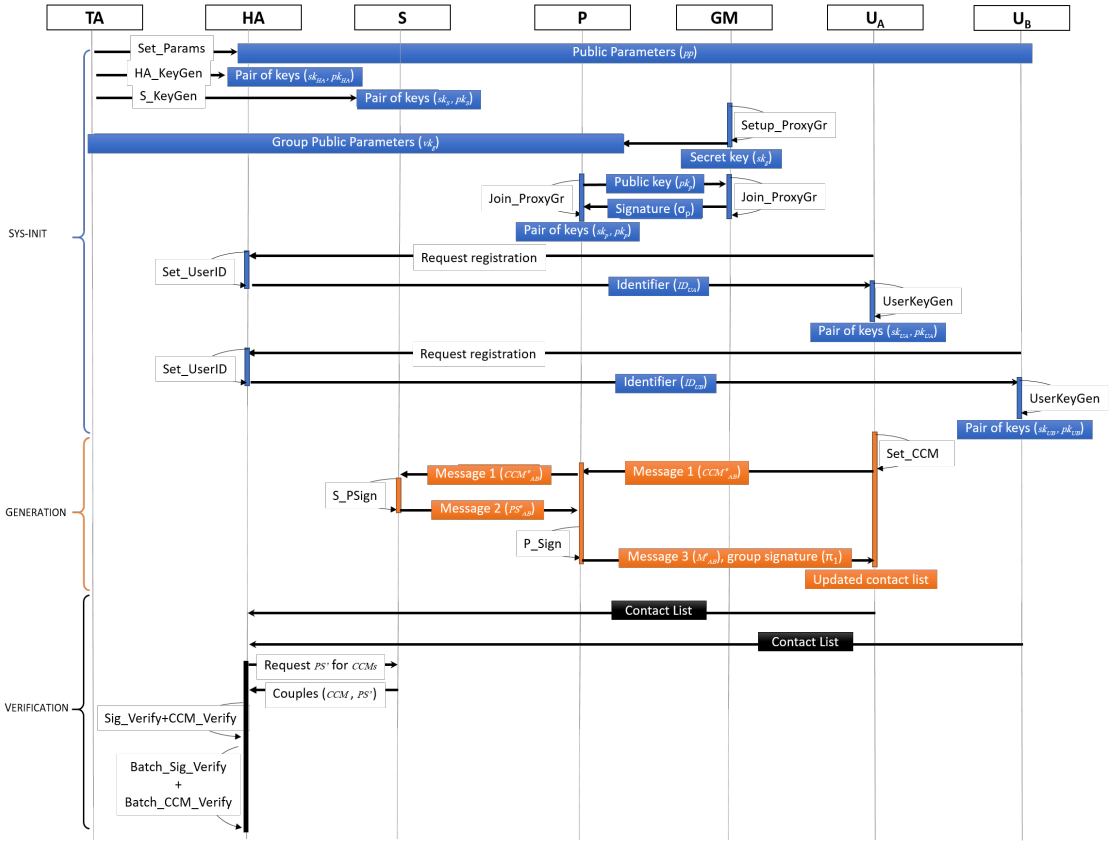


Figure 5.2: Workflow of the SPOT Protocol

- $\mathsf{Userkeygen}_{\mathsf{U}}(\mathtt{ID}_{\mathcal{U}}) \rightarrow (\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$ – performed by $\mathcal{U}$ to set his pair of keys $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$ relying on the identifier $\mathtt{ID}_{\mathcal{U}}$.

Generation phase

- $\mathsf{Set\_CCM}_{\mathcal{U}}(\mathtt{D}^e_A, \mathtt{D}^e_B) \rightarrow \mathtt{CCM}^e_{AB}$ – run by each of two users $\mathcal{U}_A$ and $\mathcal{U}_B$ being in contact during an epoch e. Given two Bluetooth identifiers $\mathtt{D}^e_A$ and $\mathtt{D}^e_B$, this algorithm generates a common contact message $\mathtt{CCM}^e_{AB}$.
- $\mathsf{S\_PSign}_{\mathcal{S}}(\mathtt{CCM}^e_{AB}, \mathtt{sk}_{\mathcal{S}}) \rightarrow (\mathtt{PS}^e_{AB}, \mathtt{PS}'^e_{AB})$ – run by $\mathcal{S}$. Given a common contact message $\mathtt{CCM}^e_{AB}$ sent by $\mathcal{U}_A$ and $\mathcal{U}_B$ through two different proxies $\mathcal{P}_1$ and $\mathcal{P}_2$, this algorithm outputs the couple $(\mathtt{PS}^e_{AB}, \mathtt{PS}'^e_{AB})$ that is stored with $\mathtt{CCM}^e_{AB}$ at $\mathcal{S}$, for Δ days. Note that only $\mathtt{PS}^e_{AB}$ is given back to $\mathcal{P}_1$ and $\mathcal{P}_2$ to prove that $\mathtt{CCM}^e_{AB}$ has been successfully received and verified by $\mathcal{S}$ (i.e., a real contact took place), while $\mathtt{PS}'^e_{AB}$ is kept secret at $\mathcal{S}$ and is sent only to $\mathcal{HA}$ to check the correctness of a contact message provided by a infected user.
- $\mathsf{P\_Sign}_{\mathcal{P}}(\mathtt{vk}_g, \mathtt{sk}_p, \mathtt{pk}_p, \sigma_p, \mathtt{ID}_{\mathcal{U}_A}, \mathtt{PS}^e_{AB}) \rightarrow (\mathtt{M}^e_{AB}, \sigma_m, \pi)$ [5] – performed by the proxy $\mathcal{P}$ ($\mathcal{P}_1$ or $\mathcal{P}_2$). This algorithm takes as input the proxies' group public parameters

[5] In this algorithm, we only consider user $\mathcal{U}_A$ with $\mathtt{ID}_{\mathcal{U}_A}$. The same operations are performed for user $\mathcal{U}_B$ with $\mathtt{ID}_{\mathcal{U}_B}$.

$\mathtt{vk}_g$, the pair of keys $(\mathtt{sk}_p, \mathtt{pk}_p)$ of $\mathcal{P}$, the signature σ_p over $\mathcal{P}$'s public key, the identifier $\mathtt{ID}_{\mathcal{U}_A}$ of user $\mathcal{U}_A$ and the message $\mathtt{PS}^e_{AB}$. It returns a signature σ_m over a new message $\mathtt{M}^e_{AB}$ and a group signature represented by a NIWI proof π over the two signatures σ_p and σ_m. The couple $(\mathtt{M}^e_{AB}, \pi)$ is sent to user $\mathcal{U}_A$ to be stored with the contact message $\mathtt{CCM}^e_{AB}$ in his contact list. Note that each input of the contact list is stored for Δ days.

Verification phase

- *Naive Verification*
  - $\mathsf{Sig\_Verify}_{\mathcal{HA}}(\mathtt{vk}_g, \mathtt{M}^e_{AB}, \pi) \rightarrow b$ – performed by $\mathcal{HA}$. Given the public parameters $\mathtt{vk}_g$, a message $\mathtt{M}^e_{AB}$ from the contact list of an infected user, and the corresponding NIWI proof π, the $\mathsf{Sig\_Verify}_{\mathcal{HA}}$ algorithm returns a bit $b \in \{0, 1\}$ stating whether the proof is valid.
  - $\mathsf{CCM\_Verify}_{\mathcal{HA}}(\mathtt{M}^e_{AB}, \mathtt{PS}'^e_{AB}, \mathtt{pk}_{\mathcal{S}}, t_{\mathcal{U}_A}) \rightarrow b$ – run by $\mathcal{HA}$. This algorithm takes as input the message $\mathtt{M}^e_{AB}$, the message $\mathtt{PS}'^e_{AB}$ requested from $\mathcal{S}$, the server's public key $\mathtt{pk}_{\mathcal{S}}$ and the secret value $t_{\mathcal{U}_A}$, and outputs a bit $b \in \{0, 1\}$, i.e., $\mathtt{CCM}^e_{AB}$ is correctly generated or not.
- *Batch Verification*
  - $\mathsf{Batch\_Sig\_Verify}_{\mathcal{HA}}(\mathtt{vk}_g, \{\mathtt{M}^e_{A_iB_i}, \Pi_i\}_{i=1}^N) \rightarrow b$ – performed by $\mathcal{HA}$. Given the public parameters $\mathtt{vk}_g$, a list of N messages $\mathtt{M}^e_{A_iB_i}$ and N corresponding proofs Π_i from the contact lists of multiple users, the $\mathsf{Batch\_Verify}_{\mathsf{HA}}$ algorithm returns a bit $b \in \{0, 1\}$ stating whether the list of proofs is valid or not.
  - $\mathsf{Agg\_Sig\_Verify}_{\mathcal{HA}}(\mathtt{vk}_g, \mathtt{M}^e_{AB}, \Pi) \rightarrow b$ – run by $\mathcal{HA}$ when $\mathsf{Batch\_Sig\_Verif}_{\mathcal{HA}}$ returns 0 over a list or a sub-list of messages. Given the public parameters $\mathtt{vk}_g$, a message $\mathtt{M}^e_{AB}$ and the corresponding proof Π, from an invalid sub-list, the $\mathsf{Agg\_Sig\_Verify}_{\mathcal{HA}}$ algorithm returns a bit $b \in \{0, 1\}$ stating whether the proof is valid or not.
  - $\mathsf{Batch\_CCM\_Verify}_{\mathcal{HA}}(\{\mathtt{M}^e_{AB_i}, \mathtt{PS}'^e_{AB_i}\}_{i=1}^N, \mathtt{pk}_{\mathcal{S}}, t_{\mathcal{U}_A}) \rightarrow b$ – run by $\mathcal{HA}$ for a user $\mathcal{U}_A$. This algorithm takes as input the list messages $\mathtt{M}^e_{AB_i}$ and $\mathtt{PS}'^e_{AB_i}$ (i.e., $i \in \{1, N\}$), the server's public key $\mathtt{pk}_{\mathcal{S}}$ and the secret value $t_{\mathcal{U}_A}$, and outputs a bit $b \in \{0, 1\}$, i.e., the list of messages $\{\mathtt{CCM}^e_{AB_i}\}_{i=1}^N$ is correctly generated or not.

5.3.4 Threat Model

In this section, we first present the adversaries considered in Spot, and then, the formal definitions of the different security and privacy properties.

- **A malicious user ($\mathcal{U}$)**: this adversary attempts to inject false contact messages or contact messages of other users in his contact list. He may also collude with corrupted proxies or malicious users.
- **A honest but curious health authority ($\mathcal{HA}$)**: given a valid group signature, $\mathcal{HA}$ tries to identify the signer (i.e., proxy) of a particular message, hence for identifying the appropriate geographical area and for tracking the user's movements. She may also attempt to link two signatures issued by the same group member. A curious $\mathcal{HA}$ may also try to identify, from a contact list of a particular user, the list of users he had been in contact with.
- **A honest but curious server ($\mathcal{S}$)**: he attempts to link several common contact messages generated by the same user, to trace users' movements.
- **A malicious proxy ($\mathcal{P}$)**: this adversary, either colluding with a malicious user or with n other proxies, attempts to forge the partial signature of the server and to generate a valid signature over a false contact message.

Both malicious users and malicious proxies are considered against security properties, i.e., unforgeability, anti-replay, while the curious health authority and server are considered against privacy requirements, i.e., unlinkability and anonymity. The different adversaries are involved in different phases.
Note that the **anti-replay** property which aims at mitigating the multi-submission of the same contact information is not formally presented below, but is informally discussed in Section 5.6. The following properties are defined w.r.t the corresponding phases and the involved adversaries.

Remark 4. We do not deeply analyze the case of a malicious $\mathcal{GM}$ although our scheme is resistant against this adversary. Indeed, proxies are responsible for generating their key pair and only their public keys are shared with $\mathcal{GM}$. Thus, unless holding a proxy's secret key, $\mathcal{GM}$ is not able to generate a valid signature on behalf of $\mathcal{P}$ thanks to the unforgeability of the signature scheme.

5.3.4.1 Unforgeability

The unforgeability property ensures the security of SPOT for the different phases. It states that a malicious user is not able to forge his contact list (i.e., forging either the group signature or the server's partial signature when colluding with a malicious proxy)[6]. Note that the unforgeability of the group signature scheme used in SPOT will be proven in Section 5.4.5, thus in the security model and analysis, we will only consider the unforgeability of the server's partial signature. Formally, this is defined in a game $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ where an adversary $\mathcal{A}$, playing the role of a corrupted proxy colluding with a malicious user, has access to a $\mathsf{S\_PSign}$ oracle. We note that, for each session i, $\mathcal{A}$ only gets PS^i from the $\mathsf{S\_PSign}$ oracle, while PS'^i is kept secret from the adversary. Then,

[6]We assume that malicious user refers to either a single user or colluding users.

given a valid message $\mathtt{PS}'$ that cannot be obtained by combining either a part of or all messages $\mathtt{PS}^i$, $\mathcal{A}$ succeeds if it outputs a valid message $\mathtt{PS}^*$ to be signed using P_Sign, such that the CCM_Verify verification holds.

Definition 13. Unforgeability – We say that SPOT satisfies the unforgeability property, if for every *PPT* adversary $\mathcal{A}$, there exists a negligible function κ such that: $Pr[\mathbf{Exp}_{\mathcal{A}}^{unforg}(1^\lambda) = 1] \leq \kappa(\lambda)$, where $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ is given below.

$\boldsymbol{Exp}_{\mathcal{A}}^{unforg}(\lambda)$
$pp \leftarrow \mathsf{Set\_params}(\lambda)$
$(\mathtt{sk}_{\mathcal{HA}}, \mathtt{pk}_{\mathcal{HA}}) \leftarrow \mathsf{HA\_keygen}(pp)$
$(\mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}}) \leftarrow \mathsf{S\_keygen}(pp)$
$(\mathtt{sk}_g, \mathtt{vk}_g) \leftarrow \mathsf{Setup\_ProxyGr}(pp)$
$(\mathtt{sk}_p, \mathtt{pk}_p, \sigma_p) \leftarrow \mathsf{Join\_ProxyGr}(pp, \mathtt{sk}_g)$
$\mathtt{ID}_{\mathcal{U}} \leftarrow \mathsf{Set\_UserID}(pp)$
$(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}}) \leftarrow \mathsf{Userkeygen}(pp, \mathtt{ID}_{\mathcal{U}})$
$\mathtt{CCM} \leftarrow \mathsf{Set\_CCM}(\mathtt{D}_1, \mathtt{D}_2)$
$(\mathtt{PS}, \mathtt{PS}') \leftarrow \{\mathsf{S\_PSign}(\mathtt{CCM}, \mathtt{sk}_{\mathcal{S}})\}$
$\mathcal{O} \leftarrow \{\mathsf{S\_PSign}(\cdot, \mathtt{sk}_{\mathcal{S}})\}$
$\mathtt{PS}^* \leftarrow \mathcal{A}^{\mathcal{O}}(\mathtt{vk_g}, \mathtt{sk_p}, \mathtt{pk_p}, \sigma_p, ID_{\mathcal{U}}, pp, \mathtt{PS}')$
letting $\mathtt{CCM}$ *and* $\mathtt{PS}^i$ *denote the queries and answers to and from oracle* S_PSign
$(\mathtt{M}^*, \sigma^*, \pi^*) \leftarrow \mathsf{P\_Sign}(\mathtt{vk}_g, \mathtt{sk}_p, \mathtt{pk}_p, \sigma_p, ID_{\mathcal{U}}, \mathtt{PS}^*)$
If $\mathsf{CCM\_Verify}(\mathtt{M}^*, \mathtt{PS}'^*, \mathtt{pk}_{\mathcal{S}}, t_{\mathcal{U}}) = 1$
return 1
Else return 0

5.3.4.2 Unlinkability

The unlinkability property can be divided into two sub-properties. The first one constitutes the *group-signature unlinkability* stating that a curious health authority is not able to link two or several group signatures issued by the same proxy during the VERIFICATION phase. The second sub-property *multi-CCM unlinkability* ensures that a curious server is not able to link two or several common contact messages to the same user during the GENERATION phase[7].

We note that the *multi-CCM unlinkability* property will be informally discussed in Section 5.6. In this section, we only focus on the *group-signature unlinkability*. Formally, this property is defined in a game $\mathbf{Exp}_{\mathcal{A}}^{unlink}$ where an adversary $\mathcal{A}$ acting as a curious $\mathcal{HA}$ has access to a P_Sign oracle. The adversary may query this oracle on the same

[7]The collusion between the health authority and the server does not pose additional and plausible threats to the different procedures of the whole framework. Indeed, during the GENERATION phase, contact messages are anonymous to the server (and a possible colluding health authority); during the VERIFICATION phase, the health authority knows the true identity of the confirmed cases with their contact information; as such, a collusion between the server and the authority does not bring extra knowledge.

message $\mathtt{PS}^*$ and on a tuple $((\mathtt{sk}_{p_j}, \mathtt{pk}_{p_j}, \sigma_{p_j})$, where $j \in \{0,1\}$ (i.e., the tuple belongs either to proxy $\mathcal{P}_0$ or proxy $\mathcal{P}_1$). A left-or-right oracle LoRSig is initialized with a secret random bit b and returns to $\mathcal{A}$ P_Sign on message $\mathtt{PS}^*$ and respectively on tuples $(\mathtt{sk}_{p_0}, \mathtt{pk}_{p_0}, \sigma_{p_0})$ and $(\mathtt{sk}_{p_b}, \mathtt{pk}_{p_b}, \sigma_{p_b})$. The adversary wins the game if he successfully predicts the bit b (i.e., the guessing probability should be greater than $\frac{1}{2}$).

Definition 14. Unlinkability – We say that SPOT satisfies the unlinkability property, if for every PPT adversary $\mathcal{A}$, there exists a negligible function κ such that: $Pr[\mathbf{Exp}_{\mathcal{A}}^{unlink}(\lambda) = 1] = \frac{1}{2} \pm \kappa(\lambda)$, where $\mathbf{Exp}_{\mathcal{A}}^{unlink}$ is defined below.

$\boldsymbol{Exp}_{\mathcal{A}}^{unlink}(\lambda)$
$pp \leftarrow$ Set_params(λ)
$(\mathtt{sk}_{\mathcal{HA}}, \mathtt{pk}_{\mathcal{HA}}) \leftarrow$ HA_keygen(pp)
$(\mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}}) \leftarrow$ S_keygen(pp)
$(\mathtt{sk}_g, \mathtt{vk}_g) \leftarrow$ Setup_ProxyGr(pp)
$(\mathtt{sk}_{p_i}, \mathtt{pk}_{p_i}, \sigma_{p_i}) \leftarrow$ Join_ProxyGr $(pp, \mathtt{sk}_g)$, $i \in \{0,1\}$
$\mathtt{ID}_{\mathcal{U}} \leftarrow$ Set_UserID(pp)
$(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}}) \leftarrow$ Userkeygen$(pp, \mathtt{ID}_{\mathcal{U}})$
$m^* \leftarrow$ Set_CCM$(\mathtt{D}_1, \mathtt{D}_2)$
$(\mathtt{PS}^*, \mathtt{PS}'^*) \leftarrow$ S_PSign$(m^*, \mathtt{sk_S})$
$b \leftarrow \{0,1\}$
$\mathcal{O} \leftarrow \{P_Sign(\cdot, \mathtt{sk}_{p_j}, \mathtt{pk}_{p_j}, \sigma_{p_j}, \cdot, \cdot)$, LoRSig$(\cdot, \cdot, b)\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathtt{sk}_{\mathcal{HA}}, \mathtt{pk}_{\mathcal{HA}}, \mathtt{vk}_g, \mathtt{pk}_{p_0}, \mathtt{pk}_{p_1}, pp, \mathtt{PS}^*)$
If $b = b'$
  return 1
Else return 0

LoRSig $(\mathtt{vk}_g, ((\mathtt{sk}_{p_0}, \mathtt{pk}_{p_0}, \sigma_{p_0}), (\mathtt{sk}_{p_1}, \mathtt{pk}_{p_1}, \sigma_{p_1})), \mathtt{ID}_{\mathcal{U}}, \mathtt{PS}^*, b)$

$(\mathtt{M}^*, \sigma^*, \pi^*) \leftarrow$ P_Sign$(\mathtt{vk}_g, \mathtt{sk}_{p_0}, \mathtt{pk}_{p_0}, \sigma_{p_0}, \mathtt{ID}_{\mathcal{U}}, \mathtt{PS}^*)$
$(\mathtt{M}^*, \sigma_b^*, \pi_b^*) \leftarrow$ P_Sign$(\mathtt{vk}_g, \mathtt{sk}_{p_b}, \mathtt{pk}_{p_b}, \sigma_{p_b}, \mathtt{ID}_{\mathcal{U}}, \mathtt{PS}^*)$
return $((\mathtt{M}^*, \pi^*), (\mathtt{M}^*, \pi_b^*))$

5.3.4.3 Anonymity

The anonymity property guarantees that no entity is able to identify users involved in a particular contact list (i.e., the owner and the contacted users), during the VERIFICATION phase, and is described through the game $\mathbf{Exp}_{\mathcal{A}}^{anon}$. The anonymity property implies that even if $\mathcal{HA}$ knows that a contact list belongs to a user ($\mathcal{U}$), $\mathcal{HA}$ is not able to identify users being in contact with $\mathcal{U}$ [8]. This should hold even if an efficient adversary, playing the role of the curious health authority, is given access to Set_CCM, S_PSign, P_Sign oracles. $\mathcal{A}$ can learn contact messages and signatures associated to the selected users' identifiers. $\mathcal{A}$ also gets access to a left-or-right oracle LoRCU which is initialized with a secret random bit $b \in \{0,1\}$. $\mathcal{A}$ may query this oracle on $\mathtt{ID}_{\mathcal{U}_0}$ and $\mathtt{ID}_{\mathcal{U}_1}$ referred to as the identifiers of respectively user $\mathcal{U}_0$ and user $\mathcal{U}_1$. Observe that user $\mathcal{U}_A$ is involved in all queries. $\mathtt{D}^*_{\mathcal{U}_A}$ and $\mathtt{D}^*_{\mathcal{U}_b}$, respectively belonging to user $\mathcal{U}_A$ and user $\mathcal{U}_b$, are randomly

[8] We assume that the probability of two confirmed users being in contact and submitting their respective contact lists to $\mathcal{HA}$ at the same period, is low.

selected in order to execute the LoRCU oracle. To win the proposed anonymity game, the adversary should predict the bit b (i.e., which one of users $\mathcal{U}_0$ and $\mathcal{U}_1$ is involved in the contact with user $\mathcal{U}_A$) with a probability greater than $\frac{1}{2}$.

Definition 15. Anonymity – We say that SPOT fulfills the anonymity requirement, if for every PPT adversary $\mathcal{A}$, there exists a negligible function κ such that: $Pr[\mathbf{Exp}_{\mathcal{A}}^{anon}(1^{\lambda}) = 1] = \frac{1}{2} \pm \kappa(\lambda)$, where $\mathbf{Exp}_{\mathcal{A}}^{anon}$ is defined as follows.

$\boldsymbol{Exp}_{\mathcal{A}}^{anon}(\lambda)$
$pp \leftarrow \mathsf{Set\_params}(\lambda)$
$(\mathtt{sk}_{\mathcal{HA}}, \mathtt{pk}_{\mathcal{HA}}) \leftarrow \mathsf{HA\_keygen}(pp)$
$(\mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}}) \leftarrow \mathsf{S\_keygen}(pp)$
$(\mathtt{sk}_g, \mathtt{vk}_g) \leftarrow \mathsf{Setup\_ProxyGr}(pp)$
$(\mathtt{sk}_p, \mathtt{pk}_p, \sigma_p) \leftarrow \mathsf{Join\_ProxyGr}(pp, \mathtt{sk}_g)$
$\mathtt{ID}_{\mathcal{U}_A} \leftarrow \mathsf{Set\_UserID}(pp)$
$(\mathtt{sk}_{\mathcal{U}_A}, \mathtt{pk}_{\mathcal{U}_A}) \leftarrow \mathsf{Userkeygen}(pp, \mathtt{ID}_{\mathcal{U}_A})$
$\mathtt{ID}_{\mathcal{U}_i} \leftarrow \mathsf{Set\_UserID}(pp)$, $i \in \{0..N\}$
$(\mathtt{sk}_{\mathcal{U}_i}, \mathtt{pk}_{\mathcal{U}_i}) \leftarrow \mathsf{Userkeygen}(pp, \mathtt{ID}_{\mathcal{U}_i})$, $i \in \{0..N\}$
$b \leftarrow \{0, 1\}$
$\mathcal{O} \leftarrow \{\mathsf{Set\_CCM}(\cdot,\cdot), \mathsf{S\_PSign}(\cdot,\mathtt{sk}_{\mathcal{S}}), \mathsf{P\_Sign}(\cdot,\mathtt{sk}_{\mathtt{p}}, \cdot,\sigma_p,\cdot,\cdot), \mathsf{LoRCU}(\cdot,\cdot,b,\cdot)$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathtt{sk}_{\mathcal{HA}}, \mathtt{pk}_{\mathcal{HA}}, pp, \mathtt{ID}_{\mathcal{U}_A}, \{\mathtt{ID}_{\mathcal{U}_i}\}_{i=0}^{N})$
If $b = b'$
  return 1
Else return 0

$\mathsf{LoRCU}(\mathtt{D}^*_{\mathcal{U}_A}, \mathtt{D}^*_{\mathcal{U}_b}, b, \mathtt{vk}_g, \mathtt{sk}_{\mathcal{S}}, \mathtt{sk}_p, \mathtt{pk}_p, \sigma_p, \mathtt{ID}_{\mathcal{U}_A}, \mathtt{ID}_{\mathcal{U}_b})$

$\mathsf{CCM}^*_b \leftarrow \mathsf{Set\_CCM}_{\mathcal{U}_A}(\mathtt{D}^*_{\mathcal{U}_A}, \mathtt{D}^*_{\mathcal{U}_b})$
$(\mathsf{PS}^*_b, \mathsf{PS}'^*_b) \leftarrow \mathsf{S\_PSign}(\mathsf{CCM}^*_b, \mathtt{sk}_{\mathcal{S}})$
$(\mathtt{M}^*_b, \sigma^*_b, \pi^*_b) \leftarrow \mathsf{P\_Sign}(\mathtt{vk}_g, \mathtt{sk}_p, \mathtt{pk}_p, \sigma_p, \mathtt{ID}_{\mathcal{U}_A}, \mathsf{PS}^*_b)$
return $(\mathsf{CCM}^*_b, \mathtt{M}^*_b, \pi^*_b)$

5.4 Building Blocks

This section first presents structure-preserving signatures [2] with their different variants as main building blocks of the SPOT protocol. Sections 5.4.1 and 5.4.2 describe respectively constant-size signatures and signatures on mixed-group messages that are instantiated with Non-Interactive Witness Indistinguishable (NIWI) proofs (cf. Section 5.4.3), in Section 5.4.4, to build a group signature scheme on group element messages. Section 5.4.5 introduces a new PoK-based group signature scheme that offers batch verification over multiple NIWI proofs.

5.4.1 Structure-preserving Constant-size Signature

Structure-preserving constant-size signature was defined by Abe *et al.* [2] as the main scheme of structure-preserving signatures used to sign a message $\vec{m} = (m_1, ..., m_k) \in {\mathbb{G}_2}^k$, considering an asymmetric bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, g_1, g_2, e)$. A constant-size signature scheme CSIG [2] relies on the following three PPT algorithms (CSIG.Key, CSIG.Sign,

CSIG.Verify):

CSIG.Key(1^λ): This algorithm takes as input the security parameter (1^λ) and outputs the pair of public and secret keys ($\mathtt{sk}, \mathtt{pk}$) of the signer. It chooses two random generators $g_r, h_u \leftarrow \mathbb{G}_1^*$ and random values $\gamma_i, \delta_i \leftarrow \mathbb{Z}_q^*$ and computes $g_i = {g_r}^{\gamma_i}$ and $h_i = {h_u}^{\delta_i}$, for $i = 1, ..., k$. It then selects $\gamma_z, \delta_z \leftarrow \mathbb{Z}_q^*$ and computes $g_z = {g_r}^{\gamma_z}$ and $h_z = {h_u}^{\delta_z}$. It also chooses $\alpha, \beta \leftarrow \mathbb{Z}_q^*$ and sets the couples (g_r, g_2^α) and (h_u, g_2^β). The public key is set as $\mathtt{pk} = (g_z, h_z, g_r, h_u, g_2^\alpha, g_2^\beta, \{g_i, h_i\}_{i=1}^k)$ and the secret key is set as $\mathtt{sk} = (\mathtt{pk}, \alpha, \beta, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^k)$.

CSIG.Sign($\mathtt{sk}, \vec{m}$): This algorithm generates a signature σ over a message $\vec{m}$ using the secret key $\mathtt{sk}$. That is, the signer randomly selects $\zeta, \rho, \tau, \varphi, \omega \leftarrow \mathbb{Z}_q^*$ and computes

$$z = g_2^\zeta, r = {g_2}^{\alpha - \rho\tau - \gamma_z\zeta} \prod_{i=1}^k {m_i}^{-\gamma_i}, s = {g_r}^\rho, t = {g_2}^\tau,$$

$$u = {g_2}^{\beta - \varphi\omega - \delta_z\zeta} \prod_{i=1}^k {m_i}^{-\delta_i}, v = {h_u}^\varphi, w = {g_2}^\omega$$

The signature is set as $\sigma = (z, r, s, t, u, v, w)$.

CSIG.Verify($\mathtt{pk}, \vec{m}, \sigma$): This algorithm checks the validity of the signature σ on the message m relying on the signer's public key $\mathtt{pk}$. It outputs 1 if the signature is valid and 0 otherwise. The verifier checks if the following equations hold:

$$A = e(g_z, z)e(g_r, r)e(s, t)\prod_{i=1}^k e(g_i, m_i) \tag{5.1}$$

$$B = e(h_z, z)e(h_u, u)e(v, w)\prod_{i=1}^k e(h_i, m_i) \tag{5.2}$$

where $A = e(g_r, g_2^\alpha)$ and $B = e(h_u, g_2^\beta)$

5.4.2 Structure-preserving Signature on Mixed-group Messages

A structure-preserving signature on mixed-group messages XSIG [2] represents a signature scheme where the message space is a mixture of the two groups $\mathbb{G}_1$ and $\mathbb{G}_2$. We consider two constant-size signature schemes CSIG1 and CSIG2. CSIG2 is the same scheme as in Section 5.4.1 where the message space is ${\mathbb{G}_2}^{k_2}$, while CSIG1 is a 'dual' scheme obtained by exchanging $\mathbb{G}_1$ and $\mathbb{G}_2$ in the same scheme, where the message space is ${\mathbb{G}_1}^{k_1}$. The message space for the XSIG is then ${\mathbb{G}_1}^{k_1} \times {\mathbb{G}_2}^{k_2}$. Let $(\vec{m}, \vec{\tilde{m}})$ be a message in ${\mathbb{G}_1}^{k_1} \times {\mathbb{G}_2}^{k_2}$. For a vector $\vec{\tilde{m}} \in {\mathbb{G}_1}^{k_1}$ and a single element $s \in \mathbb{G}_1$, let $\vec{m}||s$ denote a vector in ${\mathbb{G}_1}^{k_1+1}$ obtained by appending s to the end of $\vec{m}$.

A mixed-group messages signature scheme XSIG relies on the following three PPT

algorithms (XSIG.Key, XSIG.Sign, XSIG.Verify):

XSIG.Key(1^λ): This algorithm runs $(\mathtt{sk}_1, \mathtt{pk}_1) \leftarrow$ CSIG1.Key(1^λ) and $(\mathtt{sk}_2, \mathtt{pk}_2) \leftarrow$ CSIG2.Key(1^λ) and sets $(\mathtt{sk}, \mathtt{pk}) = ((\mathtt{sk}_1, \mathtt{sk}_2), (\mathtt{pk}_1, \mathtt{pk}_2))$.

XSIG.Sign($\mathtt{sk}, (\vec{m}, \vec{\tilde{m}})$): This algorithm runs $\sigma_2 = (z, r, s, t, u, v, w) \leftarrow$ CSIG2.Sign($\mathtt{sk}_2$, $\vec{\tilde{m}}$) and $\sigma_1 = (z', r', s', t', u', v', w') \leftarrow$ CSIG1.Sign($\mathtt{sk}_1, \vec{m}||s$), and outputs $\sigma = (\sigma_1, \sigma_2)$.

XSIG.Verify($\mathtt{pk}, (\vec{m}, \vec{\tilde{m}}), (\sigma_1, \sigma_2)$): This algorithm takes $s \in \mathbb{G}_1$ from σ_2, runs $b_2 =$ CSIG2.Verify($\mathtt{pk}_2, \vec{\tilde{m}}, \sigma_2$) and $b_1 =$ CSIG1.Verify($\mathtt{pk}_1, \vec{m}||s, \sigma_1$). If $b_1 = b_2 = 1$, the algorithm outputs 1, otherwise it outputs 0.

5.4.3 Non-Interactive Witness Indistinguishable Proof

In this section, we present the Groth-Sahai NIWI proof scheme applied on pairing product equations with an asymmetric bilinear map. Witness-indistinguishability implies that the verifier of a group signature is not able to find the group member that has generated the signature. The NIWI scheme we consider, involves four PPT algorithms (NIWI.Setup, NIWI.CRS, NIWI.Proof, NIWI.Verify):

NIWI.Setup: This algorithm outputs a setup $(\mathtt{gk}, \mathtt{sk})$ such that $\mathtt{gk} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, g_1, g_2, e)$ and $\mathtt{sk} = (p, n)$ where $q = pn$.

NIWI.CRS: This algorithm generates a common reference string CRS. It takes $(\mathtt{gk}, \mathtt{sk})$ as inputs and produces $\mathtt{CRS} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, \iota_1, p_1, \iota_2, p_2, \iota_3, p_3, \mathcal{U}, \mathcal{V})$, where $\mathcal{U} = rg_1$, $\mathcal{V} = sg_2$; $r, s \in \mathbb{Z}_q^*$ and

$$\begin{array}{llllll} \iota_1\colon & \mathbb{G}_1 \longrightarrow \mathbb{G}_1 & \iota_2\colon & \mathbb{G}_2 \longrightarrow \mathbb{G}_2 & \iota_3\colon & \mathbb{G}_3 \longrightarrow \mathbb{G}_3 \\ & x \longmapsto x & & y \longmapsto y & & z \longmapsto z \end{array}$$

$$\begin{array}{llllll} p_1\colon & \mathbb{G}_1 \longrightarrow \mathbb{G}_1 & p_2\colon & \mathbb{G}_2 \longrightarrow \mathbb{G}_2 & p_3\colon & \mathbb{G}_3 \longrightarrow \mathbb{G}_3 \\ & x \longmapsto \lambda x & & y \longmapsto \lambda y & & z \longmapsto z^\lambda \end{array}$$

NIWI.Proof: This algorithm generates a NIWI proof for satisfiability of a set of pairing product equations of the form of

$$\prod_{i=1}^{l} e(\mathcal{A}_i, \mathcal{Y}_i) \prod_{i=1}^{k} e(\mathcal{X}_i, \mathcal{B}_i) \prod_{i=1}^{k} \prod_{j=1}^{l} e(\mathcal{X}_i, \mathcal{Y}_j)^{\gamma_{ij}} = t$$

also written as

$$(\vec{\mathcal{A}} \cdot \vec{\mathcal{Y}})(\vec{\mathcal{X}} \cdot \vec{\mathcal{B}})(\vec{\mathcal{X}} \cdot \Gamma \vec{\mathcal{Y}}) = t$$

It takes as input gk, CRS and a list of pairing product equations $\{(\vec{\mathcal{A}}_i, \vec{\mathcal{B}}_i, \Gamma_i, t_i)\}_{i=1}^{N}$ and a satisfying witness $\vec{\mathcal{X}} \in \mathbb{G}_1^k$, $\vec{\mathcal{Y}} \in \mathbb{G}_2^l$. To generate a proof over a pairing product

equation, the algorithm, first, picks at random $\mathcal{R} \leftarrow Vec_k(\mathbb{Z}_q)$ and $\mathcal{S} \leftarrow Vec_l(\mathbb{Z}_q)$, commits to all variables as $\vec{\mathcal{C}} := \vec{\mathcal{X}} + \mathcal{R}\mathcal{U}$ and $\vec{\mathcal{D}} := \vec{\mathcal{Y}} + \mathcal{S}\mathcal{V}$, and computes

$$\pi = \mathcal{R}^\top \iota_2(\vec{\mathcal{B}}) + \mathcal{R}^\top \Gamma \iota_2(\vec{\mathcal{Y}}) + \mathcal{R}^\top \Gamma \mathcal{S}\mathcal{V}$$

$$\theta = \mathcal{S}^\top \iota_1(\vec{\mathcal{A}}) + \mathcal{S}^\top \Gamma^\top \iota_1(\vec{\mathcal{X}})$$

The algorithm outputs the proof (π, θ).

NIWI.Verify: This algorithm checks if the proof is valid. It takes as inputs and for each equation, gk, CRS, $\{(\vec{\mathcal{A}}_i, \vec{\mathcal{B}}_i, \Gamma_i, t_i)\}_{i=1}^N$ and $(\vec{\mathcal{C}}_i, \vec{\mathcal{D}}_i, \{(\pi_i, \theta_i)\}_{i=1}^N)$. It checks the following equation:

$$e(\iota_1(\vec{\mathcal{A}}_i), \vec{\mathcal{D}}_i) e(\vec{\mathcal{C}}_i, \iota_2(\vec{\mathcal{B}}_i)) e(\vec{\mathcal{C}}_i, \Gamma_i \vec{\mathcal{D}}_i) = \iota_3(t_i) e(\mathcal{U}, \pi_i) e(\theta_i, \mathcal{V}) \tag{5.3}$$

The algorithm outputs 1 if the equation holds, else it outputs 0.

Correctness of Equation 5.3

$$\begin{aligned}
& e(\iota_1(\vec{\mathcal{A}}_i), \vec{\mathcal{D}}_i) e(\vec{\mathcal{C}}_i, \iota_2(\vec{\mathcal{B}}_i)) e(\vec{\mathcal{C}}_i, \Gamma_i \vec{\mathcal{D}}_i) \\
= \; & e(\iota_1(\vec{\mathcal{A}}_i), \vec{\mathcal{Y}}_i + \mathcal{S}\mathcal{V}) e(\vec{\mathcal{X}}_i + \mathcal{R}\mathcal{U}, \iota_2(\vec{\mathcal{B}}_i)) e(\vec{\mathcal{X}}_i + \mathcal{R}\mathcal{U}, \Gamma_i(\vec{\mathcal{Y}}_i + \mathcal{S}\mathcal{V})) \\
= \; & e(\iota_1(\vec{\mathcal{A}}_i), \vec{\mathcal{Y}}_i) e(\iota_1(\vec{\mathcal{A}}_i), \mathcal{S}\mathcal{V}) e(\vec{\mathcal{X}}_i, \iota_2(\vec{\mathcal{B}}_i)) e(\mathcal{R}\mathcal{U}, \iota_2(\vec{\mathcal{B}}_i)) e(\vec{\mathcal{X}}_i, \Gamma_i \vec{\mathcal{Y}}_i) e(\vec{\mathcal{X}}_i, \Gamma_i \mathcal{S}\mathcal{V}) \\
& e(\mathcal{R}\mathcal{U}, \Gamma_i \vec{\mathcal{Y}}_i) e(\mathcal{R}\mathcal{U}, \Gamma_i \mathcal{S}\mathcal{V}) \\
= \; & e(\iota_1(\vec{\mathcal{A}}_i), \vec{\mathcal{Y}}_i) e(\vec{\mathcal{X}}_i, \iota_2(\vec{\mathcal{B}}_i)) e(\vec{\mathcal{X}}_i, \Gamma_i \vec{\mathcal{Y}}_i) e(\mathcal{S}^\top \iota_1(\vec{\mathcal{A}}_i), \mathcal{V}) e(\mathcal{U}, \mathcal{R}^\top \iota_2(\vec{\mathcal{B}}_i)) e(\Gamma_i^\top \mathcal{S}^\top \vec{\mathcal{X}}_i, \mathcal{V}) \\
& e(\mathcal{U}, \mathcal{R}^\top \Gamma_i \vec{\mathcal{Y}}_i) e(\mathcal{U}, \mathcal{R}^\top \Gamma_i \mathcal{S}\mathcal{V}) \\
= \; & \iota_3(t_i) e(\mathcal{U}, \mathcal{R}^\top \iota_2(\vec{\mathcal{B}}_i) + \mathcal{R}^\top \Gamma_i \vec{\mathcal{Y}}_i + \mathcal{R}^\top \Gamma_i \mathcal{S}\mathcal{V}) e(\mathcal{S}^\top \iota_1(\vec{\mathcal{A}}_i) + \Gamma_i^\top \mathcal{S}^\top \vec{\mathcal{X}}_i, \mathcal{V}) \\
= \; & \iota_3(t_i) e(\mathcal{U}, \pi_i) e(\theta_i, \mathcal{V})
\end{aligned}$$

Computational Witness-Indistinguishability of NIWI Proofs The Computational Witness-Indistinguishability property is defined as follows: Let $L \in \mathcal{NP}$ be a language and let $(\mathcal{P}, \mathcal{V})$ be an interactive proof system for L. We say that $(\mathcal{P}, \mathcal{V})$ is *witness-indistinguishable* (WI) if for every *PPT* algorithm $\mathcal{V}^*$ and every two sequences $\{w_x^1\}_{x \in L}$ and $\{w_x^2\}_{x \in L}$ such that w_x^1 and w_x^2 are both witnesses for x, the following ensembles are computationally indistinguishable, where z is an auxiliary input to $\mathcal{V}^*$:

1. $\{\langle \mathcal{P}(w_x^1), \mathcal{V}^* \rangle(x)\}_{x \in L, z \in \{0,1\}^*}$
2. $\{\langle \mathcal{P}(w_x^2), \mathcal{V}^* \rangle(x)\}_{x \in L, z \in \{0,1\}^*}$

5.4.4 Group Signatures Drawn from Structure-preserving Signatures

We present hereafter an instantiation of a group signature scheme that allows to sign a group element message relying on a constant-size signature scheme CSIG, a mixed-group

messages signature scheme XSIG and a witness indistinguishable proof of knowledge system NIWI [64] (cf. Section 5.4.3).

A group signature scheme GSIG relies on the four following algorithms (GSIG.Setup, GSIG.Join, GSIG.Sign, GSIG.Verify):

GSIG.Setup : represents the setup algorithm. It runs XSIG.Key algorithm that generates the key pair $(\mathtt{sk}_g, \mathtt{pk}_g)$ of the group manager and sets up a CRS Σ_{NIWI} for the NIWI proof. The group verification key is set as $\mathtt{vk}_g = (\mathtt{pk}_g, \Sigma_{\mathsf{NIWI}})$, while the certification secret key $\mathtt{sk}_g$ is privately stored by the group manager.

GSIG.Join: represents the join algorithm. It is composed of two steps. In the first one, the group member generates his key-pair $(\mathtt{sk}_p, \mathtt{pk}_p)$ while running the CSIG.Key algorithm. Only the public key $\mathtt{pk}_p$ is sent to the group manager. This latter generates a signature σ_p over $\mathtt{pk}_p$, using the XSIG.Sign algorithm, and sends it to the group member.

GSIG.Sign: represents the signing algorithm run by a group member on a message $m \in \mathbb{G}_2$. The group member generates, over the message m, a signature $\sigma_m \leftarrow \mathsf{CSIG.Sign}(\mathtt{sk}_p, m)$ and a non-interactive witness indistinguishable proof of knowledge $\pi \leftarrow \mathsf{NIWI.Proof}(\Sigma_{\mathsf{NIWI}}, pub, wit)$ that proves $1 = \mathsf{XSIG.Verify}(\mathtt{pk}_g, \mathtt{pk}_p, \sigma_p)$ and $1 = \mathsf{CSIG.Verify}(\mathtt{pk}_p, m, \sigma_m)$ with respect to the witness $wit = (\mathtt{pk}_p, \sigma_p, \sigma_m)$ and the public information $pub = (\mathtt{pk}_g, m)$. The signing algorithm outputs the group signature π.

GSIG.Verify: represents the group signature verification algorithm run by a verifier. It takes $(\mathtt{vk}_g, m, \pi)$ as input and verifies the correctness of the NIWI proof π w.r.t. $pub = (\mathtt{pk}_g, m)$ and the CRS Σ_{NIWI}.

5.4.5 A Group Signature Scheme with Batch Verification over Multiple Proofs of Knowledge

This section introduces a novel group signature, drawn from structure-preserving signatures, that offers batch verification over multiple Groth-Sahai Non-Interactive Witness-Indistinguishable proofs [64].

The proposed scheme includes five main algorithms, referred to as Setup, Join, Sign, Batch_Verify and Agg_Verify defined as follows.

- $\mathsf{Setup}() \rightarrow (\mathtt{sk}_{\mathcal{G}}, \mathtt{vk}_{\mathcal{G}})$ – run by a group manager $\mathcal{G}$ in order to set up the group signature parameters. It returns the signers' group verification key $\mathtt{vk}_{\mathcal{G}}$ encompassing $\mathtt{pk}_{\mathcal{G}}$ the public key of the group manager and Σ_{NIWI} of a Non-Interactive Witness-Indistinguishable (NIWI) proof [64] associated with the public key. The Setup algorithm also outputs the secret key $\mathtt{sk}_{\mathcal{G}}$ of $\mathcal{G}$.

- $\mathsf{Join}(\mathtt{sk}_{\mathcal{G}}) \rightarrow (\mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}}, \sigma_k)$ – performed through an interactive session between a signer $\mathcal{S}$ and $\mathcal{G}$. It takes as input the secret key $\mathtt{sk}_{\mathcal{G}}$ of the group manager, and outputs the pair of keys $(\mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}})$ of the group member (i.e., signer) $\mathcal{S}$, and a signature σ_k over $\mathcal{S}$'s public key $\mathtt{pk}_{\mathcal{S}}$. Indeed, $\mathcal{S}$ is responsible for generating his

pair of keys, while $\mathcal{G}$ is in charge of the signature σ_k generation aiming to certify $\mathcal{S}$'s keys.

- $\mathsf{Sign}(\mathtt{vk}_{\mathcal{G}}, \mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}}, \sigma_k, m) \rightarrow (\sigma_m, \Pi)$ – run by the signer as a group member. This algorithm takes as input the group public parameters $\mathtt{vk}_{\mathcal{G}}$, the pair of keys $(\mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}})$ of $\mathcal{S}$, the signature σ_k over $\mathcal{S}$'s public key and the message m. The Sign returns a signature σ_m over the message m and a NIWI proof Π over the two signatures σ_k and σ_m. The proof Π is locally stored with the message m.

- $\mathsf{Batch\_Verify}(\mathtt{vk}_{\mathcal{G}}, \{m_i, \Pi_i\}_{i=1}^{N}) \rightarrow b$ – performed by a verifier $\mathcal{V}$. Given the public parameters $\mathtt{vk}_{\mathcal{G}}$, a list of N messages m_i and N corresponding proofs Π_i sent by multiple signers (i.e., a signer can send to $\mathcal{V}$ more than one message), the Batch_Verify algorithm returns a bit $b \in \{0, 1\}$ stating whether the list of proofs is valid or not.

- $\mathsf{Agg\_Verify}(\mathtt{vk}_{\mathcal{G}}, m, \Pi) \rightarrow b$ – run by $\mathcal{V}$ when $\mathsf{Batch\_Verify}_{\mathcal{V}}$ returns 0 over a list or a sub-list of messages. Given the public parameters $\mathtt{vk}_{\mathcal{G}}$, a message m and the corresponding proof Π, from an invalid sub-list, the Agg_Verify algorithm returns a bit $b \in \{0, 1\}$ stating whether proof is valid or not.

Security Proof of the proposed *PoK*-based Group Signature Scheme The security of the proposed scheme refers to the inability of a malicious outsider to generate a valid group signature. That is, the group signature scheme is unforgeable.

Let us consider an adversary $\mathcal{A}$ that is allowed to query, as many times as he wants, the Sign algorithm on a message m_i. Then, during the challenge phase, $\mathcal{A}$ is asked to produce a valid message signature pair (m^*, Π^*) such that the message m^* was not queried before.

For this purpose, we consider two adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ respectively against the unforgeability of the group signature scheme GSIG (cf. Chapter 2, Section 2.3.1.1) and the soundness[9] of the proof system NIWI. The advantage of $\mathcal{A}$ to break the unforgeability of the proposed group signature scheme is expressed as follows:

$$\mathbf{Adv}^{unfor}_{GS,\mathcal{A}}(1^\lambda) \leq \mathbf{Adv}^{unfor}_{\mathsf{GSIG},\mathcal{B}_1}(1^\lambda) + \mathbf{Adv}^{sound}_{\mathsf{NIWI},\mathcal{B}_2}(1^\lambda)$$

According to [14], the unforgeability property can be directly inherited from the traceability property stating that it is not possible to generate signatures without tracing its originator. As the group signature scheme GSIG, relying on the construction of Bellare *et al.* [14], is proven to be traceable, then the unforgeability property of GSIG is also satisfied. Thus, the $\mathbf{Adv}^{unfor}_{\mathsf{GSIG},\mathcal{B}_1}(1^\lambda)$ function is negligible. The advantage function $\mathbf{Adv}^{sound}_{\mathsf{NIWI},\mathcal{B}_2}(1^\lambda)$ can be expressed as follows:

$$\mathbf{Adv}^{sound}_{\mathsf{NIWI},\mathcal{B}_2}(1^\lambda) = Pr[\mathcal{B}_2 \quad outputs \quad (m, \Pi) : \mathsf{NIWI.Verify}(m, \Pi) = 1]$$

[9]The soundness property ensures that is it not possible to prove a false statement.

Referring to [14], $Pr[\mathcal{B}_2 \quad outputs \quad (m, \Pi) : \mathsf{NIWI.Verify}(m, \Pi) = 1] \leq 2^{-\lambda}$ as the NIWI proof system is sound. As such, the advantage function $\mathbf{Adv}^{sound}_{\mathsf{NIWI},\mathcal{B}_2}(1^\lambda)$ is negligible. Thus, the advantage function $\mathbf{Adv}^{unfor}_{\mathcal{A}}(1^\lambda)$ is also negligible proving that the proposed group signature scheme is secure, i.e., satisfies the unforgeability property.

5.5 SPOT Algorithms

This section gives a concrete construction of the different phases and algorithms of SPOT, introduced in Section 5.3.1. SPOT relies on the different variants of structure-preserving signatures represented in Section 5.4.

5.5.1 Sys_Init Phase

- Set_params – a trusted authority sets an asymmetric bilinear group (q, $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$, g_1, g_2, e) relying on the security parameter λ, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are two cyclic groups of prime order q, g_1 and g_2 are generators of respectively $\mathbb{G}_1$ and $\mathbb{G}_2$ and e is a bilinear map such that $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$. The trusted authority also considers a cryptographic hash function $\mathbf{H} : \{0,1\}^* \rightarrow \mathbb{Z}_q$. The output of the Set_params algorithm represents the system global parameters that are known by all the system entities. The tuple $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, g_1, g_2, e, \mathbf{H})$ is denoted by pp, and is considered as a default input of all algorithms.

- HA_keygen – a trusted authority takes as input the public parameters pp, selects a random $x \in \mathbb{Z}_q^*$ and generates the pair of secret and public keys $(\mathtt{sk}_{\mathcal{HA}}, \mathtt{pk}_{\mathcal{HA}})$ of the health authority as follows:
$$\mathtt{sk}_{\mathcal{HA}} = x \quad ; \quad \mathtt{pk}_{\mathcal{HA}} = g_2^x$$

- S_keygen – a trusted authority generates the pair of secret and public keys $(\mathtt{sk}_{\mathcal{S}}, \mathtt{pk}_{\mathcal{S}})$ of the server as given below, relying on the system public parameters pp and two selected randoms $y_1, y_2 \in \mathbb{Z}_q^*$.
$$\mathtt{sk}_{\mathcal{S}} = (y_1, y_2) \quad ; \quad \mathtt{pk}_{\mathcal{S}} = (Y_1, Y_2) = (g_2^{y_1}, g_2^{y_2})$$

- Setup_ProxyGr$_{\mathcal{GM}}$ – $\mathcal{GM}$ sets up the group of proxies by generating a group public key $\mathtt{vk}_g$ and a certification secret key $\mathtt{sk}_g$ as shown in Algorithm 12. Note that the Setup_ProxyGr$_{\mathcal{GM}}$ algorithm is the same as the Setup algorithm of the proposed group signature scheme (cf. Section 5.4.5).

- Join_ProxyGr$_{\mathcal{P}/\mathcal{GM}}$ – $\mathcal{P}$ first generates his pair of keys $(\mathtt{sk}_p, \mathtt{pk}_p)$ w.r.t. the CSIG.Key algorithm (cf. Section 5.4.1). Afterwards, $\mathcal{GM}$ generates a signature σ_p over the public key $\mathtt{pk}_p$ w.r.t. the XSIG.Sign algorithm (cf. Section 5.4.2). The Join_ProxyGr$_{\mathcal{P}/\mathcal{GM}}$ algorithm is detailed in Algorithm 13. It is equivalent to the Join algorithm of the proposed group signature (cf. Section 5.4.5).

- Set_UserID$_{\mathcal{HA}}$ – every time, a user ($\mathcal{U}$) installs the application and wants to register, $\mathcal{HA}$ picks a secret $t_{\mathcal{U}} \in \mathbb{Z}_q^*$ and sets the user's identifier $\mathtt{ID}_{\mathcal{U}}$ as
$$\mathtt{ID}_{\mathcal{U}} = h_{\mathcal{U}} = g_2^{t_{\mathcal{U}}}$$

Algorithm 12 Setup_ProxyGr$_{\mathcal{GM}}$ algorithm

Input: the system public parameters pp
Output: the public parameters $\mathtt{vk}_g$ of the proxies' group and the secret key $\mathtt{sk}_g$
// **The next iterations are executed to generate the pair of keys of** $\mathcal{GM}$
pick at random $g_{r1}, h_{u1} \leftarrow \mathbb{G}_1^*$, $g_{r2}, h_{u2} \leftarrow \mathbb{G}_2^*$
for $i = 1$ to 2 **do**
    pick at random $\gamma_{1i}, \delta_{1i} \leftarrow \mathbb{Z}_q^*$
    compute $g_{1i} \leftarrow {g_{r1}}^{\gamma_{1i}}$, $h_{1i} \leftarrow {h_{u1}}^{\delta_{1i}}$
end for
for $j = 1$ to 7 **do**
    pick at random $\gamma_{2j}, \delta_{2j} \leftarrow \mathbb{Z}_q^*$
    compute $g_{2i} \leftarrow {g_{r2}}^{\gamma_{2j}}$ and $h_{2j} \leftarrow {h_{u2}}^{\delta_{2j}}$
end for
pick at random $\gamma_{1z}, \delta_{1z}, \gamma_{2z}, \delta_{2z} \leftarrow \mathbb{Z}_q^*$;
compute $g_{1z} \leftarrow {g_{r1}}^{\gamma_{1z}}$, $h_{1z} \leftarrow {h_{u1}}^{\delta_{1z}}$, $g_{2z} \leftarrow {g_{r2}}^{\gamma_{2z}}$ and $h_{2z} \leftarrow {h_{u2}}^{\delta_{2z}}$;
pick at random $\alpha_1, \alpha_2, \beta_1, \beta_2 \leftarrow \mathbb{Z}_q^*$;
$\mathtt{pk}_1 \leftarrow (g_{2z}, h_{2z}, g_{2r}, h_{2u}, g_1^{\alpha_2}, g_1^{\beta_2}, \{g_{2j}, h_{2j}\}_{j=1}^{7})$ and $\mathtt{sk}_1 \leftarrow (\mathtt{pk}_1, \alpha_2, \beta_2, \gamma_{2z}, \delta_{2z}, \{\gamma_{2j}, \delta_{2j}\}_{j=1}^{7})$;
$\mathtt{pk}_2 \leftarrow (g_{1z}, h_{1z}, g_{1r}, h_{1u}, g_2^{\alpha_1}, g_2^{\beta_1}, \{g_{1i}, h_{1i}\}_{i=1}^{2})$ and $\mathtt{sk}_2 \leftarrow (\mathtt{pk}_2, \alpha_1, \beta_1, \gamma_{1z}, \delta_{1z}, \{\gamma_{1i}, \delta_{1i}\}_{i=1}^{2})$;
set $\mathtt{pk}_g \leftarrow (\mathtt{pk}_1, \mathtt{pk}_2)$ and $\mathtt{sk}_g \leftarrow (\mathtt{sk}_1, \mathtt{sk}_2)$;
// **The next iterations are executed to generate the CRS** Σ_{NIWI}
pick at random $r, s \leftarrow \mathbb{Z}_q^*$ and set $\mathcal{U} = rg_1$ and $\mathcal{V} = sg_2$;
set $\Sigma_{\mathsf{NIWI}} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, \iota_1, p_1, \iota_2, p_2, \iota_3, \mathcal{U}, \mathcal{V})$;
$\mathtt{vk}_g \leftarrow (\mathtt{pk}_g, \Sigma_{\mathsf{NIWI}})$;
return $(\mathtt{sk}_g, \mathtt{vk}_g)$

Algorithm 13 Join_ProxyGr$_{\mathcal{P}/\mathcal{GM}}$ algorithm

Input: the system public parameters pp and the secret key of the group manager $\mathtt{sk}_g$
Output: the pair of keys of a proxy group member $(\mathtt{sk}_p, \mathtt{pk}_p)$ and the signature σ_p over the public key $\mathtt{pk}_p$
// **The next is set by** $\mathcal{P}$
pick at random $g_r, h_u \leftarrow \mathbb{G}_1^*$, $\gamma, \delta \leftarrow \mathbb{Z}_q^*$;
compute $g_\gamma \leftarrow {g_r}^{\gamma}$ and $h_\delta \leftarrow {h_u}^{\delta}$;
pick at random $\gamma_z, \delta_z \leftarrow \mathbb{Z}_q^*$;
compute $g_z \leftarrow {g_r}^{\gamma_z}$ and $h_z \leftarrow {h_u}^{\delta_z}$;
pick at random $\alpha, \beta \leftarrow \mathbb{Z}_q^*$;
set $\mathtt{pk}_p = (g_z, h_z, g_r, h_u, g_2^{\alpha}, g_2^{\beta}, g_\gamma, h_\delta)$ and $\mathtt{sk}_p = (\mathtt{pk}_p, \alpha, \beta, \gamma_z, \delta_z, \gamma, \delta)$;
// **The next is set by** $\mathcal{GM}$
$\sigma_p \leftarrow \mathsf{XSIG.Sign}(\mathtt{sk}_g, \mathtt{pk}_p)$;
return $(\mathtt{sk}_p, \mathtt{pk}_p, \sigma_p)$

- $\mathsf{Userkeygen}_{\mathcal{U}}$ – After receiving his identifier $\mathtt{ID}_{\mathcal{U}}$, a user generates his pair of secret and private keys $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$. Indeed, $\mathcal{U}$ randomly selects $q_{\mathcal{U}} \in \mathbb{Z}_q^*$ and sets $(\mathtt{sk}_{\mathcal{U}}, \mathtt{pk}_{\mathcal{U}})$ as

$$\mathtt{sk}_{\mathcal{U}} = q_{\mathcal{U}} \quad ; \quad \mathtt{pk}_{\mathcal{U}} = {h_{\mathcal{U}}}^{q_{\mathcal{U}}}$$

5.5.2 Generation Phase

- $\mathsf{Set\_CCM}_{\mathcal{U}}$ – For each epoch e, $\mathcal{U}_A$ and $\mathcal{U}_B$ generate random EBIDs $\mathtt{D}^e_{\mathcal{U}_A}$ and $\mathtt{D}^e_{\mathcal{U}_B}$, respectively. $\mathcal{U}_A$ and $\mathcal{U}_B$ exchange their EBIDs and each of them executes the $\mathsf{Set\_CCM}$ algorithm. $\mathcal{U}_A$ (resp. $\mathcal{U}_B$) computes $m^e_{AB} = \mathtt{D}^e_{\mathcal{U}_A} * \mathtt{D}^e_{\mathcal{U}_B}$ and sets the common contact element between $\mathcal{U}_A$ and $\mathcal{U}_B$ as $\mathtt{CCM}^e_{AB} = \mathbf{H}(m^e_{AB})$.

- $\mathsf{S\_PSign}_{\mathcal{S}}$ – After checking that he receives two copies of $\mathtt{CCM}^e_{AB}$, the server picks at random $r_s \leftarrow \mathbb{Z}_q^*$ and, relying on his secret key $\mathtt{sk}_{\mathcal{S}}$, he computes the two messages $\mathtt{PS}^e_{AB}$ and $\mathtt{PS}'^e_{AB}$ such that

$$\mathtt{PS}^e_{AB} = \mathtt{CCM}^e_{AB} y_1 r_s + y_2 \quad \textit{and} \quad \mathtt{PS}'^e_{AB} = \mathtt{CCM}^e_{AB} r_s$$

- $\mathsf{P\_Sign}_{\mathcal{P}}$ – We consider that when being requested by a user $\mathcal{U}_A$, the proxy opens a session and saves the user's identifier $\mathtt{ID}_{\mathcal{U}_A}$. This latter is used when executing the $\mathsf{P\_Sign}_{\mathcal{P}}$ algorithm (c.f. Algorithm 14) to generate a new message $\mathtt{M}^e_{AB}$ (Line 4). The proxy then signs $\mathtt{M}^e_{AB}$ (Line 6 – Line 8) following the $\mathsf{CSIG.Sign}$ algorithm and finally generates a proof π (Line 10 – Line 16) w.r.t. the $\mathsf{GSIG.Sign}$ algorithm. Note that from Line 6 to Line 16 is the same as the Sign algorithm of the proposed group signature scheme (cf. Section 5.4.5).

5.5.3 Verification Phase

Naive Verification

- $\mathsf{Sig\_Verify}_{\mathcal{HA}}$ – Given a contact list of user $\mathcal{U}_A$ (a list of tuples $(\mathtt{CCM}, \mathtt{M}, \pi)$ such that π can be parsed as $\{(\vec{\mathcal{A}}_i, \vec{\mathcal{B}}_i, \Gamma_i, t_i)\}_{i=1}^6$, $\{(\vec{\mathcal{C}}_i, \vec{\mathcal{D}}_i, \pi_i, \theta_i)\}_{i=1}^6$), $\mathcal{HA}$ verifies the validity of the group signature of each message, w.r.t. $\mathsf{GSIG.Verify}$ algorithm (cf. Section 5.4.4). That is, for every two tuples $(\vec{\mathcal{A}}_i, \vec{\mathcal{B}}_i, \Gamma_i, t_i)$ and $(\vec{\mathcal{C}}_i, \vec{\mathcal{D}}_i, \pi_i, \theta_i)$, i.e., $i \in \{1..6\}$, $\mathcal{HA}$ verifies that Equation 5.3 holds.

- $\mathsf{CCM\_Verify}_{\mathcal{HA}}$ – We consider that $\mathcal{HA}$ requests from $\mathcal{S}$ the message $\mathtt{PS}'$ corresponding to a contact message $\mathtt{CCM}$ contained in the contact list of user $\mathcal{U}_A$. The message $\mathtt{PS}'$ is taken as input with the message $\mathtt{M}$ (corresponding to $\mathtt{CCM}$), the server's public key $\mathtt{pk}_{\mathcal{S}}$ and the secret value $t_{\mathcal{U}_A}$ specific to user $\mathcal{U}_A$, to the $\mathsf{CCM\_Verify}_{\mathsf{HA}}$ algorithm that checks if the equation 5.4 holds:

$$\mathtt{M} = {Y_1}^{t_{\mathcal{U}_A}\mathtt{PS}'} {Y_2}^{t_{\mathcal{U}_A}} \tag{5.4}$$

Batch Verification

Algorithm 14 $\mathsf{P\_Sign}_{\mathcal{P}}$ algorithm

Input: the public parameters of the proxies' group $\mathtt{vk}_g$, the secret key $\mathtt{sk}_p$, the signature σ_p over the proxy's public key, the identifier $\mathtt{ID}_{\mathcal{U}_A}$ of user $\mathcal{U}_A$ and the message $\mathtt{PS}$

Output: a message $\mathtt{M}$, the corresponding signature σ_m and a proof π

// **The next is executed by $\mathcal{P}$ to generate $\mathtt{M}$**

compute $\mathtt{M} = {\mathtt{ID}_{\mathcal{U}_A}}^{\mathtt{PS}}$;

// **The next is executed by $\mathcal{P}$ to sign $\mathtt{M}$**

pick at random $\zeta, \rho, \tau, \varphi, \omega \leftarrow \mathbb{Z}_q^*$;

run $z = g_2^{\zeta}$, $r = {g_2}^{\alpha-\rho\tau-\gamma_z\zeta}\mathtt{M}^{-\gamma}$, $s = {g_r}^{\rho}$, $t = {g_2}^{\tau}$, $u = {g_2}^{\beta-\varphi\omega-\delta_z\zeta}\mathtt{M}^{-\delta}$, $v = {h_u}^{\varphi}$, $w = {g_2}^{\omega}$;

set $\sigma_m = (z, r, s, t, u, v, w)$;

// **The next is set to generate a proof on equations $\{(\vec{\mathcal{A}}_{im}, \vec{\mathcal{B}}_{im}, \Gamma_{im}, t_{im})\}_{i=1}^{2}$ where $\vec{\mathcal{A}}_{im} = \vec{\mathcal{B}}_{im} = \vec{0}$, $\Gamma_{im} = \mathcal{MAT}_{3\times3}(1)$ for $i = 1, 2$, $t_{1m} = t_{2m} = 1_{\mathbb{G}_3}$**

$\vec{\mathcal{X}}_{1m} = (g_z, g_r, s)$, $\vec{\mathcal{X}}_{2m} = (h_z, h_u, v)$, $\vec{\mathcal{Y}}_{1m} = (z, {g_2}^{\alpha-\rho\tau-\gamma_z\zeta}, t)$ and $\vec{\mathcal{Y}}_{2m} = (z, {g_2}^{\beta-\rho\tau-\delta z\zeta}, w)$;

$\pi_m = \{(\vec{\mathcal{C}}_{im}, \vec{\mathcal{D}}_{im}, \pi_{im}, \theta_{im})\}_{i=1}^{2} \leftarrow \mathsf{NIWI.Proof}(\mathtt{vk}_g, \{(\vec{\mathcal{A}}_{im}, \vec{\mathcal{B}}_{im}, \Gamma_{im}, t_{im})\}_{i=1}^{2}, \{(\vec{\mathcal{X}}_{im}, \vec{\mathcal{Y}}_{im})\}_{i=1}^{2})$;

// **The next is set to generate a proof on equations $\{(\vec{\mathcal{A}}_{ip}, \vec{\mathcal{B}}_{ip}, \Gamma_{ip}, t_{ip})\}_{i=1}^{4}$ where $\vec{\mathcal{A}}_{1p} = (g_1^{\alpha_2})$, $\vec{\mathcal{A}}_{2p} = (g_1^{\beta_2})$, $\vec{\mathcal{A}}_{3p} = (g_{1z}, g_{1r})$, $\vec{\mathcal{A}}_{4p} = (h_{1z}, h_{1u})$, $\vec{\mathcal{B}}_{1p} = (g_{2z}, g_{2r})$, $\vec{\mathcal{B}}_{2p} = (h_{2z}, h_{2u})$, $\vec{\mathcal{B}}_{3p} = (g_2^{\alpha_1})$, $\vec{\mathcal{B}}_{4p} = (g_2^{\beta_1})$, $\Gamma_{1p} = (\gamma_{2z}, -1)$, $\Gamma_{2p} = (\delta_{2z}, -1)$, $\Gamma_{3p} = (\gamma_{1z}, -1)$, $\Gamma_{4p} = (\delta_{1z}, -1)$, $t_{1p} = e(g_1^{\alpha_2}, g_{2r})$, $t_{2p} = e(g_1^{\beta_2}, h_{2u})$, $t_{3p} = e(g_{1r}, g_2^{\alpha_1})$ and $t_{4p} = e(h_{1u}, g_2^{\beta_1})$**

$\vec{\mathcal{X}}_{1p} = (z_1, {g_1}^{\alpha_2-\rho_1\tau_1-\gamma_{2z}\zeta_1})$, $\vec{\mathcal{X}}_{2p} = (z_1, {g_1}^{\beta_2-\rho_1\tau_1-\delta_{2z}\zeta_1})$, $\vec{\mathcal{X}}_{3p} = (g_{1r})$, $\vec{\mathcal{X}}_{4p} = (h_{1u})$, $\vec{\mathcal{Y}}_{1p} = (g_{2r})$, $\vec{\mathcal{Y}}_{2p} = (h_{2u})$, $\vec{\mathcal{Y}}_{3p} = (z_2, {g_2}^{\alpha_1-\rho_2\tau_2-\gamma_{1z}\zeta_2})$, and $\vec{\mathcal{Y}}_{4p} = (z_2, {g_2}^{\beta_1-\rho_2\tau_2-\delta_{1z}\zeta_2})$;

$\pi_p = \{(\vec{\mathcal{C}}_{ip}, \vec{\mathcal{D}}_{ip}, \pi_{ip}, \theta_{ip})\}_{i=1}^{4} \leftarrow \mathsf{NIWI.Proof}(\mathtt{vk}_g, \{(\vec{\mathcal{A}}_{ip}, \vec{\mathcal{B}}_{ip}, \Gamma_{ip}, t_{ip})\}_{i=1}^{4}, \{(\vec{\mathcal{X}}_{ip}, \vec{\mathcal{Y}}_{ip})\}_{i=1}^{4})$;

set $\pi_p = ((\pi_{ip}, \theta_{ip})_{i=1}^{4})$;

set $\pi = (\pi_p, \pi_m)$;

return $(\mathtt{M}, \sigma_m, \pi)$

- $\mathsf{Batch\_Sig\_Verify}_{\mathcal{HA}}$ – We consider a list of N messages m_i and the corresponding proofs Π_i. Each proof Π_i is composed of six sub-proofs (i.e., two sub-proofs generated over the signature σ_m w.r.t. the message m, and four sub-proofs generated over the signature σ_p w.r.t. the proxy's key $\mathtt{pk}_p$). The list of proofs can be presented as follows:

$$\begin{cases} \{(\vec{\mathcal{A}}_{ijm}, \vec{\mathcal{B}}_{ijm}, \Gamma_{ijm}, t_{ijm})\}_{i,j=1}^{i=N,j=2}, \\ \{(\vec{\mathcal{C}}_{ijm}, \vec{\mathcal{D}}_{ijm}, \pi_{ijm}, \theta_{ijm})\}_{i,j=1}^{i=N,j=2}, \\ \{(\vec{\mathcal{A}}_{ilp}, \vec{\mathcal{B}}_{ilp}, \Gamma_{ilp}, t_{ilp})\}_{i,l=1}^{i=N,l=4}, \\ \{(\vec{\mathcal{C}}_{ilp}, \vec{\mathcal{D}}_{ilp}, \pi_{ilp}, \theta_{ilp})\}_{i,l=1}^{i=N,l=4}. \end{cases}$$

Referring to the generation of the group signature (cf. Algorithm 14), the

tuples $\{(\vec{\mathcal{A}_{jm}}, \vec{\mathcal{B}_{jm}}, \Gamma_{jm}, t_{jm})\}_{j=1}^{2}$ and $\{(\vec{\mathcal{A}_{lp}}, \vec{\mathcal{B}_{lp}}, \Gamma_{lp}, t_{lp})\}_{l=1}^{4}$ are unchanged for all N proofs and all proxies. Thus, for a given list of messages, $\mathcal{HA}$ verifies the validity of the proofs by checking if equations 5.5 and 5.6 hold. Note that the $\mathsf{Batch\_Sig\_Verify}_{\mathcal{HA}}$ algorithm is the same as the $\mathsf{Batch\_Verify}$ algorithm of the proposed group signature scheme (cf. Section 5.4.5).

$$\prod_i \prod_j \left(e(\vec{\mathcal{C}_{ijm}}, \Gamma_m \vec{\mathcal{D}_{ijm}})\right) = e(\mathtt{U}, \sum_i \sum_j \pi_{ijm}) e(\sum_i \sum_j \theta_{ijm}, \mathtt{V}) \tag{5.5}$$

$$\prod_l e(\iota_1(\vec{\mathcal{A}_{lp}}), \sum_i \vec{\mathcal{D}_{ilp}}) e(\sum_i \vec{\mathcal{C}_{ilp}}, \iota_2(\vec{\mathcal{B}_{lp}})) \prod_i \prod_l \left(e(\vec{\mathcal{C}_{ilk}}, \Gamma_{lk} \vec{\mathcal{D}_{ilk}})\right) = \left(\prod_l \iota_3(t_l p)^N\right) e(\mathtt{U}, \sum_i \sum_l \pi_{ilp}) e(\sum_i \sum_l \theta_{ilp}, \mathtt{V}) \tag{5.6}$$

- $\mathsf{Agg\_Sig\_Verify}_{\mathcal{HA}}$ – We consider a message m belonging to an invalid proof-list and its corresponding proof Π. Using the tuples $\{(\vec{\mathcal{A}_{jm}}, \vec{\mathcal{B}_{jm}}, \Gamma_{jm}, t_{jm})\}_{j=1}^{2}$ and $\{(\vec{\mathcal{A}_{lp}}, \vec{\mathcal{B}_{lp}}, \Gamma_{lp}, t_{lp})\}_{l=1}^{4}$ along with the tuples $\{(\vec{\mathcal{C}_{jm}}, \vec{\mathcal{D}_{jm}}, \pi_{jm}, \theta_{jm})\}_{j=1}^{j=2}$ and $\{(\vec{\mathcal{C}_{lp}}, \vec{\mathcal{D}_{lp}}, \pi_{lp}, \theta_{lp})\}_{l=1}^{l=4}$ derived from Π, $\mathcal{HA}$ checks if equations 5.7 and 5.8 hold. Note that the $\mathsf{Agg\_Sig\_Verify}_{\mathcal{HA}}$ algorithm is equivalent to the $\mathsf{Agg\_Verify}$ algorithm of the proposed group signature scheme (cf. Section 5.4.5).

$$\prod_j \left(e(\vec{\mathcal{C}_{jm}}, \Gamma_m \vec{\mathcal{D}_{jm}})\right) = e(\mathtt{U}, \sum_j \pi_{jm}) e(\sum_j \theta_{jm}, \mathtt{V}) \tag{5.7}$$

$$\prod_l e(\iota_1(\vec{\mathcal{A}_{lp}}), \vec{\mathcal{D}_{lp}}) e(\vec{\mathcal{C}_{lp}}, \iota_2(\vec{\mathcal{B}_{lp}})) \prod_l \left(e(\vec{\mathcal{C}_{lp}}, \Gamma_{lk} \vec{\mathcal{D}_{lp}})\right) = \left(\prod_l \iota_3(t_l p)\right) e(\mathtt{U}, \sum_l \pi_{lp}) e(\sum_l \theta_{lp}, \mathtt{V}) \tag{5.8}$$

- $\mathsf{Batch\_CCM\_Verify}_{\mathcal{HA}}$ – We consider that $\mathcal{HA}$ requests from $\mathcal{S}$ the list of N messages $\{\mathtt{PS}'_i\}_{i=1}^{N}$ corresponding to the contact messages $\{\mathtt{CCM}_i\}_{i=1}^{N}$ contained in the contact list of user $\mathcal{U}_A$. The list $\{\mathtt{PS}'_i\}_{i=1}^{N}$ is taken as input with the list of messages $\{\mathtt{M}_i\}_{i=1}^{N}$ (corresponding to the list $\{\mathtt{CCM}_i\}_{i=1}^{N}$), the server's public key $\mathtt{pk}_{\mathcal{S}}$ and the secret value $t_{\mathcal{U}_A}$ specific to user $\mathcal{U}_A$, to the $\mathsf{Batch\_CCM\_Verify}_{\mathsf{HA}}$ algorithm that checks if the equation 5.9 holds:

$$\prod_i \mathtt{M}_i = Y_1^{t_{\mathcal{U}_A} \sum_i \mathtt{PS}'_i} Y_2^{N t_{\mathcal{U}_A}} \tag{5.9}$$

Correctness of Equations 5.5, 5.6, 5.7 and 5.8 w.r.t. Equation 5.3

$$\begin{aligned} & \prod_i \prod_j e(\iota_1(\vec{\mathcal{A}_{ij}}), \vec{\mathcal{D}_{ij}}) e(\vec{\mathcal{C}_{ij}}, \iota_2(\vec{\mathcal{B}_{ij}})) e(\vec{\mathcal{C}_{ij}}, \Gamma_{ij} \vec{\mathcal{D}_{ij}}) \\ = \; & \prod_i \prod_j \iota_3(t_{ij}) e(\mathcal{U}, \pi_{ij}) e(\theta_{ij}, \mathcal{V}) \\ = \; & \left(\prod_i \prod_j \iota_3(t_{ij})\right) e(\mathcal{U}, \sum_i \sum_j \pi_{ij}) e(\sum_i \sum_j \theta_{ij}, \mathcal{V}) \end{aligned}$$

The values of $\vec{\mathcal{A}_{ij}}$, $\vec{\mathcal{B}_{ij}}$, $\vec{\mathcal{C}_{ij}}$, $\vec{\mathcal{D}_{ij}}$, Γ_{ij}, t_{ij}, π_{ij} and θ_{ij} are replaced with the values given in Algorithm 14 in order to get the simplified equations.

Correctness of Equation 5.9

$$\begin{aligned}\prod_i \mathtt{M}_i &= \prod_i \mathtt{ID}_{\mathcal{U}_A}{}^{\mathtt{PS}_i} \\ &= \prod_i g_2^{t_{\mathcal{U}_A}(\mathtt{PS}'_i y_1 + y_2)} \\ &= \prod_i Y_1^{t_{\mathcal{U}_A}\mathtt{PS}'_i} Y_2^{t_{\mathcal{U}_A}} \\ &= Y_1^{t_{\mathcal{U}_A}\sum_i \mathtt{PS}'_i} Y_2^{N t_{\mathcal{U}_A}}\end{aligned}$$

5.6 Security and Privacy Analysis

In this section, we prove that SPOT achieves the defined security and privacy requirements with respect to the threat models defined in Section 5.3.4.

5.6.1 Unforgeability

Theorem 8 (Unforgeability)**.** If a probabilistic-polynomial time (PPT) adversary $\mathcal{A}$ wins $\mathbf{Exp}_{\mathcal{A}}^{unforg}$, as defined in Section 5.3.4.1, with a non-negligible advantage ϵ, then a PPT simulator $\mathcal{B}$ can be constructed to break the Computational Diffie Hellman (CDH) assumption with a non-negligible advantage ϵ.

Recall that the CDH assumption can be defined as follows: Let $\mathbb{G}$ be a group of prime order n, and g is a generator of $\mathbb{G}$. The CDH problem is defined as: Given the tuple of elements (g, g^x, g^y), where $\{x, y\} \leftarrow \mathbb{Z}_q$, there is no efficient algorithm $\mathcal{A}_{CDH}$ that can compute g^{xy}.

Proof. In this proof, we show that a simulator $\mathcal{B}$ can be constructed with the help of an adversary $\mathcal{A}$ having advantage ϵ against SPOT scheme.

The CDH challenger $\mathcal{C}$ sends to $\mathcal{B}$ the tuple (g_2, g_2^a, g_2^b), where $a, b \leftarrow \mathbb{Z}_q^*$ are randomly selected. $\mathcal{C}$ asks $\mathcal{B}$ to compute g_2^{ab}. Then, $\mathcal{B}$ sets $g_2^{t_{\mathcal{U}}}$ to g_2^a and $g_2^{y_2}$ to g_2^b. During the challenge phase, $\mathcal{B}$ randomly selects $y_1 \in \mathbb{Z}_q^*$ and sends $g_2^{y_1}$ to $\mathcal{A}$ as part of the server's public key. $\mathcal{A}$ forges the partial signature over the message $\mathtt{PS}'$ and generates the message $\mathtt{M}^*$ with advantage ϵ: $\mathtt{M}^* = \mathtt{ID}_{\mathcal{U}}{}^{\mathtt{PS}^*} = {g_2}^{t_{\mathcal{U}}(\mathtt{PS}' y_1 + y_2)}$. The tuple $({g_2}^{t_{\mathcal{U}}}, \mathtt{PS}', \mathtt{M}^*)$ is sent back to $\mathcal{B}$. Upon receiving this tuple and knowing y_1, $\mathcal{B}$ can compute the value of $g_2^{t_{\mathcal{U}} y_2}$ which is the same as g_2^{ab} and can then send the result to the CDH challenger. As such, $\mathcal{B}$ succeeds the forgery against the CDH assumption with advantage ϵ. □

5.6.2 Unlinkability

Theorem 9 (Unlinkability)**.** SPOT system achieves the unlinkability requirement with respect to the *group-signature unlinkability* and *multi-CCM unlinkability* properties.

We prove Theorem 9 through Lemma 10 and Lemma 11 with respect to *group-signature unlinkability* and *multi-CCM unlinkability* properties, respectively.

Lemma 10 (Group-signature unlinkability)**.** SPOT satisfies the group signature unlinkability requirement with respect to the computational witness indistinguishability property of the NIWI proof.

Proof. In this proof, the objective is to show that the adversary is not able to distinguish group signatures issued by the same proxy. For this purpose, we suppose that, for each session i, the adversary receives the message M^* (i.e., the same message M^* is returned by each oracle) and the NIWI proof $\pi^i = (\pi^i_m, \pi^i_p) = ((\pi^i_{jm}, \theta^i_{jm})^2_{j=1}, (\pi^i_{jp}, \theta^i_{jp})^4_{j=1})$.
To simplify the proof, we will only consider the NIWI proof π^i_m, as the statements used to generate the proofs π^i_p do not give any information about the proxy generating the proof (i.e., statements do not include the proxy's public key). Thus, for each session i, the adversary is given the tuples $(\vec{\mathcal{C}^i_{k1}}, \vec{\mathcal{D}^i_{k1}}, \pi^i_{k1}, \theta^i_{k1})$ and $(\vec{\mathcal{C}^i_{k2}}, \vec{\mathcal{D}^i_{k2}}, \pi^i_{k2}, \theta^i_{k2})$ referred to as the group signature generated by a proxy P_k, where $k \in \{0, 1\}$. During the challenge phase, the adversary is also given two group signatures. The first signature is represented by the tuples $(\vec{\mathcal{C}^*_1}, \vec{\mathcal{D}^*_1}, \pi^*_1, \theta^*_1)$ and $(\vec{\mathcal{C}^*_2}, \vec{\mathcal{D}^*_2}, \pi^*_2, \theta^*_2)$ generated by proxy $\mathcal{P}_0$, while the second one is represented by the tuples $(\vec{\mathcal{C}^*_{b1}}, \vec{\mathcal{D}^*_{b1}}, \pi^*_{b1}, \theta^*_{b1})$ and $(\vec{\mathcal{C}^*_{b2}}, \vec{\mathcal{D}^*_{b2}}, \pi^*_{b2}, \theta^*_{b2})$ and is generated by a proxy $\mathcal{P}_b$ ($b \in \{0, 1\}$).

Let us consider a simulator $\mathcal{B}$ that can be constructed with the help of an adversary $\mathcal{A}$ having advantage ϵ against SPOT scheme. A challenger $\mathcal{C}$ selects two couples of witnesses (X_0, Y_0) and (X_1, Y_1). $\mathcal{C}$ computes a commitment (C, D) over (X_0, Y_0), and then selects a bit $b \in \{0, 1\}$ and computes a commitment (C'_b, D'_b) over (X_b, Y_b). $\mathcal{C}$ asks $\mathcal{B}$ to guess the bit b. Then, $\mathcal{B}$ selects the tuples (A, B, Γ, t) and $(A'_b, B'_b, \Gamma'_b, t'_b)$ and computes the proofs (π, θ) and (π'_b, θ'_b). $\mathcal{B}$ returns the two proofs to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs a bit b' that it sends to $\mathcal{B}$. This latter outputs the same bit b' to its own challenger $\mathcal{C}$. As such, $\mathcal{A}$ succeeds in breaking the group-signature unlinkability with advantage ϵ, which is the same as breaking the computational witness-indistinguishability property.

□

Corollary 10.1. If SPOT satisfies the unlinkability property, then the proposed group signature, in Section 5.4.5, also ensures the unlinkability between several group signatures issued by the same signer.

Corollary 10.2. If SPOT satisfies the unlinkability property, then the proxies' group signature (i.e., NIWI proof) fulfills the anonymity requirement stating that it is not possible to identify the proxy that issued a particular group signature.

Lemma 11 (Multi-CCM unlinkability)**.** SPOT satisfies the multi-CCM unlinkability requirement with respect to the common contact message structure.

Proof. Let $\mathcal{A}$ be a successful adversary against the *multi-CCM unlinkability* property. Assume that $\mathcal{A}$ receives two messages $\texttt{CCM}_1 = \mathbf{H}(\texttt{D}_i * \texttt{D}_j)$ and $\texttt{CCM}_2 = \mathbf{H}(\texttt{D}_i * \texttt{D}_k)$ (with $j \neq k$) meaning that user $\mathcal{U}_i$ met $\mathcal{U}_j$ and $\mathcal{U}_k$ during the same epoch e.
To link $\texttt{CCM}_1$ and $\texttt{CCM}_2$ to the same user $\mathcal{U}_i$, $\mathcal{A}$ should be able to deduce that the same EBID $\texttt{D}_i$ is used to compute the two messages. However, this is not feasible as all EBIDs are randomly generated in each epoch e, and the hashing function $\mathbf{H}$ is a one-way function and behaves as a pseudo-random function. □

5.6.3 Anonymity

Theorem 12 (Anonymity)**.** SPOT satisfies the anonymity property, in the sense of Definition 15, if and only if, the CCM-unlinkability requirement is fulfilled.

Proof. We prove that our proximity-based protocol SPOT satisfies the anonymity property using an *absurdum* reasoning. We suppose that an adversary $\mathcal{A}$ can break the anonymity of SPOT, in the sense of Definition 15, by reaching the advantage $Pr[\mathbf{Exp}^{anon}_{\mathcal{A}}(1^\lambda) = 1] \geq \frac{1}{2} \pm \kappa(\lambda)$. $\mathcal{A}$ is given the pair of public-private keys $(\texttt{pk}_{\mathcal{HA}}, \texttt{sk}_{\mathcal{HA}})$ of the health authority, the identifiers $\texttt{ID}_{\mathcal{U}}$ of all users and a contact list of a particular user $\mathcal{U}_A$, obtained when relying on several sessions. Then, relying on the *left-or-right* LoRCU oracle, $\mathcal{A}$ tries to distinguish the user $\mathcal{U}_b$ being in contact with $\mathcal{U}_A$, better than a flipping coin. That is, given the tuple $(\mathsf{CCM}^*_b, \texttt{M}^*_b, \pi^*_b)$, $\mathcal{A}$ successfully predicts the identifier $\texttt{ID}_{\mathcal{U}_b}$. Obviously, $\mathcal{A}$ tries to identify user $\mathcal{U}_b$ relying on the message CCM^*_b, since both $\texttt{M}^*_b$ and π^*_b are generated based on CCM^*_b and give no further information about the user $\mathcal{U}_b$. This refers to link the message CCM^*_b to its issuer $\mathcal{U}_b$. Thus, if $\mathcal{A}$ succeeds, this means that $\mathcal{A}$ is able to link two or several common contact messages to the same user, which contradicts the *multi-CCM unlinkability* property previously discussed. As such, we prove that the adversary succeeds $\mathbf{Exp}^{anon}_{\mathcal{A}}(1^\lambda)$ with a probability $Pr = \frac{1}{2} \pm \kappa(\lambda)$, where $\kappa(\lambda)$ is negligible. Thus, SPOT satisfies anonymity.

□

5.6.4 Anti-replay

Theorem 13 (Anti-replay)**.** SPOT satisfies the anti-replay requirement and supports false positive hindrance, if SPOT is unforgeable.

Proof. To successfully replay a common contact message generated in an epoch e, in another epoch $e' \neq e$, a malicious user can perform in two ways.

(i) The user reinserts, in his contact list, the tuple ($\texttt{CCM}^e$, $\texttt{M}^e$, π^e) generated in an epoch e. The reinsertion is performed in an epoch $e' > e + \Delta$[10]. Afterwards, the contact list is sent to $\mathcal{HA}$ when the user is infected. $\mathcal{HA}$ asks the server to provide the message $\texttt{PS}'$ corresponding to $\texttt{CCM}^e$. As the server has no entry corresponding to $\texttt{CCM}^e$ in the last Δ days, the second verification performed by $\mathcal{HA}$ does not hold and the tuple is rejected. (ii) We assume that, in an epoch $e' > e + \Delta$, the user is able to replay a message $\texttt{CCM}^e$ with two different proxies and he successfully receives the corresponding message $\texttt{M}$ and the group signature π. Thus, when the user is infected, the health authority validates false positives, but this has no impact on the computation of the risk score, as no user has the same entry in his contact list. As such, we can prove the resistance of SPOT against replay attacks. □

5.7 Performance Analysis

This section discusses the experimental results, presented in Table 5.2, and demonstrates the usability of the proposed construction for real world scenarios. We, first, describe SPOT test-bed in Section 5.7.1 and we discuss the communication overhead in Section 5.7.2. Then, we analyze, in Section 5.7.3, the computation performances of SPOT w.r.t. computation improvements and batch verification.

Table 5.2: Computation Time and Communication Overhead of SPOT Algorithms

Algorithm	Entity	Synch/Asynch	Communication cost	Computation time (ms)			
				A/112-bits	A/128-bits	F/112-bits	F/128-bits
Set_params	$\mathcal{TA}$	Asynch.	$\lvert\mathbb{Z}_q\rvert + \lvert\mathbb{G}_1\rvert + \lvert\mathbb{G}_2\rvert + \lvert\mathbb{G}_3\rvert$	874	2521	1230	1364
HA_Keygen	$\mathcal{TA}$	Asynch.	$\lvert\mathbb{G}_1\rvert$	59	123	12	16
S_Keygen	$\mathcal{TA}$	Asynch.	$2\lvert\mathbb{G}_2\rvert$	119	244	24	31
Setup_ProxyGr	$\mathcal{GM}$	Asynch.	$21(\lvert\mathbb{G}_1\rvert + \lvert\mathbb{G}_2\rvert)$	1955	4075	346	451
Join_ProxyGr	$\mathcal{P}/\mathcal{GM}$	Synch.	$\mathcal{P} : 8\lvert\mathbb{G}_1\rvert + 2\lvert\mathbb{G}_2\rvert$ / $\mathcal{GM} : 7(\lvert\mathbb{G}_1\rvert + \lvert\mathbb{G}_2\rvert)$	2861	6014	1159	1409
Set_UserID	$\mathcal{HA}$	Synch.	$\lvert\mathbb{G}_2\rvert$	58	121	12	16
Userkeygen	$\mathcal{U}$	Asynch.	$\lvert\mathbb{G}_2\rvert$	117	242	24	31
Set_CCM	$\mathcal{U}$	Synch.	$\lvert\mathbb{Z}_q\rvert$	0.1	0.1	0.1	0.1
S_PSign [a]	$\mathcal{S}$	Synch.	$\lvert\mathbb{Z}_q\rvert$	0.1	0.08	0.02	0.02
P_Sign [a]	$\mathcal{P}$	Synch.	$6\lvert\mathbb{G}_1\rvert + 7\lvert\mathbb{G}_2\rvert$	19353	40371	3164	4170
Sig_Verify [a]	$\mathcal{HA}$	Asynch.	N.A.	6541	15406	31637	36892
CCM_Verify [a]	$\mathcal{HA}$	Asynch.	N.A.	174	360	148	190
Batch_Sig_Verify [b,c]	$\mathcal{HA}$	Asynch.	N.A.	222989	485233	1018375	1312879
Agg_Sig_Verify [a,c]	$\mathcal{HA}$	Asynch.	N.A.	3096	6916	16065	18834
Batch_CCM_Verify [b]	$\mathcal{HA}$	Asynch.	N.A.	139	281	133	175

NOTE: Synch./Asynch. indicates whether the algorithm must be run online (i.e., in real time) or offline (i.e., later); [a] indicates that the algorithm is performed on a single contact message that is generated by the Set_CCM algorithm; [b] indicates that the algorithm is performed on N messages where $N = 100$ for computation times; [b] indicates that the computation times are the ones obtained after applying the improvements (cf. Section 5.8.3.2); $\lvert\mathbb{G}_1\rvert$ (resp. $\lvert\mathbb{G}_2\rvert$, $\lvert\mathbb{G}_3\rvert$ and $\lvert\mathbb{Z}_q\rvert$) indicates the size of an element in $\mathbb{G}_1$ (resp. $\mathbb{G}_2$, $\mathbb{G}_3$ and $\mathbb{Z}_q$); N.A. is the abbreviation for Not Applicable.

[10]It makes no sense to reinsert an element in an epoch $e' < e + \Delta$, as duplicated messages will be deleted either by the server or at the user's end-device.

5.7.1 Test-bed and Methodology

For our experiments, we developed a prototype of the SPOT protocol that implements the three phases SYS_INIT, GENERATION and VERIFICATION including the fifteen algorithms[11]. The tests are made on the test-bed Device 1 described in Chapter 3, Table 3.3. All algorithms were implemented based on JAVA version 11, and the cryptographic library *JPBC*[12]. We evaluate the computation time of each algorithm relying on two types of bilinear pairings, i.e., *type A* and *type F*. The pairing *type A* is the fastest symmetric pairing type in the *JPBC* library constructed on the curve $y^2 = x^3 + x$ with an embedding degree equal to 2. The pairing *type F* is an asymmetric pairing type introduced by Barreto and Naehrig [75]. It has an embedding degree equal to 12. For the two types of pairing, we consider two different levels of security i.e., 112-bits and 128-bits security levels recommended by the US National Institute of Standards and Technology[13] (NIST).
Based on the selected cryptographic library and the implementation of Groth-Sahai proofs[14], the SPOT test-bed is built with six main java classes, w.r.t. to the different entities of SPOT, referred to as *TrustedAuthority.java*, *GroupManager.java*, *Proxy.java*, *HealthAuthority.java*, *User.java* and *Server.java*. Each class encompasses the algorithms that are performed by the relevant entity as described in Section 5.3.1. In order to obtain accurate measurements of the computation time, each algorithm is run 100 times. Thus, the computation times represent the mean of the 100 runs while considering a standard deviation of an order 10^{-2}.

5.7.2 Communication Cost

This section proposes a theoretical analysis of the communication cost of the SPOT protocol.

As presented in Table 5.2, the communication cost is evaluated according to the size of the elements $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ and $\mathbb{Z}_q$ exchanged between entities. Each pairing type and each security level are characterized with different group sizes.
From Table 5.2, it is worth noticing that the SYS_INIT phase is the most consuming in terms of bandwidth. In fact, it includes the Set_params and Setup_ProxyGr algorithms that output the system and group public parameters shared with other entities. The SYS_INIT phase also includes the interactive Join_ProxyGr algorithm that introduces communication overheads of $8|\mathbb{G}_1| + 2|\mathbb{G}_2|$ and $7(|\mathbb{G}_1| + |\mathbb{G}_2|)$ to respectively send the proxy's keys to the $\mathcal{GM}$ and give back $\mathcal{GM}$'s signature over the keys of $\mathcal{P}$. The communication cost introduced by the SYS_INIT phase must be put into perspective as both algorithms Set_params and Setup_ProxyGr are executed once, and the Join_ProxyGr algorithm is performed only when a new proxy wants to join the group.
For the GENERATION phase, except from the P_Sign algorithm, all algorithms intro-

[11] The source code is available at https://github.com/soumasmoudi/SPOTv2
[12] http://gas.dia.unisa.it/projects/jpbc/
[13] http://keylength.com
[14] https://github.com/gijsvl/groth-sahai

duce a communication overhead of one group element size. The P_Sign algorithm, which repeatedly performed by proxies, adds an acceptable communication overhead of $6|\mathbb{G}_1| + 7|\mathbb{G}_2|$ due to the size of the NIWI proof.

5.7.3 Computation Overhead

From Table 5.2, it is worth noticing that the computation time depends on the selected pairings types and is strongly related to the security level. Some algorithms of the SYS_INIT phase, like Set_params and Setup_ProxyGr, are consuming but they are limited to only one execution, respectively from a powerful trusted authority and a group manager. The performances of the Join_ProxyGr algorithm depend on the selected pairing type and security level. Indeed, for symmetric pairing settings (i.e., pairing *type A*), it requires 2 and 6 seconds, for 112-bits and 128-bits security levels, respectively. For asymmetric pairing settings (i.e., pairing *type F*), the computation time of the Join_ProxyGr algorithm reaches 1,2 and 1,4 seconds, for 112-bits and 128-bits security levels, respectively.

For the GENERATION phase, the most consuming algorithm is P_Sign which requires 19 seconds (resp. 40 seconds) in symmetric pairing settings, and 3 seconds (resp. 4 seconds) in asymmetric pairing settings. The computation time of Set_CCM and S_PSign are negligible, which means that the user and the server are not required to have important computation capacities.

For the VERIFICATION phase, to verify the correctness of a single contact message, through a naive verification, the Sig_Verify and CCM_Verify algorithms together require approximately 7 seconds (resp. 15 seconds) for pairing *type A* and 32 seconds (resp. 37 seconds) for pairing *type F*. Meanwhile, the Batch_Sig_Verify algorithm that is run to verify 100 messages simultaneously, requires approximately 4 and 8 minutes for pairing *type A* and 17 and 22 minutes for pairing *type F*. However, when it is needed to verify a single message, the Agg_Sig_Verify algorithm requires 3 and 7 seconds for pairing *type A* and 16 and 19 seconds for pairing *type F*. It is worth noticing that, for a number of messages $N = 100$, the execution of the Batch_Sig_Verify algorithm gives improved computational costs compared to the Agg_Sig_Verify algorithm performed 100 times, separately. Also the Batch_CCM_Verify algorithm gives promising results when verifying 100 messages at once.

It is, thus, clear that P_Sign, Sig_Verify, Batch_Sig_Verify and Agg_Sig_Verify are the most consuming algorithms in terms of computation time as they include a large number of exponentiations and pairing functions. However, this result must be put into perspective as both the proxy and the health authority are assumed to have advanced hardware features.

From Table 5.2, it is also worth mentioning that the P_Sign algorithm performed with asymmetric pairing settings is faster than with symmetric settings. Indeed, the elementary functions of multiplication and exponentiation are more consuming for

pairing *type A* than for pairing *type F*[15]. However, the Sig_Verify, Batch_Sig_Verify and Agg_Sig_Verify algorithms have an opposite behavior with a faster execution with pairing *type A* than with pairing *type F*. This can be explained by the excessive memory allocation and deallocation needed by pairing *type F*.

We finally deduce, from Table 5.2, that the algorithms run at the user's side, have very low computation and communication overhead, which confirms the usability of SPOT, even when being run on a smartphone with low capacities.

For consuming algorithms that are repeatedly run namely, P_Sign, Sig_Verify, Batch_Sig_Verify and Agg_Sig_Verify, some performance improvement means are proposed in the next section.

5.7.4 Impact of Multithreading and Preprocessing Improvements on Computation Overhead

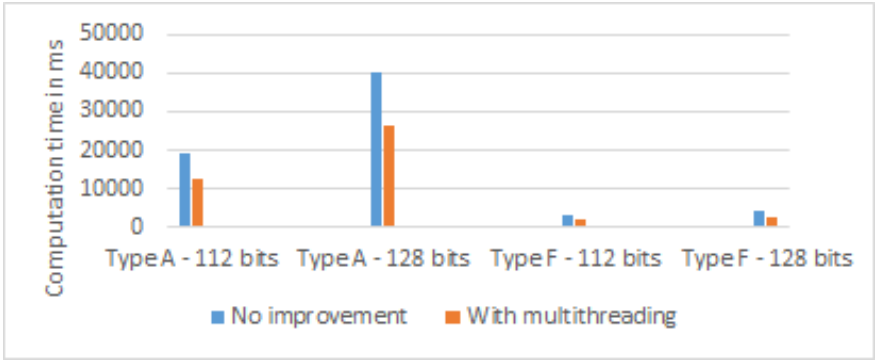
In an effort to make the computation time as efficient as possible for the four algorithms P_Sign, Sig_Verify, Batch_Sig_Verify and Agg_Sig_Verify, we rely on a two step improvement:

- **Multithreading:** applied on the four algorithms. It enables simultaneous multiple threads execution. The multithreading helps P_Sign to compute different parts of the NIWI proof simultaneously, while for Sig_Verify, Batch_Sig_Verify and Agg_Sig_Verify algorithms, it enables higher computation throughput on both sides of the verification equations of NIWI proofs.
- **Preprocessing:** applied only on the three algorithms Sig_Verify, Batch_Sig_Verify and Agg_Sig_Verify. It helps reduce the computation time when some variables need to be computed several times during the execution of the algorithm. This is the case for the variables (e.g., $\mathtt{U}$ and $\mathtt{V}$ of the CRS Σ_{NIWI}) which can be prepared in advance (i.e., before the execution of algorithms) for next be provided as input to the pairing functions of the Sig_Verify, Batch_Sig_Verify and Agg_Sig_Verify algorithms.

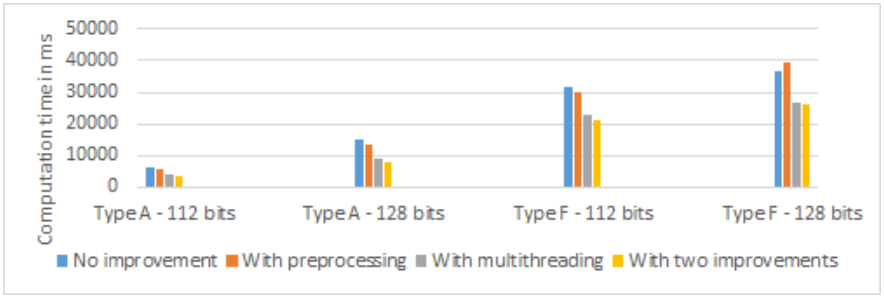
Note that, in our experiments, we only test the impact of one or two combined improvements on the P_Sign and Sig_Verify algorithms. For the Batch_Sig_Verify and Agg_Sig_Verify algorithms, as shown in Table 5.2, we only consider the computation times after improvements.

Figure 5.3 exposes the impacts of one or two combined improvements applied to P_Sign and Sig_Verify. From Figure 5.3a, we notice that multithreading reduces the computation time for P_Sign of approximately 35%, for the two types of pairing and the two levels of security. For Sig_Verify, Figure 5.3b shows that multithreading has a

[15]The experimental results are thus compliant to the *JPBC* library http://gas.dia.unisa.it/projects/jpbc/benchmark.html



(a) Influence of Multithreading on P_Sign Algorithm



(b) Influence of Preprocessing or/and Multithreading on Sig_Verify Algorithm

Figure 5.3: Influence of Improvements on P_Sign and Sig_Verify Algorithms

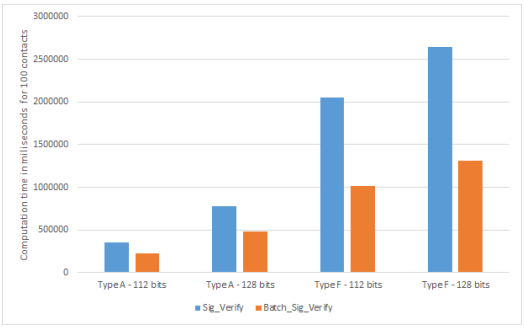
greater impact on the computation time (i.e., approximately 40% for pairing type A and 28% for pairing type F) than preprocessing (i.e., approximately 10% for pairing type A and 5% for pairing type F [16]). The two combined improvements ensure a gain of almost 50% for pairing type A and 30% for pairing type F.

5.7.5 Impact of Batch Verification on Computation Overhead

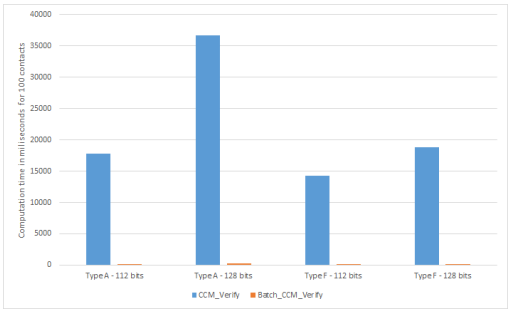
In this section, we focus on the VERIFICATION phase. We first, give a comparative analysis of the computation time of SPOT batch verification against the naive verification. Then, we evaluate the impact of the messages' number on the computation time.

Benefit of Batch Verification over Naive Verification We consider 100 contact messages partially and fully signed with the S_PSign and P_Sign algorithms, respectively. The resulting proofs (resp. partial signatures) are given as input to both Sig_Verify and Batch_Sig_Verify algorithms (resp. both CCM_Verify and Batch_CCM_Verify algorithms). The Sig_Verify and CCM_Verify algorithms are executed 100 times as they allows to perform verification over a single contact message, while the Batch_Sig_Verify and Batch_CCM_Verify algorithms perform the verification of all the 100 contact mes-

[16] For type F - 128 bits, the preprocessing decreases the performances. This is due to the excessive memory allocation and deallocation required by the pairing type F.



(a) Sig_Verify vs Batch_Sig_Verify



(b) CCM_Verify vs Batch_CCM_Verify

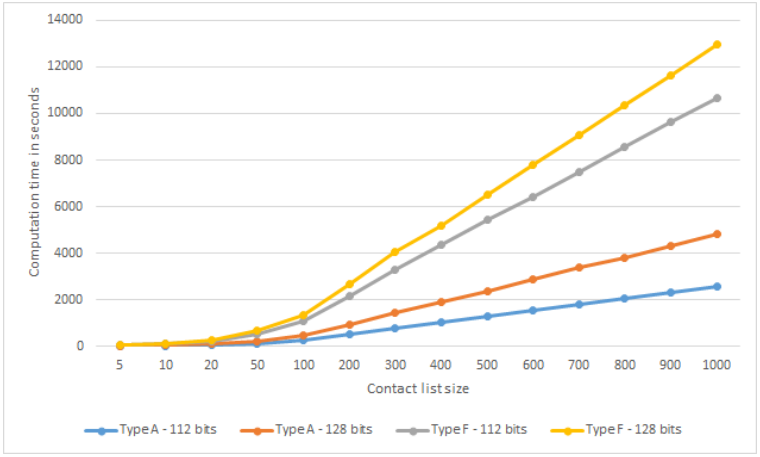
Figure 5.4: Computation Time of Batch Verification vs Naive Verification over 100 Messages

sages at once. Thus, we compare the computation time required by the naive and batch verification when being executed over 100 messages.

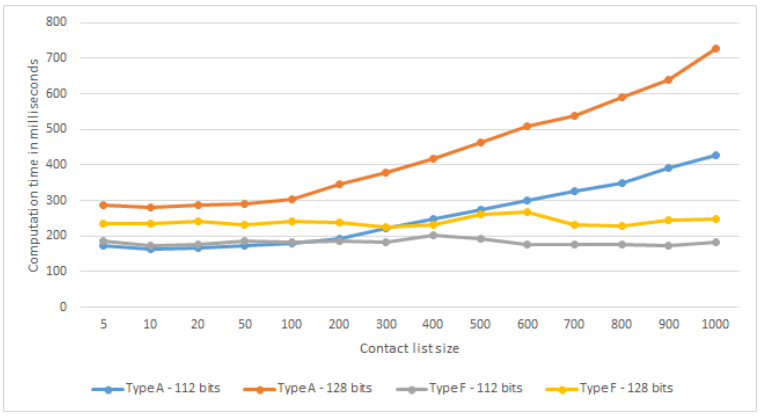
Figure 5.4 confirms that the batch verification is more efficient than the naive one. On the one hand, as depicted in Sub-Figure 5.4a, the batch verification of proofs reduces the computation time, for verifying 100 messages, by approximately 37%, for pairing *type A* for the two security levels. The computation time moves from 356 seconds (resp. 777 seconds) with the naive signature verification to 223 seconds (resp. 485 seconds) with group signatures' batch verification. For pairing *type F*, the gain reaches 50% for the two security levels. The computation time moves from 2048 seconds (resp. 2642 seconds) to 1018 seconds (resp. 1313 seconds).
The gain obtained through the proofs batch verification is substantiated by the decrease in the number of pairings. To verify N messages (i.e., 6 NIWI proofs are verified per message), the Sig_Verify algorithm requires $30N$ pairings (according to Equation 5.3), while the Batch_Sig_Verify algorithm only requires $6N + 9$ pairings. Thus, we expect to obtain a gain of approximately 80%, but experimental results show smaller gains than expected. These results are justified by the number of additions introduced while aggregating the verification equations (i.e., $14N$ additions). The elementary addition operations are more consuming for pairing *type A* than for pairing *type F*. Hence, the gain is more significant with asymmetric pairing settings.

On the other hand, Sub-Figure 5.4b shows that the batch verification over 100 partial signatures, reduces the computation time by 99% for the two types of pairings with the two different levels of security. Indeed, to verify N partial signatures, the CCM_Verify algorithm requires $2N$ exponentiations, while the Batch_CCM_Verify algorithm requires only two exponentiations and N multiplications.

Impact of Contact list Size on the Verification Referring to equations 5.5, 5.6 and 5.9, it is clear that the number N of contact messages to be verified, influences the time computation of both Batch_Sig_Verify and Batch_CCM_Verify algorithms. Indeed, the greater the number of messages, the greater the number of pairing functions, exponentiations and multiplications. For this objective, we evaluate the computation time of the Batch_Sig_Verify and Batch_CCM_Verify algorithms when varying the



(a) Batch_Sig_Verify Algorithm



(b) Batch_CCM_Verify Algorithm

Figure 5.5: Influence of Contact List Size on Batch Verification Computation Time

number of messages from 5 to 1000. Note that all contact messages are partially and fully signed with the S_PSign and P_Sign algorithms, respectively.

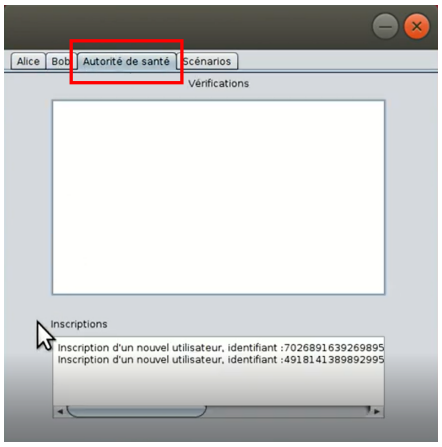
Both Figures 5.5a and 5.5b show that the computation time of both Batch_Sig_Verify and Batch_CCM_Verify algorithms is a rising affine function of the messages number, for the two types of pairings and the two security levels.

On the one hand, for the Batch_Sig_Verify algorithm, when varying the size of the contact list from 5 to 1000, the computation time varies from 15 to 2602 seconds (resp. from 59 to 10677) for the pairing *type A* (resp. pairing *type F*), for 112-bits level. For the 128-bits security, the computation time varies from 26 to 4817 seconds (resp. 72 to 12978) for the pairing *type A* (resp. pairing *type F*).

On the other hand, for the Batch_CCM_Verify algorithm, when the size of the contact list varies from 5 to 1000, the computation time, for the pairing *type A*, varies from 174 to 426 milliseconds, for the 112-bits security level (resp. from 287 to 728, for the 128-bits security level). For the pairing *type F*, the computation time is almost constant. The curve slopes are very low compared to those for pairing *type A*. This can be justified by the fact that (i) we have a constant number of exponentiations and (ii)

(a) SPOT Interface for User Alice

(b) SPOT Interface for User Bob



(c) SPOT Interface for The Health Authority

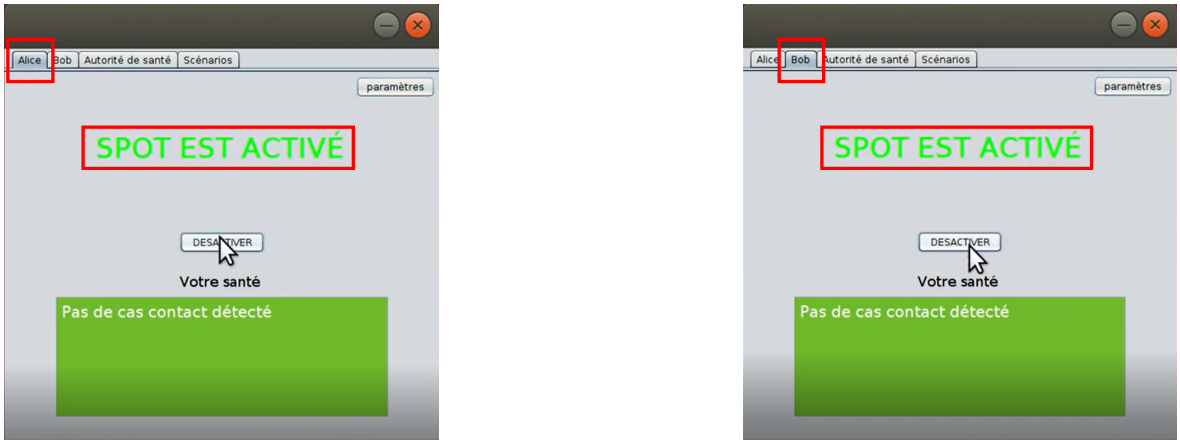
Figure 5.6: Interfaces of The SPOT Demonstrator

the elementary multiplication operations are more consuming for the pairing *type A* compared to the pairing *type F*.

5.8 SPOT Demonstrator

We have tested SPOT with its different phases and algorithms in a demonstrator. In this section, we illustrate two different scenarios (i) a contact between two users Alice and Bob, referred to as "Contact scenario" and (ii) a trial to falsify a contact list, referred to as "Forgery scenario". In this demonstrator, we describe the events that occur at three entities, namely two users Alice and Bob, and the health authority, as depicted in Figure 5.6.

Note that when we launch the demonstrator, SYS_INIT phase is automatically performed. That is, the two users Alice and Bob are registered at the health authority. To benefit from the SPOT functionalities, each user should activate the application by pressing the button "Activer", as depicted in Figure 5.7. The status of the application



(a) User Alice (b) User Bob

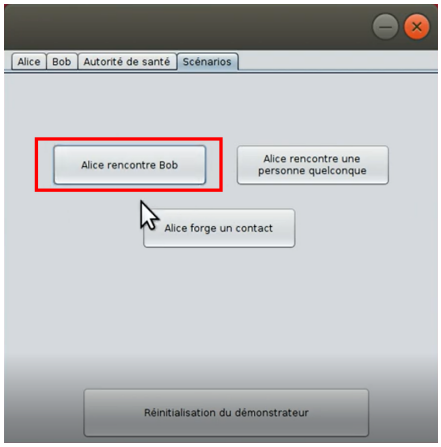
Figure 5.7: Activation of The SPOT Protocol

moves from "SPOT EST DESACTIVE" to "SPOT EST ACTIVE". As such, the user can exchange EBIDs with people in proximity.

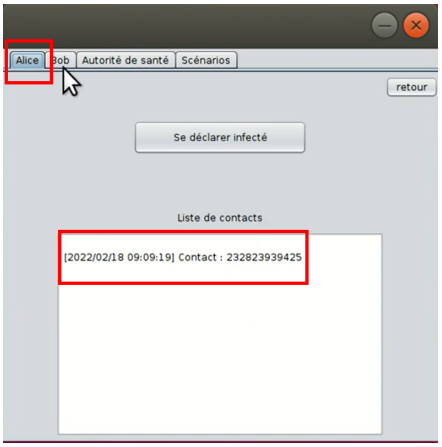
5.8.1 Contact Scenario

The GENERATION phase is initialized by pressing the button "Alice rencontre Bob" (cf. Sub-Figure 5.8a). That is, both users Alice and Bob exchange their EBIDs and perform the Set_CCM algorithm in order to compute a common contact message. Sub-Figures 5.8b and 5.8c confirm that the same contact message in computed by the two users based on their EBIDs. Note that the display of the contact message at the users' interfaces implies that all algorithms of the GENERATION phase were performed, i.e., the S_PSign algorithm is run to generate the server's partial signature and the P_Sign algorithm is performed to generate the proxy's group signature. The corresponding signatures are stored at the user's device.

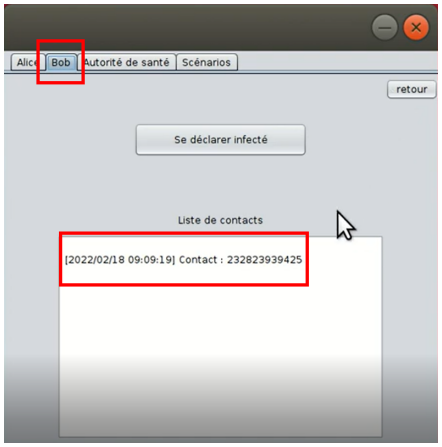
Afterwards, the VERIFICATION phase occurs when, for example, Bob is a confirmed case and reports the fact to the health authority, by pressing the button "Se déclarer infecté" in his SPOT application (cf. Sub-Figure 5.9a). Bob sends his contact list, that contains the contact message generated when being in proximity with Alice, to the health authority. As such, the health authority proceeds to a naive verification over Bob's contact list, by running the Sig_Verify and CCM_Verify algorithms, as shown in Sub-Figure 5.9b. Both verifications holds as the message has (i) been correctly generated between two users in proximity, (ii) successfully reached the server ($\mathcal{S}$) and partially signed by him, and (iii) signed by a proxy ($\mathcal{P}$).
Since the correctness of the contact message is verified by the health authority, it is shared with other users with no risque of false positive alerts. Then, Alice, being the user concerned by the contact, is notified by the SPOT application to take precautions and take actions following the government's recommendations (cf. Sub-Figure 5.9c).



(a) Simulation of a Contact



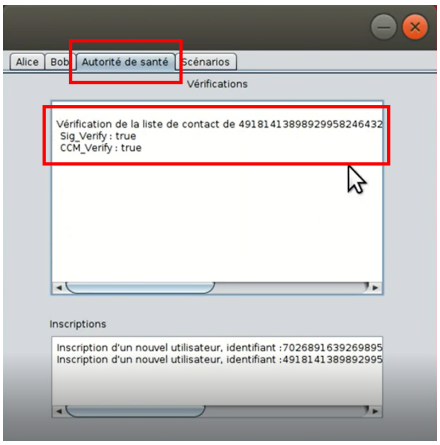
(b) Contact List of Alice



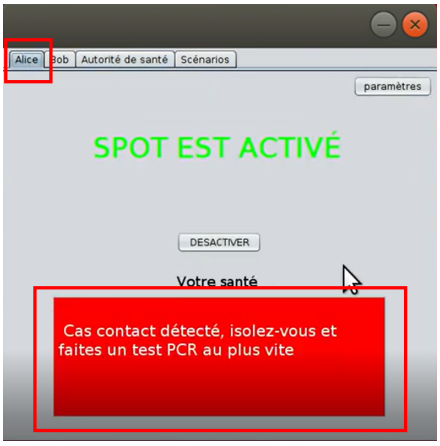
(c) Contact List of Bob

Figure 5.8: Scenario of Contact between Alice and Bob

(a) Infection Report by Bob

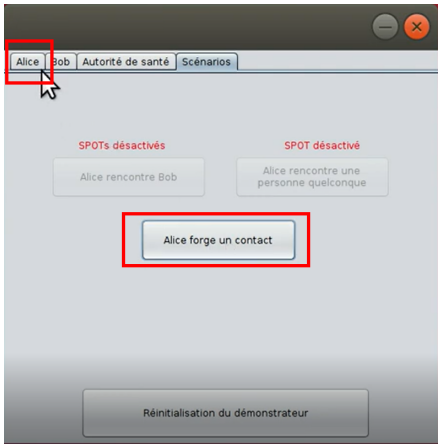


(b) Bob Contact List Verification

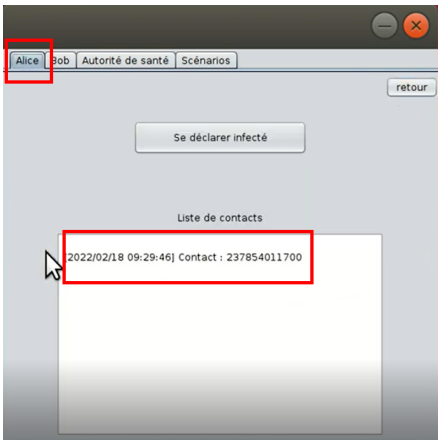


(c) Spot Notification to Alice

Figure 5.9: Verification of a Contact List in Case of Infection

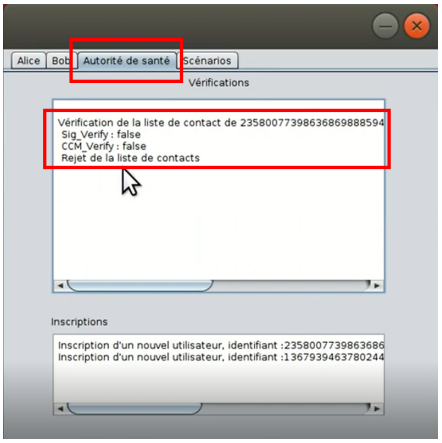


(a) Fake Contact Generation



(b) Fake Contact List

(c) Infection Report by Alice



(d) Alice Contact List Verification

Figure 5.10: A Forgery Scenario by Alice

5.8.2 Forgery Scenario

In this scenario, we consider that Alice attempts to forge his contact list and generate a fake contact message by pressing the button "Alice forge un contact" in the SPOT demonstrator (cf. Sub-Figure 5.10a). That is, Alice performs the Set_CCM algorithm on random messages of her choice. In order to generate valid signatures (i.e., the server's partial signature and a proxy's group signature) over the new contact message, Alice attempts to forge SPOT algorithms (i.e., S_PSign and P_Sign algorithms). We suppose that Alice generated a partial signature and a group signature that she stores along with the contact message, in her contact list. As depicted in Sub-Figure 5.10b, the new contact message is added to Alice contact list and is displayed on her application.

In case of infection, Alice informs the health authority by transmitting his contact list (cf. Sub-Figure 5.10c). As usual, the health authority preforms a naive verification over Alice contact list by running the Sig_Verify and CCM_Verify algorithms. As illustrated in Sub-Figure 5.10d, both verifications return "False" as the partial signature and the group signature have not been correctly generated. The health authority, then detects the forgery and rejects the fake contact message. As such, we prove that SPOT

protect its users from receiving false positive alerts.

5.9 Conclusion

In this chapter, a novel secure and privacy-preserving proximity-based Spot protocol for e-healthcare systems is introduced. The objective of Spot is to help governments and healthcare systems to deal with pandemics by automating the process of contact tracing, with security guarantees against fake contacts injection and privacy preservation for users. Thanks to the underlying network architecture relying on a centralized computing server and decentralized proxies, Spot enables users to determine whether they were in close proximity with infected people, with no risk of false positive alerts. The strength of this contribution is to provide a group signature construction that supports an efficient aggregated and batch verification over multiple proofs of knowledge. The novel group signature scheme is used as a main building block of the full concrete construction of Spot which is proven to be secure and to support several privacy properties under standard assumptions. Another strength of the contribution is a PoC of Spot including a full implementation of the different algorithms, where practical computation costs measurements demonstrate the feasibility of our proposed protocol. This implementation is tested in a demonstrator that provides two scenarios of use.

Chapter 6

Conclusions and Perspectives

We have the duty of formulating, of summarizing, and of communicating our conclusions, in intelligible form, in recognition of the right of other free minds to utilize them in making their own decisions.

– Ronald Fisher

This thesis challenged many issues in identity management systems, mainly in terms of security, privacy and usability. Our main goal was to find the best trade-off between these three requirements. In this work, three concrete identity management solutions were proposed to fulfill our research objectives when accessing services or when sharing data with different parties.

In chapter 3, we presented our first contribution about user-centric and privacy-preserving identity management to control users' access to services and resources [94]. The proposed system, named Pima, is built upon a novel malleable signature scheme. In order to fulfill *Objective A* consisting of providing users the full control over their identities and attributes, a user that receives certified attributes from a trusted identity provider, is able to select and disclose only the necessary information to service providers. In response to *Objective B*, Pima allows users to perform controlled and restricted modifications over their attributes, while service providers are still able to verify the authenticity of the data provided, through pseudonymized sessions. Different pseudonyms are used at different service providers in order to achieve *Objective D*. That is, service providers are able neither to link several transactions of the same user nor to retrieve his real identity.

In chapter 4, we proposed Puba, a privacy-preserving biometric authentication scheme for identity management systems that provides a physical control access to services and resources. To anonymously enroll at a service provider, the user is able to

modify encrypted and certified biometric templates, received from an identity provider, in response to *Objective A*. For authentication, the user is asked to provide a fresh biometric template when being present at the service provider's desk. The service provider, then, computes with the help of the user, a matching distance between the two templates. In order to fulfill *Objective B*, PUBA's design enables to verify the validity of users' biometric data and the correctness of users' computations over matching distances. We relied on polymorphic and randomizability features to prevent service providers from either linking several enrollment sessions of the same user or inferring users' real identities, in response to *Objective D*.

In order to fulfill *Objective F* which consists of implementing the proposed schemes and validating their feasibility on real hardware, experimental results have demonstrated the efficiency of PIMA and PUBA solutions. On the one hand, algorithms performed by either the identity provider or the service provider were tested on a laptop with acceptable hardware capacities. On the other hand, tests of algorithms run by the user, have been performed on a resource-constrained device, i.e., a smartphone. On both devices, experimental results have shown the practical usability of PIMA and PUBA with reasonable computation times.

In Chapter 5, we proposed SPOT, a privacy-preserving proximity-tracing protocol for e-healthcare systems. In response to *Objective C*, which consists of designing a secure mechanism for ephemeral identities management, SPOT is designed to support secure and privacy-preserving ephemeral contact information sharing among different parties, through an hybrid architecture. In order to fulfill *objective A*, the user is first responsible for computing contact messages, when being in proximity with other users. Second, while relying on a centralized server and decentralized proxies, the user collects certified contact information that he locally stored. Then, in case of infection, the user shares these information with the health authority, which verifies their correctness before being sent to other users. The security of SPOT resides in preventing the injection of false positive alerts, while the privacy is considered against linkability and identification attacks, which complies with *Objective D*. An implementation of the proposed protocol with an effort to aggregate computations, has demonstrated the efficiency of SPOT. This implementation was tested in a demonstrator through two scenarios, in response to *Objective F*.

Objective E consists of providing proofs of the security and privacy properties of the proposed identity management solutions. To achieve this objective, proofs are presented for each proposed scheme, relying on standard computational assumptions, with respect to detailed security games. First, we demonstrate the unforgeability of each scheme while considering a malicious user that attempts to generate valid credentials, by himself or when colluding with other parties. Furthermore, while considering colluding curious service providers, the three contributions are proven to be resistant against the linkability between transactions of the same user.
In addition, while considering a collusion between an identity provider and one or several service providers against the anonymity property, we demonstrated that PUBA resists to re-identification attacks. The anonymity of users involved in a contact list with an infected user is also proven to be fulfilled by SPOT.

Perspectives

Regarding the increasing need to digital identity and the multitude of deployed identity management solutions, further perspectives of interest should be addressed in terms of performances and interoperability between identity solutions. These perspectives include:

- proposing an extension of Pima where multiple identity providers are considered. That is, a user is able to prove to a service provider different attributes certified by different identity providers. Thus, it would be interesting to investigate the user's capacities to aggregate several attributes provided by several identity providers under the same pseudonym. Two ways could be explored and evaluated with respect to security, privacy and performance requirements. The first way consists of aggregating multiple malleable signatures generated by different identity providers to the same user. Thus, it would be important to define the key(s) used by service providers to verify the authenticity of users' attributes. The second way considers that all identity providers belong to the same group of signers. As such, a malleable group signature scheme could be proposed. It would be also interesting to study the possibility of aggregating several malleable group signatures, such that the service provider verifies the authenticity of users' attributes in a single transactions using the group public key. The impact of the two proposed ways on users' privacy deserves to be studied, namely unlinkability among both service providers and identity providers.

- extending Pima to a user-centric identity management system where the governance is decentralized. Indeed, it would be interesting to study the use of the $\mathcal{UMS}$ scheme in a distributed environment like the blockchain.

- inspiring from Puba to design a new solution that supports online verification of a biometric authentication, with respect to European and international legislations. Indeed, regarding the increasing need to prove the link to the physical person in order to avoid frauds, it would be interesting to propose secure, reliable and privacy-preserving biometric authentication mechanism without the need to be present at the service provider. It would be also important to consider biometric credentials auditing and revocation, by a trusted authority, to reveal the identity of users responsible for malicious behaviors and revoke certificates for example in case of identity thefts.

- proposing a solution that ensures the link between users' attributes and their biometric identities, relying on the Puba biometric authentication. That is, it would be interesting, to prove the authenticity of a user's attributes when being associated to his biometric identity which has been verified during authentication. The possibility of aggregating, under the same biometric identity, multiple attributes delivered by different attribute providers, could be also investigated.

- performing additional experimental evaluations of Spot on devices with advanced hardware capacities. Indeed, it would be interesting to evaluate the order of

magnitude of some system's parameters. For instance, the time spent by the server to wait the reception of two copies of the same contact message, could be evaluated with respect to the number of both accepted connections by the server and requested connections.

- evaluating SPOT communication costs, while varying parameters like the distance between the user and the proxy responsible for transmitting his contact information. As such, we are able to evaluate the size of a geographical area and the number of proxies required.

In conclusion, this thesis was an opportunity to understand identity management requirements with the aim of proposing concrete solutions that comply with new regulations. With the rapid development of trends like Internet of Things and Big Data, and the new forms of users' identities and identifiable information they introduce, we believe that individuals will no more rely on traditional methods of authentication. Artificial Intelligence (AI) will come on, in the next few years, to authenticate users according to their profiles, while combining several users' information and analyzing their behaviors. AI will be also used to ensure systems' security, for example by predicting and detecting malicious behaviors. Hence, new relevant challenges for identity management systems may arise in the few coming years, resulting in new research problems.

Glossary of Acronyms

ABC	Attribute-based Credentials
AC	Anonymous Credentials
AI	Artificial Intelligence
BLE	Bluetooth Low Energy
CCPA	California Consumer Privacy Act
CDH	Computational Diffie Hellman
CPRA	California Privacy Rights Act
CRS	Common Reference String
DAA	Direct Anonymous Attestation
DoS	Denial of Service
DSA	Digital Signature Algorithm
EBID	Ephemeral Bluetooth IDentifier
EHR	Electronic Healthcare Records
EUF-CMA	Existential Unforgeability under Chosen Message Attacks
FHE	Fully Homomorphic Encryption
FTC	Federal Trade Commission
GDPR	General Data Protection Regulation
IAM	Identity and Access Management
IDM	Identity Management
IoT	Internet of Things
IPFS	InterPlanetary File System
LRSW	Lysyanskaya-Rivest-Sahai-Wolf
ML	Machine Learning
NIST	National Institute of Standards and Technology
NIWI	Non-Interactive Witness-Indistinguishable
OASIS	Organization for the Advancement of Structured Information Standards
PET	Privacy-Enhancing Technology
PHE	Partially Homomorphic Encryption
PKI	Public Key Infrastructure
PoK	Proof of Knowledge
PPT	Probabilistic-Polynomial Time
PS	Pointcheval-Sanders
QR	Quick Response
RSA	Rivest–Shamir–Adleman
SAML	Security Assertion Markup Language
SHE	Somewhat Homomorphic Encryption
SSO	Single Sign-On
TPE	Threshold Predicate Encryption
URL	Uniform Resource Locator
VAT	Value-Added Tax
VC	Verifiable Credentials
WIFI	WIreless FIdelity
XRI	EXtensible Resource Identifier

Author's Publications

Peer-reviewed Publications

- Souha Masmoudi, Maryline Laurent and Nesrine Kaaniche, *SPOT+: Secure and Privacy-preserving Proximity-tracing Protocol with Efficient Verification over Multiple Contact Information*, (Submitted), 2022.

- Souha Masmoudi, Maryline Laurent and Nesrine Kaaniche, *PUBA: Privacy-preserving and User-centric Biometric Authentication through Malleable Signatures*, (Submitted), 2022.

- Souha Masmoudi, Nesrine Kaaniche and Maryline Laurent, *SPOT: Secure and Privacy-preserving Proximity-tracing Protocol for E-healthcare Systems*, IEEE Open Access Journal, October 2022.

- Souha Masmoudi, Maryline Laurent and Nesrine Kaaniche, *PIMA: A privacy-preserving identity management system based on an unlinkable MAlleable signature*, Journal of Network and Computer Applications, September 2022.

- Souha Masmoudi, Maryline Laurent and Nesrine Kaaniche, *SEVIL: Secure and Efficient VerifIcation over Massive Proofs of KnowLedge*, 19th International Conference on Security and Cryptography SECRYPT, July 2022.

- Seryne Rahali, Maryline Laurent, Souha Masmoudi, Charles Roux and Brice Mazeau, *A Validated Privacy-Utility Preserving Recommendation System with Local Differential Privacy*, IEEE 15th International Conference on Big Data Science and Engineering (BigDataSE 2021), August 2021.

- Nesrine Kaaniche, Souha Masmoudi, Souha Znina, Maryline Laurent, and Levent Demir, *Privacy preserving cooperative computation for personalized web search applications*, Proceedings of the 35th Annual ACM Symposium on Applied Computing (SAC '20), March 2020.

Publications Without Review Committee

- Souha Masmoudi and Maryline Laurent, *Un nouveau système de recommandations conjuguant le respect de la vie privée et l'utilité*, Letter N°22 of the Chair Values and Policies of Personal Information (VP-IP), October 2021. de la recommandation

- Souha Masmoudi and Maryline Laurent, *Les Pays-Bas : pionniers dans le domaine de la cybersanté*, Letter N°20 of the Chair Values and Policies of Personal Information (VP-IP), March 2021.

- Souha Masmoudi and Maryline Laurent, *Une gestion des identités placée sous le contrôle des individus : une certification sous pseudonymat*, Letter N°19 of the Chair Values and Policies of Personal Information (VP-IP), December 2020.
- Souha Masmoudi and Maryline Laurent, *Better management and control over multiple digital identities: State of the art*, provided to the Chair Values and Policies of Personal Information (VP-IP), May 2020.
- Souha Masmoudi and Maryline Laurent, *Travaux de la Chaire sur la personnalisation de services respectueux de la vie privée*, Letter N°17 of the Chair Values and Policies of Personal Information (VP-IP), April 2020.
- Maryline Laurent, Nesrine Kaaniche and Souha Masmoudi, *Un service de recherche web coopératif fournissant un service personnalisé respectueux de la vie privée*, Letter N°15 of the Chair Values and Policies of Personal Information (VP-IP), September 2019.

Bibliography

[1] Patrick Waelbroeck, Armen Khatchatourov, and Claire Levallois-Barth. *Personal Data and Trust: What are the strategies of french citizen-consumers in 2017?* https://cvpip.wp.imt.fr/files/2017/08/Personal-Data-and-Trust-Synthesis.pdf. 2017.

[2] Masayuki Abe, Kristiyan Haralambiev, and Miyako Ohkubo. "Signing on Elements in Bilinear Groups for Modular Protocol Design." In: *IACR Cryptology ePrint Archive* (2010).

[3] Aysajan Abidin et al. "Efficient Verifiable Computation of XOR for Biometric Authentication". In: *Cryptology and Network Security*. 2016. DOI: 10.1007/978-3-319-48965-0_17.

[4] Abdulrahman Alamer. "An efficient group signcryption scheme supporting batch verification for securing transmitted data in the Internet of Things". In: *Journal of Ambient Intelligence and Humanized Computing* (June 2020). DOI: 10.1007/s12652-020-02076-x.

[5] Gergely Alpár. *Attribute-based identity management : Bridging the cryptographic design of ABCs with the real world*. https://hdl.handle.net/2066/135177. 2015.

[6] Gergely Alpar, Jaap-Henk Hoepman, and Johanneke Siljee. "The Identity Crisis. Security, Privacy and Usability Issues in Identity Management". In: *CoRR* 9 (Jan. 2011).

[7] Apple Inc. and Google Inc. *Contact Tracing, Bluetooth Specification*. https://www.blog.google/documents/58/Contact_Tracing_-_Bluetooth_. 2020.

[8] Will Arthur and David Challener. *A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security*. USA: Apress, 2015. ISBN: 1430265833.

[9] Giuseppe Ateniese and Gene Tsudik. "Group Signatures a La Carte". In: *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '99. USA: Society for Industrial and Applied Mathematics, 1999, 848–849. ISBN: 0898714346.

[10] Giuseppe Ateniese et al. "A Practical and Provably Secure Coalition-Resistant Group Signature Scheme". In: *Advances in Cryptology — CRYPTO 2000*. Aug. 2000, pp. 255–270. ISBN: 978-3-540-67907-3. DOI: 10.1007/3-540-44598-6_16.

[11] Giuseppe Ateniese et al. "Sanitizable Signatures". In: *Computer Security – ESORICS 2005*. Springer, Berlin, Heidelberg, 2005, pp. 159–177.

[12] Australian Government Department of Health. *COVIDSafe app*. https://www.health.gov.au/resources/apps-and-tools/covidsafe-app. 2020.

[13] Vidhi Bansal and Surabhi Garg. "A cancelable biometric identification scheme based on bloom filter and format-preserving encryption". In: *Journal of King Saud University - Computer and Information Sciences* (2022). DOI: 10.1016/j.jksuci.2022.01.014.

[14] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. "Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions". In: *Advances in Cryptology — EUROCRYPT 2003*. Vol. 2656. May 2003, pp. 614–629. ISBN: 978-3-540-14039-9. DOI: 10.1007/3-540-39200-9_38.

[15] Mihir Bellare, Haixia Shi, and Chong Zhang. "Foundations of Group Signatures: The Case of Dynamic Groups". In: *Topics in Cryptology – CT-RSA 2005*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 136–153.

[16] Patrik Bichsel et al. "D2.2 Architecture for attribute-based credential technologies-final version". In: *ABC4TRUST project deliverable* (2014).

[17] Arne Bilzhause, Henrich Christopher Pöhls, and Kai Samelin. "Position Paper: The Past, Present, and Future of Sanitizable and Redactable Signatures". In: *Proceedings of the 12th International Conference on Availability, Reliability and Security* (2017), pp. 1–9.

[18] Eleanor Birrell and Fred B. Schneider. "Federated Identity Management Systems: A Privacy-Based Characterization". In: *IEEE Security & Privacy* 11.5 (2013), pp. 36–48. DOI: 10.1109/MSP.2013.114.

[19] Dan Boneh et al. "Private Database Queries Using Somewhat Homomorphic Encryption". In: *Applied Cryptography and Network Security*. Vol. 7954. June 2013, pp. 102–118. ISBN: 978-3-642-38979-5. DOI: 10.1007/978-3-642-38980-1_7.

[20] Angèle Bossuat and Xavier Bultel. "Unlinkable and Invisible γ-Sanitizable Signatures". In: *Applied Cryptography and Network Security*. Cham: Springer International Publishing, 2021, pp. 251–283. ISBN: 978-3-030-78372-3.

[21] Samia Bouzefrane, Khaled Garri, and Pascal Thoniel. "A user-centric PKI based-protocol to manage FC² digital identities". In: *International Journal of Computer Science Issues* 8 (2011).

[22] Ernie Brickell and Jiangtao Li. "Enhanced Privacy Id: A Direct Anonymous Attestation Scheme with Enhanced Revocation Capabilities". In: *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society*. New York, NY, USA: Association for Computing Machinery, 2007, 21–30. DOI: 10.1145/1314333.1314337.

[23] Christina Brzuska, Henrich C. Pöhls, and Kai Samelin. "Efficient and Perfectly Unlinkable Sanitizable Signatures without Group Signatures". In: *Public Key Infrastructures, Services and Applications*. Springer, Berlin, Heidelberg, 2014, pp. 12–30.

[24] Christina Brzuska et al. “Redactable Signatures for Tree-Structured Data: Definitions and Constructions”. In: *Applied Cryptography and Network Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 87–104.

[25] Christina Brzuska et al. “Security of Sanitizable Signatures Revisited”. In: *Public Key Cryptography – PKC 2009*. Springer, Berlin, Heidelberg, 2009, pp. 317–336.

[26] Christina Brzuska et al. “Unlinkability of Sanitizable Signatures”. In: *Public Key Cryptography – PKC 2010*. Springer, Berlin, Heidelberg, 2010, pp. 444–461.

[27] Xavier Bultel et al. “Efficient Invisible and Unlinkable Sanitizable Signatures”. In: *Public-Key Cryptography – PKC 2019*. Cham: Springer International Publishing, 2019, pp. 159–189.

[28] *CA Notify*. https://canotify.uchealth.edu/ucberkeley/.

[29] Jan Camenisch and Anna Lysyanskaya. “Signature schemes and anonymous credentials from bilinear maps”. In: *Annual International Cryptology Conference - CRYPTO 2004* (2004), pp. 56–72.

[30] Jan Camenisch and Els Van Herreweghen. “Design and Implementation of the Idemix Anonymous Credential System”. In: *CCS 2002*. New York, NY, USA: Association for Computing Machinery, 2002, 21–30. ISBN: 1581136129.

[31] Jan Camenisch et al. “Composable and Modular Anonymous Credentials: Definitions and Practical Constructions”. In: *Advances in Cryptology – ASIACRYPT 2015*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 262–288.

[32] Technology Analysis Division of the Office of the Privacy Commissioner of Canada. *Privacy Enhancing Technologies – A Review of Tools and Techniques*. Nov. 2017.

[33] Sébastien Canard, Amandine Jambert, and Roch Lescuyer. “Sanitizable Signatures with Several Signers and Sanitizers”. In: *Progress in Cryptology - AFRICACRYPT 2012*. Springer, Berlin, Heidelberg, 2012, pp. 35–52.

[34] Sébastien Canard and Roch Lescuyer. “Protecting privacy by sanitizing personal data: A new approach to anonymous credentials”. In: *ASIA CCS 2013* (2013), pp. 381–392.

[35] Yuan Cao and Lin Yang. “A survey of Identity Management technology”. In: *2010 IEEE International Conference on Information Theory and Information Security*. 2010, pp. 287–293. DOI: 10.1109/ICITIS.2010.5689468.

[36] Troncoso Carmela et al. *Decentralized privacy-preserving proximity tracing*. https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf. 2020.

[37] Justin Chan et al. “PACT: Privacy-Sensitive Protocols And Mechanisms for Mobile Contact Tracing”. In: *arXiv:2004.03544* (2020).

[38] Donghoon Chang et al. “BIOFUSE: A framework for multi-biometric fusion on biocryptosystem level”. In: *Information Sciences* 546 (2021), pp. 481–511. DOI: 10.1016/j.ins.2020.08.065.

[39] Donghoon Chang et al. “Cancelable Multi-Biometric Approach Using Fuzzy Extractor and Novel Bit-Wise Encryption”. In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 3152–3167. DOI: 10.1109/TIFS.2020.2983250.

[40] Barton Chao and Lin Yang. “A survey of Identity Management technology”. In: *Proceedings 2010 IEEE International Conference on Information Theory and Information Security, ICITIS 2010* (2011), pp. 287 –293.

[41] M. Chase et al. “Malleable Signatures: New Definitions and Delegatable Anonymous Credentials”. In: *2014 IEEE 27th Computer Security Foundations Symposium*. 2014, pp. 199–213.

[42] David Chaum and Eugène van Heyst. “Group Signatures”. In: *Advances in Cryptology — EUROCRYPT ’91*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 257–265.

[43] L. T. Chean, V. Ponnusamy, and S. M. Fati. “Authentication scheme using unique identification method with homomorphic encryption in Mobile Cloud Computing”. In: *2018 IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE)*. 2018, pp. 195–200.

[44] L. Chen and T. P. Pedersen. “New group signature schemes”. In: *Advances in Cryptology — EUROCRYPT’94*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 171–181.

[45] Sherman S. M. Chow et al. “SPICE – Simple Privacy-Preserving Identity-Management for Cloud Environment”. In: *Applied Cryptography and Network Security*. Springer, Berlin, Heidelberg, 2012, pp. 526–543.

[46] Castelluccia Claude et al. *DESIRE: A Third Way for a European Exposure Notification System*. https://github.com/3rd-ways-for-EU-exposure-notification/project-DESIRE/blob/master/DESIRE-specification-EN-v1_0.pdf. 2020.

[47] CloudTweaks. *Infographic: How much data is produced every day?* https://cloudtweaks.com/2015/03/how-much-data-is-produced-every-day/. 2021.

[48] Federal Trade Commission. *FTC COVID-19 an Stimulus Reports*. https://public.tableau.com/app/profile/federal.trade.commission/viz/COVID-19andStimulusReports/FraudLosses. 2022.

[49] Sam Cook. *Identity theft facts & statistics: 2019-2022*. https://www.comparitech.com/identity-theft-protection/identity-theft-statistics/. 2022.

[50] Ivan Damgard et al. “Multiparty Computation from Somewhat Homomorphic Encryption”. In: *IACR Cryptology ePrint Archive* 2011 (Jan. 2011), p. 535. DOI: 10.1007/978-3-642-32009-5_38.

[51] Cécile Delerablée and David Pointcheval. “Dynamic Fully Anonymous Short Group Signatures”. In: *Progress in Cryptology - VIETCRYPT 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 193–210.

[52] Google+ Developers. *Introducing Google+ Sign-In: simple and secure, minus the social spam*. http://googleplusplatform.blogspot.com/2013/02/google-plus-sign-in.html. 2013.

[53] Jean-Guillaume Dumas et al. “LocalPKI: An Interoperable and IoT Friendly PKI”. In: *E-Business and Telecommunications*. Cham: Springer International Publishing, 2019, pp. 224–252.

[54] T. El Maliki and J. Seigneur. “A Survey of User-centric Identity Management Technologies”. In: *The International Conference on Emerging Security Information, Systems, and Technologies (SECUREWARE 2007)*. 2007, pp. 12–17.

[55] T. Elgamal. “A public key cryptosystem and a signature scheme based on discrete logarithms”. In: *IEEE Transactions on Information Theory* 31.4 (1985), pp. 469–472. DOI: 10.1109/TIT.1985.1057074.

[56] Keita Emura and Takuya Hayashi. “A Revocable Group Signature Scheme with Scalability from Simple Assumptions and Its Application to Identity Management”. In: *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* 103-A (2019), pp. 125–140.

[57] Council Europe. “Proposal for a Regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data”. In: *General Data Protection Regulation*. 2016.

[58] Farnaz Farid et al. “A Smart Biometric Identity Management Framework for Personalised IoT and Cloud Computing-Based Healthcare Services”. In: *Sensors* 21.2 (2021). DOI: 10.3390/s21020552.

[59] Wei Feng, Zheng Yan, and Haomeng Xie. “Anonymous Authentication on Trust in Pervasive Social Networking Based on Group Signature”. In: *IEEE Access* 5 (2017), pp. 6236–6246. DOI: 10.1109/ACCESS.2017.2679980.

[60] Anna Ferrara et al. “Practical Short Signature Batch Verification”. In: *Topics in Cryptology – CT-RSA 2009*. Apr. 2009, pp. 309–324. ISBN: 978-3-642-00861-0. DOI: 10.1007/978-3-642-00862-7_21.

[61] Nils Fleischhacker et al. “Efficient Unlinkable Sanitizable Signatures from Signatures with Re-randomizable Keys”. In: *Public-Key Cryptography – PKC 2016*. Springer, Berlin, Heidelberg, 2016, pp. 301–330.

[62] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. "A Digital Signature Scheme Secure against Adaptive Chosen-Message Attacks". In: *SIAM J. Comput.* 17.2 (1988), 281–308. ISSN: 0097-5397. DOI: 10.1137/0217017.

[63] Marta Gomez-Barrero et al. "Multi-biometric template protection based on Homomorphic Encryption". In: *Pattern Recognition* 67 (2017), pp. 149–163. DOI: 10.1016/j.patcog.2017.01.024.

[64] Jens Groth and Amit Sahai. "Efficient Non-interactive Proof Systems for Bilinear Groups". In: *Advances in Cryptology – EUROCRYPT 2008*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 415–432.

[65] Hasini Gunasinghe and Elisa Bertino. "PrivBioMTAuth: Privacy Preserving Biometrics-Based and User Centric Protocol for User Authentication From Mobile Phones". In: *IEEE Transactions on Information Forensics and Security* 13.4 (2018), pp. 1042–1057. DOI: 10.1109/TIFS.2017.2777787.

[66] Asghar Hassan et al. *On the privacy of TraceTogether, the Singaporean COVID-19 contact tracing app and recommendations for Australia*. 2020.

[67] Jaap-Henk Hoepman. "Hansel and Gretel and the Virus: Privacy Conscious Contact Tracing". In: *arXiv preprint arXiv:2101.03241* (2021).

[68] Hai Huang and Luyao Wang. "Efficient privacy-preserving face verification scheme". In: *Journal of Information Security and Applications* 63 (2021), p. 103055. ISSN: 2214-2126.

[69] Alberto Ibarrondo, Hervé Chabanne, and Melek Önen. "Practical Privacy-Preserving Face Identification Based on Function-Hiding Functional Encryption". In: *Cryptology and Network Security*. Cham: Springer International Publishing, 2021, pp. 63–71.

[70] Inria and Fraunhofer AISEC. *ROBERT: Robust and privacy-preserving proximity tracing*. https://github.com/ROBERT-proximity-tracing/documents/blob/master/ROBERT-specification-EN-v1_1.pdf. 2020, [Online accessed June 2022].

[71] iProov. *70 Biometric Statistics*. https://www.iproov.com/blog/biometric-statistics-70. 2022.

[72] ISO/IEC29100. *International Standard, Information technology - Security techniques - Privacy framework*. Dec. 2011.

[73] Toshiyuki Isshiki et al. "Using group signatures for identity management and its implementation". In: *DIM '06*. 2006.

[74] K. Liu Joseph et al. "Privacy-Preserving COVID-19 Contact Tracing App: A Zero-Knowledge Proof Approach". In: *IACR Cryptol. ePrint Arch, 2020* 528 (2020).

[75] *JPBC Library: Bilinear Pairing Parameters Generators*. http://gas.dia.unisa.it/projects/jpbc/docs/ecpg.html.

[76] Audun Jøsang and Simon Pope. "User centric identity management". In: *AusCERT Conference 2005* (Jan. 2005).

[77] Audun Jøsang et al. "Local user-centric identity management". In: *Journal of Trust Management* 2 (Dec. 2015). DOI: 10.1186/s40493-014-0009-6.

[78] Nesrine Kaaniche and Maryline Laurent. "Attribute-Based Signatures for Supporting Anonymous Certification". In: *ESORICS 2016*. Springer, 2016, pp. 279–300. ISBN: 978-3-319-45744-4.

[79] Armen Khatchatourov and Pierre-Antoine Chardel. *Identités numériques / identités multiples*. https://cvpip.wp.imt.fr/2016/03/04/identites-numeriques-identites-multiples/. 2016.

[80] Aggelos Kiayias and Moti Yung. "Group Signatures with Efficient Concurrent Join". In: *Advances in Cryptology – EUROCRYPT 2005*. Vol. 2005. May 2005, p. 345. ISBN: 978-3-540-25910-7. DOI: 10.1007/11426639_12.

[81] Jinsu Kim et al. "Efficient Privacy-Preserving Matrix Factorization for Recommendation via Fully Homomorphic Encryption". In: *ACM Trans. Priv. Secur.* 21.4 (2018). ISSN: 2471-2566.

[82] Kitae Kim et al. "Batch Verification and Finding Invalid Signatures in a Group Signature Scheme". In: *International Journal of Network Security* 12 (Apr. 2011), pp. 229–238.

[83] D. Nancy Kirupanithi and A. Antonidoss. "Self-Sovereign Identity Management System on blockchain based applications using Chameleon Hash". In: *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*. 2021, pp. 386–390.

[84] Mahesh Kumar, Munaga Prasad, and U. Raju. "Privacy-preserving iris authentication using fully homomorphic encryption". In: *Multimedia Tools and Applications* 79 (July 2020). DOI: 10.1007/s11042-020-08680-5.

[85] Russell W. F. Lai et al. "Efficient Sanitizable Signatures Without Random Oracles". In: *Computer Security – ESORICS 2016*. Cham: Springer International Publishing, 2016, pp. 363–380.

[86] California State Legislature. "California Code, Civil COde - CIV § 1798.100-192". In: *California Consumer Privacy Act*. 2018.

[87] California State Legislature. "California Code, Civil COde - CIV § 1798.199.10". In: *California Privacy Rights Act*. 2020.

[88] Douglas J Leith and Stephen Farrell. "Contact tracing app privacy: What data is shared by Europe's GAEN contact tracing apps". In: *IEEE INFOCOM 2021*. IEEE. 2021, pp. 1–10.

[89] Song Li et al. "Redactable Signature-Based Public Auditing Scheme With Sensitive Data Sharing for Cloud Storage". In: *IEEE Systems Journal* 16.3 (2022), pp. 3613–3624. DOI: 10.1109/JSYST.2022.3159832.

[90] Ying Luo, Sen-ching S. Cheung, and Shuiming Ye. "Anonymous Biometric Access Control based on homomorphic encryption". In: *2009 IEEE International Conference on Multimedia and Expo*. 2009, pp. 1046–1049. DOI: 10.1109/ICME.2009.5202677.

[91] Yong Ma et al. "Research Review on the Application of Homomorphic Encryption in Database Privacy Protection". In: *International Journal of Cognitive Informatics and Natural Intelligence* 15 (Oct. 2021), pp. 1–22. DOI: 10.4018/IJCINI.287600.

[92] Lukas Malina et al. "Secure electronic voting based on group signatures". In: *2015 38th International Conference on Telecommunications and Signal Processing (TSP)*. 2015, pp. 6–10.

[93] Souha Masmoudi, Nesrine Kaaniche, and Maryline Laurent. ""SPOT: Secure and Privacy-preserving Proximity-tracing Protocol for E-healthcare Systems"". In: *IEEE Access*. 2022. DOI: 10.1109/ACCESS.2022.3208697.

[94] Souha Masmoudi, Maryline Laurent, and Nesrine Kaaniche. "PIMA: A privacy-preserving identity management system based on an unlinkable MAlleable signature". In: *Journal of Network and Computer Applications* (2022), p. 103517. ISSN: 1084-8045. DOI: 10.1016/j.jnca.2022.103517.

[95] Souha Masmoudi, Maryline Laurent, and Nesrine Kaaniche. *"PUBA : Privacy-preserving and User-centric Biometric Authentication through Malleable Signatures*. Under submission. 2022.

[96] Souha Masmoudi., Maryline Laurent., and Nesrine Kaaniche. "SEVIL: Secure and Efficient VerifIcation over Massive Proofs of KnowLedge". In: *Proceedings of the 19th International Conference on Security and Cryptography - SECRYPT*. INSTICC. SciTePress, 2022, pp. 13–24. ISBN: 978-989-758-590-6. DOI: 10.5220/0011125800003283.

[97] Md. Faruk Hossain Mazumder and Mohammad Amdad Ullah Chowdhury. *Windows CardSpace*. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.132.6635&rep=rep1&type=pdf. 2007.

[98] Bhabendu Mohanta and Debasis Gountia. "Fully homomorphic encryption equating to cloud security: An approach". In: *IOSR Journal of Computer Engineering* 9 (2013), pp. 46–50.

[99] Mahesh Kumar Morampudi, Munaga V. N. K. Prasad, and U. S. N. Raju. "Privacy-Preserving and Verifiable Multi-Instance Iris Remote Authentication Using Public Auditor". In: *Applied Intelligence* 51.10 (2021), 6823–6836. DOI: 10.1007/s10489-021-02187-8.

[100] Dave Morin. *Announcing Facebook Connect*. https://developers.facebook.com/blog/post/2008/05/09/announcing-facebook-connect/. 2008.

[101] Kundan Munjal and Rekha Bhatia. "A systematic review of homomorphic encryption and its contributions in healthcare industry". In: *Complex & Intelligent Systems* (2022). DOI: 10.1007/s40747-022-00756-z.

[102] Montreal Holocaust Museum. *Abolishing online anonymity won't tackle the underlying problems of racist abuse*. https://www.theguardian.com/commentisfree/2021/jul/15/abolishing-online-anonymity-racist-abuse-id-verification. 2019.

[103] David Naccache et al. "Can D.S.A. be Improved? Complexity Trade-Offs with the Digital Signature Standard". In: *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994*. Lecture Notes in Computer Science. Springer, 1994, pp. 77–85. DOI: 10.1007/BFb0053426.

[104] Aimee O'driscoll. *25+ Password statistics (that may change your password habits)*. https://www.comparitech.com/blog/information-security/password-statistics/. 2022.

[105] OECD. *Electronic Authentication and OECD Guidance for Electronic Authentication*. http://www.oecd.org/internet/ieconomy/38921342.pdf. 2007.

[106] Taiwo Blessing Ogunseyi, Cossi Blaise Avoussoukpo, and Yiqiang Jiang. "Privacy-Preserving Matrix Factorization for Cross-Domain Recommendation". In: *IEEE Access* 9 (2021), pp. 91027–91037.

[107] Pascal Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes". In: *Advances in Cryptology — EUROCRYPT '99*. Vol. 5. May 1999, pp. 223–238. ISBN: 978-3-540-65889-4. DOI: 10.1007/3-540-48910-X_16.

[108] Christian Paquin. *U-Prove Technology Overview V1.1*. http://research.microsoft.com/apps/pubs/default.aspx?id=166980. 2013.

[109] Jarosław Pastuszak et al. "Identification of Bad Signatures in Batches". In: *Public Key Cryptography*. Mar. 2004, pp. 28–45. ISBN: 978-3-540-66967-8. DOI: 10.1007/978-3-540-46588-1_3.

[110] Vishal M. Patel, Nalini K. Ratha, and Rama Chellappa. "Cancelable Biometrics: A review". In: *IEEE Signal Processing Magazine* 32.5 (2015), pp. 54–65. DOI: 10.1109/MSP.2015.2434151.

[111] Chaire Valeurs et politiques des informations personnelles. *Identités numériques*. Cahier n° 1 «Identités numériques» coordonné par Claire Levallois-Barth. Mar. 2016.

[112] Krzysztof Pietrzak. "Delayed Authentication: Preventing Replay and Relay Attacks in Private Contact Tracing". In: *Progress in Cryptology – INDOCRYPT 2020*. Cham: Springer International Publishing, 2020, pp. 3–15.

[113] David Pointcheval and Olivier Sanders. “Short Randomizable Signatures”. In: *Topics in Cryptology - CT-RSA 2016*. Cham: Springer International Publishing, 2016, pp. 111–126.

[114] Manish M. Potey, C.A. Dhote, and Deepak H. Sharma. “Homomorphic Encryption for Security of Cloud Data”. In: *Procedia Computer Science* 79 (2016), pp. 175 –181.

[115] Gaëtan Pradel and Chris Mitchell. “Privacy-Preserving Biometric Matching Using Homomorphic Encryption”. In: *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. 2021, pp. 494–505. DOI: 10.1109/TrustCom53373.2021.00079.

[116] H. C. Pöhls and K. Samelin. “Accountable Redactable Signatures”. In: *2015 10th International Conference on Availability, Reliability and Security*. 2015, pp. 60–69.

[117] Christian Rathgeb and Andreas Uhl. “A survey on biometric cryptosystems and cancelable biometrics”. In: *EURASIP Journal on Information Security* 2011 (Sept. 2011). DOI: 10.1186/1687-417X-2011-3.

[118] Manisha Rawat and Nitin Kumar. “Cancelable Biometrics: a comprehensive survey”. In: *Artificial Intelligence Review* 53 (June 2020). DOI: 10.1007/s10462-019-09767-8.

[119] David Recordon and Drummond Reed. “OpenID 2.0: A Platform for User-Centric Identity Management”. In: *Proceedings of the Second ACM Workshop on Digital Identity Management*. New York, NY, USA: Association for Computing Machinery, 2006, 11–16. DOI: 10.1145/1179529.1179532.

[120] Leonie Reichert, Samuel Brack, and BjÖRN Scheuermann. “A Survey of Automatic Contact Tracing Approaches Using Bluetooth Low Energy”. In: *ACM Trans. Comput. Healthcare* (2021). ISSN: 2691-1957.

[121] R. L. Rivest, A. Shamir, and L. Adleman. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. In: *Commun. ACM* 21.2 (1978). ISSN: 0001-0782. DOI: 10.1145/359340.359342.

[122] Ronald L. Rivest and Michael L. Dertouzos. “On Data Banks and Privacy Homomorphisms”. In: 1978, pp. 169–180.

[123] Bragg Roberta. *CISSP Security Management and Practices*. https://www.pearsonitcertification.com/articles/article.aspx?p=30287.

[124] Rivest Ronald L. et al. *The PACT protocol specification*. 2020.

[125] Kai Samelin and Daniel Slamanig. “Policy-Based Sanitizable Signatures”. In: *Topics in Cryptology – CT-RSA 2020*. Cham: Springer International Publishing, 2020, pp. 538–563. ISBN: 978-3-030-40186-3.

[126] Olivier Sanders. "Efficient Redactable Signature and Application to Anonymous Credentials". In: *Public-Key Cryptography – PKC 2020: 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4–7, 2020, Proceedings, Part II*. Berlin, Heidelberg: Springer-Verlag, 2020, 628–656. ISBN: 978-3-030-45387-9.

[127] Arpita Sarkar and Binod Singh. "A review on performance,security and various biometric template protection schemes for biometric authentication systems". In: *Multimedia Tools and Applications* 79 (Oct. 2020). DOI: 10.1007/s11042-020-09197-7.

[128] Viktoriia Shubina et al. "Survey of Decentralized Solutions with Mobile Devices for User Location Tracking, Proximity Detection, and Contact Tracing in the COVID-19 Era". In: *Data* 5.4 (2020).

[129] Koen Simoens et al. "A Framework for Analyzing Template Security and Privacy in Biometric Authentication Systems". In: *IEEE Transactions on Information Forensics and Security* 7.2 (2012), pp. 833–841. DOI: 10.1109/TIFS.2012.2184092.

[130] N. P. Smart and F. Vercauteren. "Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes". In: PKC'10. Berlin, Heidelberg: Springer-Verlag, 2010, 420–443. ISBN: 3642130127. DOI: 10.1007/978-3-642-13013-7_25.

[131] Thomas J Smedinghoff. *Introduction to online identity management*. 2008.

[132] International Organization for Standardization (ISO). *ISO 7498-2:1989 Information processing systems — Open Systems Interconnection — Basic Reference Model — Part 2: Security Architecture*. https://www.iso.org/obp/ui/#iso:std:iso:7498:-2:ed-1:v1:en. 1990.

[133] International Organization for Standardization (ISO). *Personal identification — ISO-compliant driving licence — Part 5: Mobile driving licence (mDL) application*. https://www.iso.org/obp/ui/#iso:std:iso-iec:18013:-5:dis:ed-1:v1:en. 2021.

[134] Ron Steinfeld, Laurence Bull, and Yuliang Zheng. "Content Extraction Signatures". In: *Information Security and Cryptology — ICISC 2001*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 285–304.

[135] *Stop COVID-19 app*. https://github.com/Stop-COVID-19-Croatia/stopcovid19-docs, [Online accessed June 2022].

[136] Susan Svrluga. *Florida university removes some anti-racism statements, worrying faculty*. https://www.washingtonpost.com/education/2022/07/14/ucf-anti-racism-statements-removed/. 2022.

[137] Susan Svrluga. *How to Confront Online Racism?* https://museeholocauste.ca/en/resources-training/how-to-confront-online-racism/#Chap6. 2022.

[138] Thales. *Digital identity trends – 5 forces that are shaping 2022*. https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/identity/digital-identity-services/trends. 2021.

[139] Yiannis Tsiounis and Moti Yung. "On the security of ElGamal based encryption". In: *Public Key Cryptography*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 117–134.

[140] Maneesh Upmanyu et al. "Blind Authentication: A Secure Crypto-Biometric Verification Protocol". In: *Trans. Info. For. Sec.* 5.2 (2010), 255–268. DOI: 10.1109/TIFS.2010.2043188.

[141] Philippe Vallée. *What is Digital Identity, and why is it important?* https://dis-blog.thalesgroup.com/identity-biometric-solutions/2021/06/18/what-is-digital-identity-and-why-is-it-important/. 2021.

[142] Eric R. Verheul et al. "Polymorphic Encryption and Pseudonymisation for Personalised Healthcare". In: *IACR Cryptology ePrint Archive* (2016), p. 411.

[143] A. Wasef and X. Shen. "Efficient Group Signature Scheme Supporting Batch Verification for Securing Vehicular Networks". In: *2010 IEEE International Conference on Communications*. 2010, pp. 1–5. DOI: 10.1109/ICC.2010.5502136.

[144] Jie Xu et al. "An Identity Management and Authentication Scheme Based on Redactable Blockchain for Mobile Networks". In: *IEEE Transactions on Vehicular Technology* 69.6 (2020), pp. 6688–6698.

[145] Zhiyan Xu et al. "Privacy-Protection Scheme Based on Sanitizable Signature for Smart Mobile Medical Scenarios". In: *Wireless Communications and Mobile Computing* (Nov. 2020), pp. 1–10.

[146] Wencheng Yang et al. "Secure Fingerprint Authentication with Homomorphic Encryption". In: *2020 Digital Image Computing: Techniques and Applications (DICTA)*. 2020, pp. 1–6. DOI: 10.1109/DICTA51227.2020.9363426.

[147] Sahraoui Yesin et al. "TraceMe: Real-Time Contact Tracing and Early Prevention of COVID-19 based on Online Social Networks". In: *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*. 2022, pp. 893–896.

[148] Xiaohan Yue et al. "A Practical Group Signatures for Providing Privacy-Preserving Authentication with Revocation". In: *Security and Privacy in New Computing Environments, Second EAI International Conference, SPNCE 2019, Tianjin, China*. June 2019, pp. 226–245.

[149] Aiqing Zhang et al. "Application-Oriented Block Generation for Consortium Blockchain-Based IoT Systems With Dynamic Device Management". In: *IEEE Internet of Things Journal* 8.10 (2021), pp. 7874–7888. DOI: 10.1109/JIOT.2020.3041163.

[150] Kai Zhou and Jian Ren. "PassBio: Privacy-Preserving User-Centric Biometric Authentication". In: *IEEE Transactions on Information Forensics and Security* 13.12 (2018), pp. 3050–3063. DOI: 10.1109/TIFS.2018.2838540.

[151] Sujing Zhou and Dongdai Lin. "Unlinkable Randomizable Signature and Its Application in Group Signature". In: *Information Security and Cryptology*. Springer, Berlin, Heidelberg, 2008, pp. 328–342.

[152] Fei Zhu et al. "Cost-Effective Authenticated Data Redaction With Privacy Protection in IoT". In: *IEEE Internet of Things Journal* 8.14 (2021), pp. 11678–11689.

Appendix A

French Summary

L'identité numérique est considérée comme étant une des tendances technologiques les plus répandues dans le monde. Une accélération dans l'utilisation des identités numériques a été conduite par l'explosion des services en ligne due à la crise du Covid-19. Cette utilisation excessive entraîne une collecte massive de données sensibles par différents acteurs [48], et plusieurs attaques exercées sur ces données collectées [49].

La gestion des identités est le concept de base permettant de gérer les identités numériques. Plusieurs critères ont été définis pour concevoir les systèmes de gestion des identités. Ils incluent l'entité responsable de la gouvernance des identités, l'emplacement de stockage des identités et des attributs, le pouvoir donné aux entités centrales, comme les fournisseurs d'identités, et aussi le niveau de confiance accordé aux différents acteurs. En fonction de ces critères, différents modèles d'identité ont été déployés au cours des dernières décennies [35]. Tout d'abord, le modèle centralisé fait confiance au fournisseur d'identités chargé de fournir les identités et les attributs des utilisateurs. Le fournisseur d'identités authentifie les utilisateurs auprès de plusieurs fournisseurs de services. Ce rôle lui donne la capacité de suivre les comportements des utilisateurs. Deuxièmement, lorsque différents fournisseurs de services cherchent à collaborer entre eux, le modèle fédéré a été développé. En effet, les utilisateurs peuvent accéder à différents services en s'authentifiant auprès d'un seul fournisseur de services appartenant au domaine fédéré. Néanmoins, les utilisateurs peuvent toujours être tracés par les fournisseurs de services et leurs profils peuvent être construits [131]. La sensibilisation croissante à la vie privée des utilisateurs, grâce à l'émergence de nouvelles réglementations (par exemple, le règlement général sur la protection des données (RGPD) [57]), a conduit au développement du modèle centré sur l'utilisateur. Ce concept donne aux utilisateurs le contrôle total sur leurs identités. Ce modèle a suscité des préoccupations supplémentaires quant à l'authenticité des données fournies par les utilisateurs. En choisissant d'utiliser l'un des modèles susmentionnés, le premier objectif des utilisateurs est de bénéficier d'une gestion fluide de leurs identités et de leurs informations sur Internet.

A.1 Problématiques, objectifs et contributions

Depuis plusieurs décennies, les solutions de gestion des identités numériques sont utilisées dans les entreprises et les instituts. Elles visent à contrôler les droits d'accès des utilisateurs aux différents services et ressources. De nos jours, l'identité numérique est confrontée à de nouveaux défis car son utilisation est de plus en plus répandue dans toute la société [141]. Par exemple, de nombreux gouvernements ont adopté les programmes d'identité numérique régalienne. Certains de ces programmes incluent les identités biométriques, à savoir les passeports électroniques qui ont rapidement émergé avec plus d'un milliard de passeports en circulation aujourd'hui [138].
La définition de l'identité numérique ne comprend pas seulement les représentations numériques d'une personne physique ou morale (identifiants et attributs caractéristiques), mais couvre également toutes les traces qu'elle laisse sur Internet et qui lui sont attachées. L'identité numérique devient alors le lien entre l'utilisateur et la société (c'est-à-dire les individus, le gouvernement, les groupes et organisations, les réseaux sociaux, etc.). En vue de cette multitude, les utilisateurs se retrouvent submergés par des identités délivrées et/ou gérées par les différentes parties de la société. Le défi consiste alors, à concevoir des solutions d'identités interopérables et cohérentes afin d'améliorer l'expérience de l'utilisateur et de faciliter l'utilisation de ses identités numériques. Ce défi fait référence à l'**utilisabilité** des systèmes de gestion des identités.
Pour le relever, il est convenu que donner à l'utilisateur le contrôle total sur ses identités est une solution prometteuse. En effet, l'utilisateur est la partie commune à tous les systèmes d'identités, et il est de plus en plus sensible aux problématiques de préservation de la vie privée en utilisant les services numériques [1].

La gestion des identités est confrontée à un grand nombre de fraudes, d'attaques et de collectes de données. Par exemple, depuis le début de la crise Covid-19, la commission fédérale de commerce (FTC) a traité plus de 70 000 rapports d'usurpation d'identité pour des services d'achat en ligne [48]. De plus, selon [47], Google, par exemple, collecte 10 exaoctets de données par jour. Pour faire face à ces problèmes, il est crucial d'établir un cadre de confiance entre les entités impliquées dans une solution de gestion des identités, à savoir les fournisseurs d'identités, les utilisateurs et les fournisseurs de services. Ce cadre doit garantir la protection des données et de leurs propriétaires, dite **sécurité** des données et préservation de la **vie privée** des utilisateurs :

- Sécurité des données : concerne deux enjeux majeurs. Le premier est la confidentialité, qui consiste à garder les données secrètes vis-à-vis des parties non autorisées. La seconde concerne la cohérence et l'intégrité des données. Elle garantit que des données falsifiées ou modifiées ne peuvent être partagées avec d'autres parties, ni utilisées pour accéder à des services. Cela signifie qu'un contrôle rigoureux doit être appliqué aux données pour vérifier leur validité.

- Préservation de la vie privée des utilisateurs : suggère que même si plusieurs informations d'un même utilisateur sont collectées par plusieurs parties, elles ne peuvent pas être utilisées pour le tracer ou établir un profil partiel/complet de lui. La préservation de la vie privée implique également que les utilisateurs ne puissent

pas être réidentifiés en fonction des informations communiquées et connues par plusieurs parties indépendantes.

En résumé, la gestion des identités est confronté à des défis en termes de sécurité, de vie privée et d'utilisabilité.

Dans cette thèse, financée par la chaire Valeurs et Politiques des Informations Personnelles (VP-IP), nous voulons concevoir des solutions concrètes qui garantissent le compromis entre sécurite, vie privé et utilisabilité. En d'autres termes, un utilisateur est capable de prouver aux fournisseurs de services la validité de son identité et de ses attributs, d'accéder à des services et à des ressources ou de partager des données avec différentes parties sans que sa vie privée en soit pour autant menacée. Il s'agit donc de répondre aux objectifs suivants dans cette thèse :

- ***Objectif A*** – donner aux utilisateurs le contrôle total sur leurs identités et leurs attributs afin de sélectionner et de divulguer le minimum de données qui répondent aux politiques des fournisseurs de services.
- ***Objectif B*** – contrôler et vérifier la validité des données et des calculs effectués sur ces données, grâce à des mécanismes rigoureux, avant d'autoriser les utilisateurs à accéder aux services et aux ressources.
- ***Objectif C*** – concevoir un mécanisme sécurisé pour gérer les identités éphémères.
- ***Objectif D*** – garantir l'anonymat/pseudonymat des utilisateurs et la non-associabilité entre leurs transactions afin d'éviter le traçage et le profilage involontaires des utilisateurs.
- ***Objectif E*** – fournir des preuves formelles ou informelles des propriétés de sécurité et de vie privée des constructions proposées.
- ***Objectif F*** – proposer un prototype des solutions proposées et implémenter les différents algorithmes, afin de valider leur faisabilité et d'évaluer leur efficacité sur du matériel réel.

Pour satisfaire les objectifs susmentionnés, nous nous appuyons sur deux propriétés : le pseudonymat qui permet d'interagir avec les autres parties sous un pseudonyme et la malléabilité des mécanismes cryptographiques qui permet à l'utilisateur de manipuler des éléments cryptographiques sans que leurs propriétés en soient affectées.

Les contributions de cette thèse sont résumées ci-dessous.

- *Contribution 1* – conception d'un système de gestion des identités préservant la vie privée et basé sur des pseudonymes et une signature malléable et non-associable [94]. Le système proposé permet de **contrôler l'accès des utilisateurs** aux services et aux ressources. En effet, chaque utilisateur reçoit un certificat, sur ses attributs, délivré par un fournisseur d'identités de confiance. Grâce aux

signatures malléables, il est en mesure de supprimer les attributs qu'il ne souhaite pas divulguer et ne conserve que les données minimales requises pour accéder aux services. L'utilisateur adapte le certificat en conséquence. Cette solution offre aux utilisateurs un contrôle total sur leurs identités et leurs attributs, tout en permettant aux fournisseurs de services de vérifier l'authenticité des données fournies, à travers des sessions pseudonymisées. Ainsi, deux fournisseurs de services ne sont pas en mesure de relier les transactions à un même utilisateur, ni même d'identifier l'utilisateur. Une mise en œuvre de la solution proposée démontre de très bonnes performances. En effet, en considérant un message de 100 blocs, des pourcentages de 50% de blocs admissibles et de 25% (de tous les blocs) de blocs modifiés, les temps de calcul, obtenus à la fois sur un ordinateur portable et un smartphone, sont de l'ordre de quelques millisecondes (*Objectif A*, *Objectif B*, *Objectif D*, *Objectif E* et *Objectif F*).

- *Contribution 2* – proposition d'un schéma d'authentification biométrique préservant la vie privée pour les systèmes de gestion des identités [95] et permettant de mettre en œuvre un **contrôle d'accès** physique à un service. En effet, un utilisateur reçoit un modèle représentant son identité biométrique. Ce modèle est chiffré et certifié par un fournisseur d'identités. L'utilisateur est en mesure de modifier ces credentials sans compromettre leur authenticité, et de s'enregistrer de façon anonyme auprès d'un fournisseur de services. Pour s'authentifier, l'utilisateur est invité à présenter un nouveau modèle biométrique au fournisseur de services, qui calcule la différence avec celui fourni à l'enregistrement. La solution proposée démontre la résistance à l'utilisation de fausses identités biométriques, la fiabilité de l'authentification et la préservation de la vie privée des utilisateurs. En considérant un modèle biométrique de 600 échantillons, les temps de calculs acceptables ne dépassant pas 6 secondes pour un algorithme, prouvent l'efficacité de la solution (*Objectif A*, *Objectif B*, *Objectif D*, *Objectif E* et *Objectif F*).

- *Contribution 3* – conception d'un protocole préservant la vie privée lors du **partage de données**. Ce protocole considère un type spécifique d'attributs, notamment les attributs éphémères, dans le contexte du traçage de proximité pour les systèmes e-santé [93]. La solution proposée s'appuie sur une architecture hybride qui implique un serveur centralisé et des proxies décentralisés. Elle permet aux utilisateurs de partager leurs informations de contact avec d'autres personnes et d'être avertis en cas de contact avec des personnes infectées. Grâce à la sécurité du protocole, les utilisateurs sont assurés de ne recevoir que des alertes correctes, c'est-à-dire que les fausses alertes positives sont détectées. Une analyse de sécurité détaillée démontre la résistance aux tentatives d'associabilité et d'identification. Une mise en œuvre de la solution avec un effort pour minimiser les coûts de calcul du protocole proposé, montre son efficacité. Par exemple, pour garantir la non-falsification d'un contact, le temps de calcul atteint 4 secondes pour un niveau de sécurité de 128 bits. La vérification de l'authenticité de ce contact nécessite 19 secondes sur une machine de faible capacité (*Objectif A*, *Objectif C*, *Objectif D*, *Objectif E* et *Objectif F*).

- *Contribution 4* – proposition d'un schéma de signature de groupe qui offre une vérification efficace, agrégée et par lot sur de multiples preuves de connaissance [96].

Ce schéma assure l'intégrité pendant le partage des données. L'implémentation de la solution démontre une grande efficacité de la vérification agrégée et par lot par rapport à une vérification naïve d'une signature de groupe, sans compromettre la sécurité et la vie privée (*Objectif D*, *Objectif E* et *Objectif F*).

A.2 Schémas de chiffrement et de signatures malléables pour les systèmes de gestion des identités

Dans cette section, nous présentons un état de l'art sur les systèmes de gestion des identités en nous focalisant sur les besoins de préservation de la vie privée. À cette fin, nous fournissons d'abord une revue complète des systèmes de gestion des identités en présentant les différents acteurs impliqués, en énumérant les propriétés qui doivent être satisfaites, en identifiant les modèles d'identité. Ensuite, nous introduisons certains mécanismes cryptographiques malléables souvent utilisées pour gérer les identités et protéger les données sensibles.

A.2.1 Identité et gestion des identités

Le terme d'identité numérique fait souvent référence à la représentation d'une entité au sein d'un domaine [6]. Une entité peut avoir une ou plusieurs identités dans un domaine particulier. L'identité numérique représente les moyens informatiques et technologiques qui permettent à cette entité de se projeter dans le monde numérique, et cette projection peut prendre plusieurs formes en fonction de son contexte (administratif, professionnel, réseaux sociaux, etc.) [111]. D'un point de vue technique, cette représentation consiste à associer à un individu un ensemble de données numériques, appelées attributs et credentials[1]. Ces attributs comprennent des caractéristiques permanentes comme l'origine ethnique et la date de naissance. Les attributs peuvent également être modérément statiques ou très difficiles à modifier au cours de la vie d'une personne, comme le nom de famille, le prénom, l'empreinte vocale et la couleur des yeux. Les attributs concernent également des caractéristiques hautement dynamiques, par exemple les centres d'intérêt de la personne, sa géolocalisation, les informations relatives à la communication et au logement telles que le numéro de téléphone, l'adresse électronique et l'adresse postale. De nouvelles formes d'attributs numériques ont évolué, notamment la biométrie (par exemple, les empreintes digitales, la reconnaissance faciale, l'analyse de l'iris, etc.) qui permet de créer des identifiants uniques pour les individus, par exemple lors de la validation d'une transaction de paiement sur un appareil mobile. Ainsi, une identité numérique peut être considérée comme le moyen de distinguer une entité des autres, en ligne.

Le concept de gestion des identités a été introduit afin de simplifier la gestion des identités numériques dans les systèmes informatiques. Il permet d'enregistrer des

[1] Un credential représente les éléments d'information utilisés dans l'authentification des attributs revendiqués ou de l'identité appartenant à une entité particulière [105]

utilisateurs dans le système, de gérer l'authentification, l'autorisation et le contrôle de l'accès aux services et aux ressources, et de fournir les mesures techniques assurant la sécurité et la vie privée des identités [40].

Entités Un système de gestion des identités comprend trois entités principales, à savoir l'utilisateur ($\mathcal{U}$), le fournisseur de services ($\mathcal{SP}$) et le fournisseur d'identités ($\mathcal{IP}$) et une entité facultative appelée fournisseur d'attributs ($\mathcal{AP}$) :

- **User:** est une personne physique, une organisation ou un appareil, qui dispose d'une ou plusieurs identités numériques gérées par un ou plusieurs fournisseurs d'identités. Ces identités lui autorise à accéder aux services et aux ressources fournis par un ou plusieurs fournisseurs de services.
- **Fournisseur de services:** est l'entité chargée de fournir des services en ligne aux utilisateurs tels que le commerce électronique, la banque en ligne, la e-santé, le stockage en nuage, etc. À cette fin, $\mathcal{SP}$ doit authentifier l'utilisateur en s'assurant qu'il est bien celui qui prétend accéder au service. En respectant les politiques d'accès prédéfinies, les fournisseurs de services collectent auprès des utilisateurs des attributs qui les caractérisent. Dans ce cas, un fournisseur de services doit être considéré comme un tiers de confiance.
- **Fournisseur d'identités:** est responsable de la délivrance des identités et des attributs de base aux utilisateurs. $\mathcal{IP}$ maintient et confirme les attributs requis dans le processus d'authentification des utilisateurs auprès des fournisseurs de services tels que le nom, le numéro de compte/carte bancaire, numéro de sécurité sociale, etc.
- **Fournisseur d'attributs:** représentent des acteurs facultatifs qui délivrent des attributs supplémentaires décrivant l'utilisateur. Ils confirment aux founisseurs de services ces attributs supplémentaires tels que le revenu fiscal.

Besoins des systèmes de gestion des identités Dans le monde du numérique, plusieurs défis se posent aux systèmes de gestion des identités qui cherchent à établir la confiance entre les différentes entités (c'est-à-dire les utilisateurs, les fournisseurs d'identités, d'attributs et de services). Ces défis sont formalisés en trois types de besoins notamment les besoins de sécurité, de vie privée et d'utilisabilité.

- **Besoins de sécurité**
  - (S1) Confidentialité – garantit que les données sensibles des utilisateurs sont protégées pendant leur transfert et leur stockage et ne peuvent être lues que par des entités autorisées.
  - (S2) Intégrité des données – garantit que les données des utilisateurs n'ont pas été modifiées par des entités non autorisées.

- (S3) Authenticité – garantit que les données des utilisateurs sont originales et qu'elles arrivent à destination telles qu'elles ont été envoyées par leur propriétaire.
- (S4) Authentification de l'entité homologue – garantit que l'entité homologue dans une communication est bien celle revendiquée.
- (S5) Authentification de l'origine des données – garantit que la source des données reçues est bien celle revendiquée.
- (S6) Autorisation – détermine les droits et les privilèges des utilisateurs en matière d'accès aux services, ressources, fichiers, programmes, etc.
- (S7) Disponibilité – garantit que le système fonctionne correctement à tout moment et partout où les entités en ont besoin.
- (S8) Non-répudiation – garantit qu'une entité ayant exécuté une transaction ne peut pas nier l'action.

- **Besoins de vie privée**
  - (P1) Anonymat – garantit qu'un utilisateur peut accéder à un service ou à une ressource tout en maintenant son identité anonyme (non identifiable). Par conséquent, l'utilisateur ne peut pas "être identifié directement ou indirectement" [72] par $\mathcal{IP}$ ou $\mathcal{SP}$.
  - (P2) Pseudonymat – représente un niveau inférieur d'anonymat. En effet, lors des interactions, l'utilisateur est connu des $\mathcal{IP}$s et/ou $\mathcal{SP}$s par un alias (temporaire ou permanent) qui peut être inversé à l'identité de l'utilisateur [72]. Le pseudonymat permet de prendre en charge le concept d'un utilisateur détenant plusieurs identités virtuelles.
  - (P3) Minimisation des données – est un principe fondamental de préservation de la vie privée et l'une des exigences du RGPD. Elle implique que les applications ne sollicitent et ne traitent que le minimum d'informations strictement nécessaires à une transaction particulière [32]. L'objectif est de minimiser la quantité de données personnelles collectées et utilisées par les fournisseurs de services, notamment pour réduire le risque de profilage basé sur le comportement de l'utilisateur.
  - (P4) Non-associabilité – fait référence à l'incapacité de lier deux ou plusieurs éléments d'information distincts (enregistrements, messages, URL, actions, identifiants) concernant un utilisateur ou un groupe d'utilisateurs. D'une part, la non-associabilité peut concerner l'incapacité de lier les preuves d'identité à l'identité/au certificat d'origine, en se basant sur les informations fournies lors du processus de délivrance de l'identité. Une telle propriété est appelée ***non-associabilité issue-show***. D'autre part, la ***non-associabilité multi-show*** se réfère à l'incapacité de lier plusieurs preuves d'identité générées sur la même identité/credentials et transmises sur plusieurs sessions.

- **Besoins d'utilisabilité**
  - (U1) Gestion des identités multiples – est un principe qui a été fortement soutenu par la chaire Valeurs et politiques des informations personnelles

(VP-IP) [79]. Il consiste à établir les meilleures pratiques pour aider les utilisateurs à gérer leurs multiples identités qu'ils utilisent pour se présenter dans différents contextes. Par exemple, un même utilisateur peut être présenté comme un père, un enseignant, un client de banque, un fan de musique, une personne diabétique, etc. À cette fin, différentes pratiques pourraient être prises en considération, notamment les technologies de pseudonymisation, en fonction du contexte. Les utilisateurs peuvent sélectionner les attributs à divulguer et les associer à un pseudonyme propre à chaque fournisseur de services.

– (U2) Identité unique pour des transactions multiples – indique qu'un utilisateur peut avoir une identité unique qui contient tous les credentials possibles dont il a besoin pour effectuer des transactions avec plusieurs fournisseurs de services. L'utilisateur a alors la possibilité de gérer ses credentials de manière à fournir que ceux qui sont nécessaires pour une transaction particulière.

Catégories des systèmes de gestion des identités Il existe trois catégories de systèmes de gestion des identités, notamment le modèle centralisé, fédéré et centré sur l'utilisateur.

- **Modèle centralisé** – suggère qu'un identifiant unique et plusieurs ceredentials, générés par le même $\mathcal{IP}$, sont utilisés par chaque $\mathcal{SP}$ [76]. $\mathcal{U}$ fournit le même identifiant et les mêmes credentials, stockés chez $\mathcal{IP}$, à plusieurs $\mathcal{SP}$ afin d'accéder aux services. Pour authentifier $\mathcal{U}$, $\mathcal{SP}$ demande à $\mathcal{IP}$, considéré comme une partie de confiance, de vérifier l'identité de $\mathcal{U}$.

- **Modèle fédéré** – considère qu'un groupe de $\mathcal{SP}$s établit un ensemble d'accords et de normes afin que les identifiants des utilisateurs soient reconnus d'un $\mathcal{SP}$ à l'autre au sein du domaine fédéré [40]. Ce modèle suggère que $\mathcal{U}$ peut avoir des identifiants différents pour chaque $\mathcal{SP}$. Mais il pourrait utiliser un identifiant unique pour s'authentifier auprès de tous les $\mathcal{SP}$s appartenant au domaine fédéré, grâce à une correspondance entre les identifiants appartenant au même utilisateur.

- **Modèle centré sur l'utilisateur** – a été introduit pour mettre en œuvre la gestion des identités du côté de l'utilisateur. Les utilisateurs ont le contrôle sur leurs identités [76]. Ils sont en mesure de sélectionner les informations à divulguer et d'être informés lorsque leurs informations sont collectées. Le consentement des utilisateurs est également requis pour tout type d'analyse et de manipulation de leurs informations collectées. L'identité et les attributs de $\mathcal{U}$ peuvent être stockés dans un dispositif matériel (par exemple, une carte à puce, un dispositif personnel portable). Ainsi, $\mathcal{U}$ ne mémorise que le credential lui permettant d'accéder au dispositif matériel, au lieu de se souvenir de plusieurs identifiants.

Dans la Figure A.1, nous évaluons les différentes catégories des systèmes de gestion des identités en termes de satisfaction des propriétés de sécurité, de vie privée et d'utilisabilité.

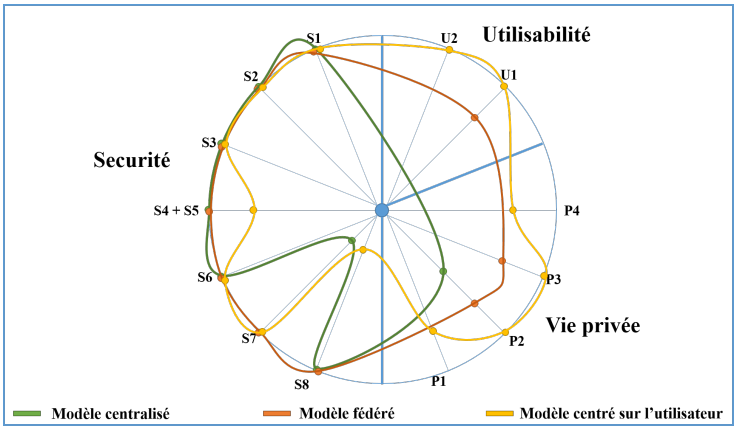


Figure A.1: Propriétés satisfaites par les modèles d'identités

En termes de sécurité, nous affirmons que tous les modèles d'identité prennent en charge les principales propriétés. Cependant, selon la conception et la construction des systèmes, certaines propriétés n'ont pas pu être atteintes. Par exemple, pour le modèle centralisé, la vulnérabilité à l'attaque déni de service affecte la propriété de disponibilité. De plus, compte tenu de la conception et de la construction concrète du système et de la technologie cryptographique sous-jacente, la propriété de non-répudiation peut ne pas être atteinte par certains systèmes centrés sur l'utilisateur.
En ce qui concerne la vie privée, à l'exception du pseudonymat, d'autres propriétés ne sont pas satisfaites par le modèle centralisé. En effet, les utilisateurs sont identifiés de manière unique et peuvent être suivis par le fournisseur d'identités centralisé. Le modèle fédéré, tel qu'il est actuellement bien utilisé, a évolué pour se conformer aux nouvelles réglementations telles que le RGPD, prenant ainsi en charge certaines exigences en matière de respect de la vie privée comme le pseudonymat et la minimisation des données. Le modèle centré sur l'utilisateur semble être le plus prometteur pour répondre aux exigences souhaitées, car l'utilisateur joue un rôle central dans ce modèle de gestion des identités. En effet, lors du traitement de leurs attributs et leurs credentials, les utilisateurs peuvent rester anonymes ou sous pseudonymat vis-à-vis des fournisseurs de services. Ils peuvent également empêcher ces fournisseurs de relier leurs transactions entre elles.
Enfin, en termes d'utilisabilité, l'identité centrée sur l'utilisateur semble être le meilleur modèle pour garantir cette propriété. En effet, les utilisateurs sont capables de gérer soit leurs identités multiples, soit une seule identité et des attributs multiples.

A.2.2 Technologies malléables pour préserver la vie privée

Dans cette section, nous nous concentrons sur les mécanismes cryptographiques malléables, notamment les schémas de signature et de chiffrement qui offrent des caractéristiques de malléabilité telles que la randomisation et la modification.

- **Signatures de groupe** – permettent à tout membre du groupe, notamment un

utilisateur, de signer un message au nom du groupe, tout en restant anonyme. Ainsi, les vérifieurs authentifient l'utilisateur en tant que membre du groupe, mais ne sont pas en mesure de l'identifier. Il s'agit l'authentification de l'entité homologue. Chaque groupe est composé d'un gestionnaire de groupe et de plusieurs membres, et est caractérisé par une clé privée de groupe avec laquelle le message est signé, et le vérifieur vérifie la validité de la signature en utilisant la clé publique de groupe correspondante.

- **Signatures assainissables** – permettent, en connaissant les clés secrètes appropriées, de dériver efficacement, à partir d'une signature σ sur un message m=(m_1,m_2,...,m_n) (où n est le nombre de blocs du message m), une signature σ' sur un message m'=(m'_1,m'_2,...,m'_n) où m' est une transformation admissible de m [41]. Les signatures assainissables, ont été introduites par Ateniese *et al.* [11] permettant à une partie désignée, appelée l'assainisseur, de modifier certaines parties, choisies par le signataire, d'un message signé. La signature correspondante reste valide sous la clé du signataire et l'authenticité du message m' est garantie.

- **Signatures caviardables** – ont été introduites par Steinfeld *et al.* [134] permettant à toute entité de supprimer certaines parties d'un message signé sans interagir avec le signataire original. Le nouveau message reste authentique et la signature correspondante reste valide sous la clé du signataire.

- **Chiffrement homomorphe** – constitue une forme particulière de chiffrement. Il permet à des tiers d'appliquer certaines fonctions sur des données chiffrées, de manière aveugle, c'est-à-dire sans avoir accès au contenu original du texte chiffré. Le texte chiffré peut être déchiffré par une seule clé étant la clé secrète de l'utilisateur.

- **Chiffrement polymorphe** – est un nouveau type de chiffrement des données. Il reproduit la même idée que le chiffrement homomorphe, cependant, la principale différence est que le texte chiffré peut être déchiffré par plusieurs clés. En effet, une entité chiffre des données en clair à l'aide d'une clé de chiffrement K_{enc}. Ce chiffré est transféré à une partie intermédiaire, appelée transcrypteur aveugle, qui le transfome et désigne l'entité responsable de le déchiffrer.

A.2.3 Conclusion

Dans cette section, une introduction générale aux systèmes de gestion des identités a été présentée en soulignant les différentes exigences qui se posent en termes de sécurité, de vie privée et d'utilisabilité. Quelques technologies pour préserver la vie privée ont été introduites en se focalisant sur celles qui offrent des propriétés de malléabilité.

La section suivante, présente notre première contribution qui détaille un système de gestion des identités préservant la vie privée et basé sur des signatures malléables non-associables.

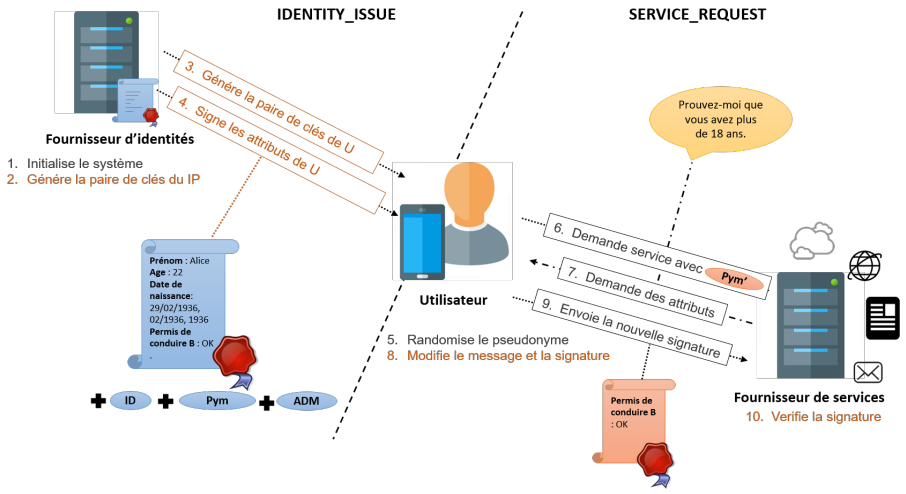


Figure A.2: Architecture de PIMA et principales interactions entre les entités

A.3 Système de gestion des identités respectueux la vie privée et basé sur des signatures malléables non-associables

Cette section présente notre première contribution intitulée PIMA [94]. PIMA est un nouveau système de gestion des identités centré sur l'utilisateur et respectueux de la vie privée. En se basant sur un schéma de siganture malléable, l'utilisateur a le contrôle total sur les attributs afin d'accéder aus services et aux ressources.

A.3.1 Architecture

PIMA fait intervenir trois entités principales, notamment l'utilisateur ($\mathcal{U}$), le fournisseur d'identités ($\mathcal{IP}$) et le fournisseur de services ($\mathcal{SP}$), comme le montre la figure A.2. Lors de la phase IDENTITY_ISSUE, $\mathcal{U}$ reçoit de $\mathcal{IP}$ un pseudonyme et une identité certifiée $\sigma_{\mathcal{U}}$ sur ses attributs qu'il stocke localement.

Ensuite, lors de la phase SERVICE_REQUEST, $\mathcal{U}$ est en mesure de sélectionner les attributs qu'il souhaite divulguer, de dériver un nouveau credential de $\sigma_{\mathcal{U}}$ et d'interagir de manière pseudonyme avec le $\mathcal{SP}$. $\mathcal{SP}$ vérifie le credential fourni afin de donner à $\mathcal{U}$ l'accès aux services.

L'architecture de PIMA, représentée dans la Figure A.2 inclut trois phases, à savoir SETUP, IDENTITY_ISSUE et SERVICE_REQUEST.

SETUP – Cette phase consiste à mettre en place et à initialiser l'ensemble du système. En effet, elle inclut la génération des paramètres publics du système et la génération des paires de clés de $\mathcal{IP}$ et $\mathcal{U}$, respectivement, par $\mathcal{IP}$.

IDENTITY_ISSUE – Cette phase consiste à délivrer l'identité d'un utilisateur $\mathcal{U}$ par

$\mathcal{IP}$. Premièrement, $\mathcal{IP}$ génère un identifiant ID auquel il associe un pseudonyme Pym et un ensemble d'attributs $Attr = \{a_k\}_{k=1}^{l}$ où l est le nombre d'attributs décrivant l'identité de $\mathcal{U}$ et $Attr \subseteq \mathtt{A}$ ($\mathtt{A}$ est l'univers des attributs). Deuxièmement, $\mathcal{IP}$ définit un message m sous la forme $m = \{ID_{\mathcal{IP}}, D, opt, Attr\}$, où $ID_{\mathcal{IP}}$ est l'identifiant de $\mathcal{IP}$, D est un ensemble de valeurs temporelles telles que les dates de délivrance et d'expiration de l'identité et *opt* représente d'autres options spécifiques à $\mathcal{IP}$. Ces éléments constituent la partie fixe du message m. Finalement, $\mathcal{IP}$ signe le message m en tenant en considération un ensemble de modifications admissibles ADM. A la fin de cette procédure, l'identifiant ID, le pseudonyme Pym, le message m, la signature correspondante et l'ensemble de modifications admissibles ADM sont envoyés à $\mathcal{U}$, qui les stocke localement.

Service_Request – Cette phase se produit lorsque $\mathcal{U}$ a besoin d'accéder à un service offert par $\mathcal{SP}$. D'abord, $\mathcal{U}$ sollicite $\mathcal{SP}$ en utilisant un nouveau pseudonyme qu'il dérive de son pseudonyme local Pym. Ce nouveau pseudonyme s'écrit $Pym_{\mathcal{U}}@\mathcal{SP}$ pour dire que c'est le pseudonyme de $\mathcal{U}$ chez $\mathcal{SP}$. Nous supposons que chaque utilisateur a un pseudonyme différent pour chaque $\mathcal{SP}$s afin d'assurer la non-associabilité entre ses transactions. Après avoir reçu le pseudonyme $Pym_{\mathcal{U}}@\mathcal{SP}$, $\mathcal{SP}$ choisit un ensemble d'attributs $AttrReq = \{attr_j\}_{j=1}^{n}$ permettant à $\mathcal{U}$ d'avoir accès au service demandé s'il parvient à prouver leur possession. Alors, en respectant l'ensemble des modifications admissibles ADM, $\mathcal{U}$ définit un ensemble de modifications possibles MOD et modifie la signature, par conséquent. Notez que la clé publique de $\mathcal{U}$ est aussi randomisée. Tous les éléments calculés par $\mathcal{U}$ sont envoyés à $\mathcal{SP}$, qui vérifie la validité de la signature.

A.3.2 Propriétés satisfaites

En nous appuyant sur des jeux de sécurité et des preuves formelles, nous montrons que, Pima satisfait les propriétés de sécurité et de vie privée suivantes :

- **Infasifiabilité** – signifie qu'un utilisateur malveillant, ne connaissant pas la clé secrète du $\mathcal{IP}$ et les pondérations secrètes associées aux attributs et n'étant pas autorisé par $\mathcal{IP}$, n'est pas capable de générer une paire message/signature valide.
- **Non-associabilité** – couvre deux sous-propriétés : (i) la *non-associabilité multi-transactions* garantit que deux ou plusieurs fournisseurs de services curieux ne sont pas en mesure de lier plusieurs signatures modifiées dérivées d'une signature sur le même message et transmises sur plusieurs sessions, (ii) la *non-associabilité to-original* garantit qu'un adversaire ne peut pas lier une signature modifiée à la signature d'origine même si cette dernière est connue.
- **Vie privée renforcée** – signifie qu'un adversaire malveillant n'est pas capable d'extraire des informations sur l'utilisateur ou les messages échangés, au-delà de ce qui a été volontairement révélé. En effet, il n'est pas possible pour un fournisseur curieux d'identifier un utilisateur particulier sur la base des informations relatives aux différentes transactions, ni de décider quelles données ont été modifiées ou supprimées d'un message donné.

A.3.3 Conclusion

Cette première contribution propose PIMA, un système de gestion d'identité respectueux de la vie privée et centré sur l'utilisateur, basé sur une nouvelle signature malléable non-associable [94]. PIMA permet à un utilisateur de sélectionner et de cacher les attributs qu'il ne veut pas divulguer. Les fournisseurs de services sont toujours en mesure d'authentifier l'origine des données afin d'autoriser les utilisateurs à accéder aux services.

Cependant, dans certains cas, prouver la possession d'attributs certifiés n'est pas suffisant pour accéder aux services et ressources. Les fournisseurs de services doivent vérifier que les attributs appartiennent à la personne physique qui effectue la transaction, ce qui fait référence à l'authentification de l'entité homologue. Ainsi, dans la prochaine section, nous aborderons les problèmes d'authentification des utilisateurs dans les systèmes de gestion des identités, en nous focalisant sur les identités biométriques.

A.4 Protocole d'authentification biométrique centré sur l'utilisateur et respectueux de la vie privée

Cette section présente notre deuxième contribution intitulée PUBA [95]. PUBA est un nouveau schéma d'authentification biométrique préservant la vie privée des utilisateurs, et conçu pour les sytèmes de gestion des identités centrés sur l'utilisateur.

A.4.1 Architecture

PUBA implique trois acteurs, notamment l'utilisateur ($\mathcal{U}$), le fournisseur d'identités ($\mathcal{IP}$) et le fournisseur de services ($\mathcal{SP}$). Lorsqu'il est physiquement présent chez $\mathcal{IP}$, $\mathcal{U}$ obtient un modèle biométrique chiffré et certifié sur ses données biométriques, appelé credentials. $\mathcal{U}$ s'enregistre à distance chez un $\mathcal{SP}$ en fournissant des credentials modifiés valides. Pour accéder aux services, $\mathcal{U}$, étant physiquement présent chez le $\mathcal{SP}$, fournit un nouveau modèle biométrique qui sera comparé à celui enregistré pour décider d'authentifier ou non $\mathcal{U}$.

L'architecture de PUBA, représentée dans la Figure A.3 inclut quatre phases, à savoir SETUP, IDENTITY_ISSUE, ENROLLMENT et VERIFICATION.

SETUP – Cette phase consiste à initialiser le système, à configurer les paramètres publics et à générer les clés des différentes entités du système.

IDENTITY_ISSUE – Cette phase consiste à délivrer une identité biométrique certifiée à $\mathcal{U}$. En effet, $\mathcal{U}$ doit être présent chez $\mathcal{IP}$ et prouver son identité au moyen d'un document légal (par exemple, un passeport), afin d'obtenir son identité biométrique. $\mathcal{U}$ reçoit, de la part de $\mathcal{IP}$, un modèle de référence chiffré E_r de son modèle biométrique numérique T_r, auquel est associé un credential signé $\sigma_{r_{\mathcal{IP}}}$. Le couple $(E_r, \sigma_{r_{\mathcal{IP}}})$ est

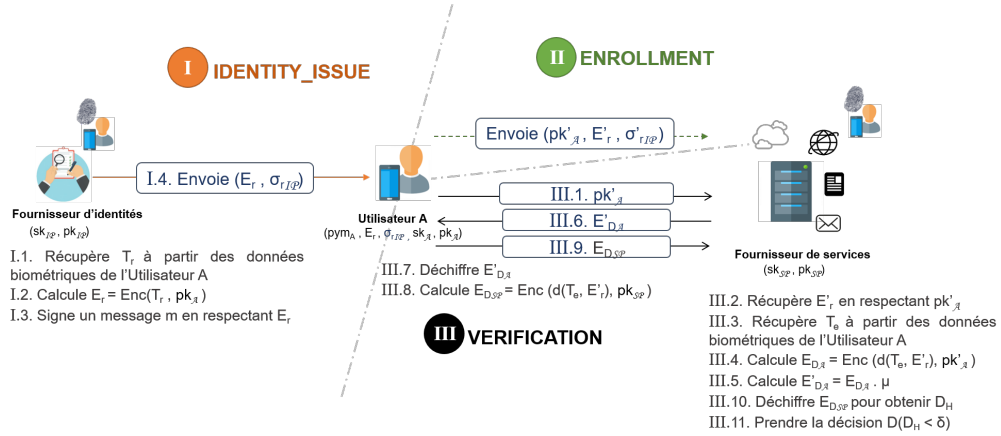


Figure A.3: Architecture de PUBA et principales interactions entre les entités

stocké localement chez $\mathcal{U}$.

ENROLLMENT – Cette phase se produit lorsque $\mathcal{U}$ souhaite s'enregistrer chez un $\mathcal{SP}$. Pour ce faire, $\mathcal{U}$ dérive un nouveau couple $(E'_r, \sigma'_{r_{\mathcal{IP}}})$, à partir du couple $(E_r, \sigma_{r_{\mathcal{IP}}})$, qu'il envoie à $\mathcal{SP}$. Notez que $\mathcal{U}$ est capable de dériver plusieurs credentials à partir du couple $(E_r, \sigma_{r_{\mathcal{IP}}})$ afin de demeurer non-associable entre plusieurs fournisseurs de services. Une version randomisée de la clé de chiffrement de $\mathcal{U}$ est également partagée avec $\mathcal{SP}$ et considérée comme le pseudonyme de $\mathcal{U}$ chez $\mathcal{SP}$. $\mathcal{SP}$ vérifie la signature et stocke les credentials et la clé fournis.

VERIFICATION – Cette phase consiste à authentifier $\mathcal{U}$ lorsqu'il est physiquement présent chez le $\mathcal{SP}$. En effet, un nouveau modèle T_e, dit extrait, est récupéré par $\mathcal{SP}$ à partir de la donnée biométrique de $\mathcal{U}$. $\mathcal{SP}$ récupère aussi le modèle de référence enregistré et calcule de manière homomorphe la distance entre les deux modèles enregistrés. $\mathcal{SP}$ randomise la distance chiffrée et la renvoie à $\mathcal{U}$ qui la rechiffre en aveugle avec la clé de $\mathcal{SP}$. Finalement, $\mathcal{SP}$ vérifie si $\mathcal{U}$ a correctement rechiffré la distance, et décide d'authentifier ou non $\mathcal{U}$, en comparant la distance à un seuil prédéfini.

A.4.2 Propriétés satisfaites

PUBA est démontrée satisfaire plusieurs propriétés de sécurité et de vie privée, définies ci-dessous, à travers des jeux de sécurité et des preuves formelles.

- **Infalsifiabilité** – signifie qu'un utilisateur malveillant n'est pas capable de générer une signature valide sur un modèle biométrique à moins que la clé secrète du $\mathcal{IP}$ soit connue. Par conséquent, l'utilisateur n'est pas en mesure de s'enregistrer chez un fornisseur de services en utilisant de faux modèles biométriques sans que la falsification ne soit détectée.
- **Fiabilité de l'authentification** – garantit qu'un adversaire malveillant non légitime n'est pas capable de se faire passer pour un utilisateur légitime et de s'authentifier

avec succès pour accéder aux services et aux ressources.

- **Non-associabilité** – garantit que deux ou plusieurs fournisseurs de services curieux ne sont pas capables de lier plusieurs sessions d'enregistrement appartenant au même utilisateur. Ceci implique que plusieurs modèles biométriques chiffrés appartenant au même utilisateur ne peuvent pas être liés entre eux.
- **Anonymat** – signifie que des entités curieuses en collusion ne sont pas capables de déterminer l'identité réelle d'un utilisateur enregistré. En d'autres termes, les entités curieuses ne sont pas en mesure de relier des credentials modifiés à leur origine, même si cette dernière est connue.

A.4.3 Conclusion

Cette deuxième contribution présente, PUBA, un nouveau schéma d'authentification biométrique pour la gestion des identités centrée sur l'utilisateur.

Bien que les deux premières solutions PIMA et PUBA sont conçues pour contrôler l'accès aux services et aux ressources, les systèmes de gestion des identités peuvent également être utilisés pour permettre aux utilisateurs de partager des données entre différentes entités. Ainsi, dans la section suivante, nous proposons une solution qui répond au besoin de partager des données dans un système de gestion des identités. Nous nous focalisons en particulier sur la gestion des attributs éphémères des utilisateurs dans le contexte du traçage de proximité pour les systèmes d'e-santé.

A.5 Protocole de traçage de proximité sécurisé et respectueux de la vie privée pour les systèmes de e-santé

Cette section présente une fusion de nos troisième et quatrième contributions. En effet, nous proposons un nouveau protocol de traçage de proximité sécurisé et respectueux de la vie privée, appelé SPOT. Elle propose également une analyse des performances et des améliorations introduites lorsqu'un nouveau schéma de signature de groupe offrant une vérification par lot est utilisé.

A.5.1 Architecture

SPOT implique quatre principaux acteurs, notamment l'utilisateur ($\mathcal{U}$), l'autorité de santé ($\mathcal{HA}$), le serveur ($\mathcal{S}$) et un group de proxies ($\mathcal{P}$) gérés par un gestionnaire de groupe ($\mathcal{GM}$). $\mathcal{U}$ dispose d'une application de traçage de proximité installé sur son smartphone, à partir de laquelle il s'enregistre à $\mathcal{HA}$. $\mathcal{U}$ diffuse son EBID (Ephemeral Bluetooth IDentifier) et collecte les EBID des autres utilisateurs à proximité afin d'être notifié en cas de contact avec des personnes infectées. $\mathcal{S}$ et $\mathcal{P}$ interviennent pour garantir la cohérence et l'intégrité des informations de contact des utilisateurs. Les

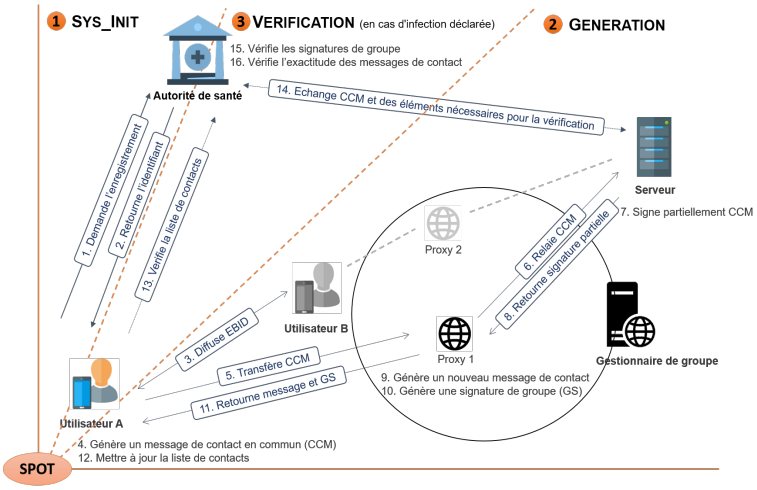


Figure A.4: Architecture de SPOT et principales interactions entre les entités

proxies jouent également un rôle important pour préserver la vie privée des utilisateurs grâce aux signatures de groupe. En cas d'infection d'un utilisateur, $\mathcal{HA}$ intervient pour vérifier l'exactitude des information de contact fournies par l'utilisateur en question.

L'architecture de SPOT, représentée dans la Figure A.4 inclut trois phases, à savoir SYS_INIT, GENERATION et VERIFICATION. Il s'agit d'une architecture hybride qui inclut un serveur centralisé et des proxies distribués sur différentes zones géographiques.

SYS_INIT – Cette phase consiste à initialiser l'ensemble du système. En effet, une autorité de confiance ($\mathcal{TA}$) génère les paramètres publics du système partagés avec toutes les entités du système et la paire de clés de $\mathcal{HA}$ et de $\mathcal{S}$. Au cours de cette phase, le gestionnaire de groupe définit le groupe de proxies. Il génère les paramètres publics du groupe et interagit aussi avec chaque membre du groupe (proxy) pour générer et certifier ses clés. $\mathcal{HA}$ intervient également dans cette phase pour enregistrer un utilisateur lors de l'installation de l'application de traçage de proximité. $\mathcal{HA}$ génère une valeur secrète spécifique $t_{\mathcal{U}}$ (connue uniquement par $\mathcal{HA}$) et un identifiant unique $\mathtt{ID}_{\mathcal{U}}$ pour chaque utilisateur ($\mathcal{U}$). Enfin, $\mathcal{U}$ utilise son identifiant pour générer sa paire de clés. L'identifiant de l'utilisateur $\mathtt{ID}_{\mathcal{U}}$, la valeur secrète $t_{\mathcal{U}}$ et la clé publique sont stockés dans une base de données DB_{USER} appartenant à $\mathcal{HA}$.

GENERATION – Cette phase se produit lorsque deux utilisateurs $\mathcal{U}_A$ et $\mathcal{U}_B$ sont en contact. Elle représente le processus de génération des messages de contact et des listes de contacts pour les utilisateurs. Trois entités principales participent à cette phase. Premièrement, $\mathcal{U}_A$ et $\mathcal{U}_B$ génèrent un message de contact en commun ($\mathtt{CCM}^e_{AB}$) en se basant sur leurs $EBID$ aléatoires pendant une époque e, où e désigne une période de temps pendant laquelle l'EBID reste inchangé. Deuxièmement, $\mathcal{U}_A$ et $\mathcal{U}_B$ choisissent deux proxies différents de la même zone géographique pour relayer $\mathtt{CCM}^e_{AB}$ au serveur. Pour ce faire, ils comparent leurs $EBID$. Si $\mathcal{U}_A$ possède l'$EBID$ le plus élevé, il choisit le premier proxy et $\mathcal{U}_B$ le second, et vice versa. Troisièmement, $\mathcal{S}$ vérifie s'il reçoit deux copies similaires du même message. Si c'est le cas, $\mathcal{S}$ signe partiellement le $\mathtt{CCM}^e_{AB}$, prouvant ainsi que le message de contact a correctement atteint le serveur. Finalement, chaque proxy étend le message, donné par $\mathcal{S}$, avec l'identifiant

de l'utilisateur correspondant et signe le message résultant au nom du groupe. $\mathcal{P}$ renvoie le message et la signature de groupe correspondante à l'utilisateur et ferme la session de communication, en supprimant toutes les informations de contact échangées et générées. A la fin de cette phase, chaque utilisateur ajoute la signature de groupe, ainsi que le message de contact commun, la date, l'heure et la durée du contact, dans sa liste de contacts $CL_{\mathcal{U}}$. Notez que chaque information de contact est stockée pendant Δ jours.

VERIFICATION – Cette phase est exécutée par $\mathcal{HA}$ pour vérifier l'exactitude des listes de contacts des utilisateurs infectés pendant une période de temps t. Pour ce faire, $\mathcal{HA}$ effectue trois vérifications successives. Lors de la première vérification, $\mathcal{HA}$ vérifie si, dans sa base de données DB_{USER}, $\mathcal{U}$ est infecté. Pour les deux autres vérifications, nous proposons deux options, (i) une vérification naïve pour vérifier l'exactitude d'un seul message de contact et (ii) une vérification par lot pour vérifier l'exactitude de plusieurs messages de contact appartenant au même utilisateur ou à des utilisateurs différents.

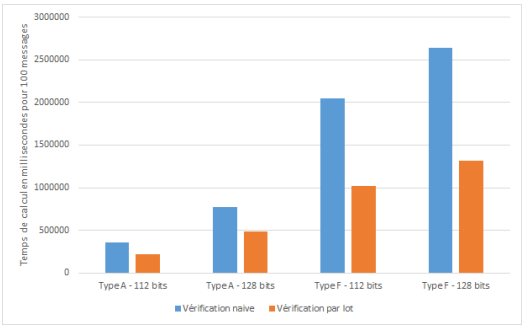
- *Vérification naïve :* $\mathcal{HA}$ considère un seul message de contact. Elle vérifie d'abord la validité d'une signature de groupe unique. Ensuite, $\mathcal{HA}$ vérifie qu'un message de contact a été correctement généré et vérifié en temps réel par $\mathcal{S}$.
- *Vérification par lot :* $\mathcal{HA}$ considère plusieurs messages de contact provenant du même utilisateur ou d'utilisaterus différents, en une seule vérification. Si la vérification par lot de plusieurs signatures échoue, $\mathcal{HA}$ doit procéder progressivement en divisant la liste de contacts en sous-listes, puis en vérifiant la sous-liste invalide message par message.

Notez que si l'une des vérifications décrités ci-dessus échoue, le message de contact est rejeté. Sinon, $\mathcal{HA}$ rassemble tous les messages vérifiés de tous les utilisateurs infectés dans un ensemble $S_{\texttt{CCM}}$ qu'elle signe. $S_{\texttt{CCM}}$ et la signature correspondante sont envoyés au serveur qui les partage avec tous les utilisateurs. Pour calculer le score de risque, chaque utilisateur compare l'ensemble $S_{\texttt{CCM}}$ avec sa liste de contacts, en tenant compte du nombre d'utilisateurs infectés contactés et de la durée du contact.

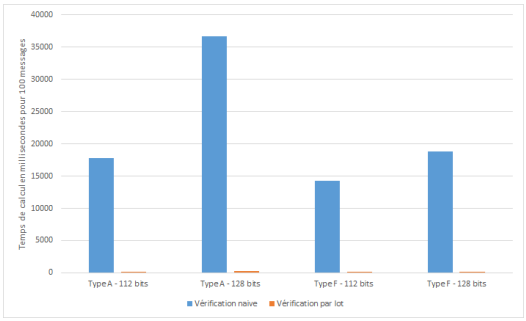
A.5.2 Propriétés satisfaites

SPOT est démontrée satisfaire les propriétés de sécurité et de vie privée suivantes :

- **Infalsifiabilité** – garantit qu'un utilisateur malveillant n'est pas en mesure de falsifier sa liste de contacts. C'est-à-dire il n'est pas capable de falsifier une signature de groupe ou une signature partielle du serveur en cas de collusion avec un proxy malveillant.
- **Non-associabilité** – peut être divisée en deux sous-propriétés. La première constitue la propriété de *non-associabilité des signatures de groupe* indiquant qu'une autorité de santé curieuse n'est pas capable de lier deux ou plusieurs signatures de groupe émises par le même proxy pendant la phase VERIFICATION. La deuxième



(a) Vérification de la validité des signatures de groupe



(b) Vérification de l'exactitude des messages de contact

Figure A.5: Temps de calcul de la vérification par lot vs vérification naïve sur 100 messages

sous-propriété de *non-associabilité de multi-CCM* garantit qu'un serveur curieux n'est pas en mesure de lier deux ou plusieurs messages de contact en commun au même utilisateur pendant la phase GENERATION.

- **Anonymat** – garantit qu'une autorité de santé curieuse n'est en mesure d'identifier les utilisateurs impliqués dans une liste de contacts d'une personne infectée, pendant la phase VERIFICATION. Pour cette propriété, nous supposons que la probabilité que deux utilisateurs confirmés soient en contact et soumettent leurs listes de contacts respectives à $\mathcal{HA}$ à la même période, est faible.

- **Anti-rejeu** – garantit qu'un adversaire malveillant n'est pas capable de soumettre la même information de contact sur plusieurs sessions.

A.5.3 Analyse de performances

Pour mettre en œuvre les trois phases SYS_INIT, GENERATION et VERIFICATION, les algorithmes de SPOT ont été implémentés sur une machine fonctionnant sous Ubuntu version 20.04.1 *LTS* - avec un processeur 8 cœurs et $16GB$ de mémoire, en utilisant le langage de programmation JAVA version 11, et de la bibliothèque cryptographique *JPBC*[2]. Nous évaluons le temps de calcul de chaque algorithme en nous appuyant sur deux types de pairings bilinéaires, à savoir *type A* et *type F*. Pour les deux types de pairings, nous considérons deux niveaux de sécurité différents, à savoir les niveaux de sécurité de 112-bits et de 128-bits.

Dans cette section, nous nous concentrons sur la phase VERIFICATION. Nous présentons une analyse comparative du temps de calcul de la vérification par lot par rapport à la vérification naïve. Pour ce faire, nous considérons 100 messages de contact partiellement et entièrement signés et nous comparons les temps de calcul d'une vérification naive effectuée 100 fois sur les 100 messages et les signatures correspondantes et une vérification par lot exécutée une seule fois sur les 100 messages.

[2] http://gas.dia.unisa.it/projects/jpbc/

Figure A.5 confirme que la vérification par lot est plus efficace que la vérification naïve. D'une part, comme le montre la sous-figure A.5a, la vérification par lot des signatures de groupe réduit le temps de calcul d'environ 37%, pour le pairing *type A* avec les deux niveaux de sécurité. Pour le pairing *type F*, le gain atteint 50% pour les deux niveaux de sécurité. D'autre part, la sous-figure A.5b montre que la vérification par lot sur 100 signatures partielles, réduit le temps de calcul de 99% pour les deux types de pairing avec les deux différents niveaux de sécurité, ce qui prouve son efficacité.

A.5.4 Conclusion

Ce chapitre présente un nouveau protocole de traçage de proximité Spot sécurisé et préservant la vie privée pour les systèmes d'e-santé. L'objectif de Spot est d'aider les gouvernements et les systèmes de santé à faire face aux pandémies en automatisant le processus de recherche des contacts, avec des garanties de sécurité contre l'injection de faux contacts et la préservation de la vie privée des utilisateurs. Une implémentation complète de Spot démontre sa faisabilité avec des temps de calcul raisonnable, surtout en utilisant une vérification par lot sur plusieurs messages de contact.

A.6 Conclusions et perspectives

Cette thèse a remis en question de nombreuses défis relatifs aux systèmes de gestion des identités, principalement en termes de sécurité, de vie privé et d'utilisabilité. Notre objectif principal était de trouver le meilleur compromis entre ces trois exigences. Dans ce travail, trois solutions concrètes de gestion des identité ont été proposées pour répondre à cet objectif.

La première solution représente un nouveau système de gestion des identités centré sur l'utilisateur et respectueux de la vie privée, permettant de contrôler l'accès des utilisateurs aux services et aux ressources [94]. Le système proposé, appelé Pima, est basé sur un nouveau schéma de signature malléable. Afin de satisfaire l'*Objectif A* qui consiste à fournir aux utilisateurs le contrôle total de leurs identités et de leurs attributs, un utilisateur reçoit des attributs certifiés d'un fournisseur d'identités de confiance. Il est ensuite capable de sélectionner et de divulguer uniquement les informations nécessaires aux fournisseurs de services. En réponse à l'*Objectif B*, Pima permet aux utilisateurs d'effectuer des modifications contrôlées et restreintes sur leurs attributs, tandis que les fournisseurs de services peuvent toujours vérifier l'authenticité des données fournies, dans des sessions pseudonymisées. Différents pseudonymes sont utilisés par les différents fournisseurs de services afin d'atteindre l'*Objectif D*. En effet, les fournisseurs de services ne sont pas en mesure de relier plusieurs transactions d'un même utilisateur entre elles, ni de retrouver sa véritable identité.

La deuxième solution est appelée Puba. Il s'agit d'un schéma d'authentification biométrique respectueux de la vie privée pour les systèmes de gestion des identités. Pour s'enregistrer de façon anonyme auprès d'un fournisseur de services, l'utilisateur peut modifier des modèles biométriques chiffrés et certifiés, reçus d'un fournisseur

d'identités, en réponse à l'*Objectif A*. Pour l'authentification, l'utilisateur est invité à fournir un nouveau modèle biométrique lorsqu'il se présente chez le fournisseur de services. En comparant les deux modèles biométriques à travers un calcul collaboratif entre l'utilisateur et le fournisseur de services, ce dernier décide d'authentifier ou non l'utilisateur. Afin de satisfaire l'*Objectif B*, la conception de PUBA permet de vérifier la validité des données biométriques et l'exactitude des calculs effectués par les utilisateurs. Nous nous sommes appuyés sur des caractéristiques de polymorphisme et de randomisation pour empêcher les fournisseurs de services de relier plusieurs sessions d'enregistrement du même utilisateur ou de déduire son identité réelle, en réponse à l'*Objectif D*.

Les deux solutions ont été implémentées pour répondre à l'*Objectif F* qui consiste à mettre en œuvre les schémas proposés et à valider leur faisabilité sur du matériel réel. Les résultats expérimentaux ont démontré l'efficacité de PIMA et PUBA. D'une part, les algorithmes exécutés par le fournisseur d'identités ou le fournisseur de services ont été testés sur un ordinateur portable doté de capacités matérielles acceptables. D'autre part, les tests des algorithmes exécutés par l'utilisateur, ont été réalisés sur un smartphone. Sur les deux appareils, des temps de calcul raisonnables ont été obtenus.

La troisième solution a été conçue pour répondre à l'*Objectif C* qui consiste à concevoir un mécanisme sécurisé pour la gestion des identités éphémères. Il s'agit d'un protocole de traçage de proximité respectueux de la vie privée pour les systèmes d'e-santé, appelé SPOT. Cette solution repose sur une architecture hybride. Afin de satisfaire l'*Objectif A*, l'utilisateur est d'abord responsable du calcul des messages de contact lorsqu'il se trouve à proximité d'autres utilisateurs. Ensuite, tout en s'appuyant sur un serveur centralisé et des proxies décentralisés, l'utilisateur collecte des informations de contact certifiées qu'il stocke localement. En cas d'infection, l'utilisateur partage ces informations avec l'autorité de santé, qui vérifie leur exactitude avant de les envoyer aux autres utilisateurs. SPOT permet d'éviter l'injection de fausses alertes positives et de se prémunir contre les attaques d'associabilité et d'identification, en réponse à l'*Objectif D*. Une mise en œuvre du protocole proposé avec un effort d'agrégation des calculs, a démontré l'efficacité de SPOT. Cette implémentation a été testée dans un démonstrateur à travers deux scénarios, permettant de satisfaire l'*Objectif F*.

Les trois solutions proposées satisfont l'*Objectif E* qui consiste à prouver que les constructions proposées satisfont les propriétés de sécurité et de vie privée souhaitées. Pour atteindre cet objectif, des preuves ont été présentées pour chaque schéma proposé, en s'appuyant sur des hypothèses de calcul standards et en faisant référence à des jeux de sécurité détaillés.

Nos perspectives de recherche comprennent:

- une extension pour PIMA, où de multiples fournisseurs d'identités sont considérés. Dans ce cas, un utilisateur sera capable de prouver à un fournisseur de services différents attributs certifiés par différents fournisseurs d'identités. Ainsi, il serait intéressant d'étudier les capacités de l'utilisateur à agréger plusieurs attributs fournis par plusieurs fournisseurs d'identités sous le même pseudonyme. Deux

méthodes pourraient être explorées et évaluées en fonction des exigences de sécurité, de vie privée et de performance. La première consiste à agréger plusieurs signatures malléables générées par différents fournisseurs d'identités pour le même utilisateur. Il serait donc important de définir la ou les clés utilisées par les fournisseurs de services pour vérifier l'authenticité des attributs des utilisateurs. La deuxième solution considère que tous les fournisseurs d'identités appartiennent au même groupe de signataires. À ce titre, un schéma de signature de groupe malléable pourrait être proposé. Il serait également intéressant d'étudier la possibilité d'agréger plusieurs signatures de groupe malléables, de sorte que le fournisseur de services vérifie l'authenticité des attributs des utilisateurs en une seule transaction à l'aide de la clé publique du groupe. L'impact des deux solutions proposées sur la vie privée des utilisateurs devrait être étudié, à savoir la non-associabilité, à la fois, entre les fournisseurs de services et les fournisseurs d'identité.

- une extension de Pima vers un système de gestion des identités centré sur l'utilisateur où la gouvernance est décentralisée. En effet, il serait intéressant d'étudier l'utilisation du schéma $\mathcal{UMS}$ dans un environnement distribué comme la blockchain.

- une solution qui permet de vérifier en ligne une authentification biométrique, en respectant les législations européennes et internationales. En effet, face à la nécessité croissante de prouver le lien avec la personne physique afin d'éviter les fraudes, il serait intéressant de concevoir un mécanisme d'authentification biométrique sûr, fiable et préservant la vie privée sans qu'il soit nécessaire d'être présent chez le fournisseur de services. Il serait également important d'étendre la solution pour garantir l'audit et la révocation des credentials biométriques, par une autorité de confiance, dans le but de révéler l'identité des utilisateurs responsables de comportements malveillants et de révoquer les certificats, par exemple en cas de vol d'identités.

- une solution qui assure le lien entre les attributs des utilisateurs et leurs identités biométriques, en s'appuyant sur l'authentification biométrique Puba. En effet, après avoir vérifier l'identité biométrique d'un utilisateur lors de l'authentification, il serait intéressant de prouver l'authenticité des attributs d'un utilisateur lorsqu'ils sont associés à cette identité. La possibilité d'agréger, sous la même identité biométrique, plusieurs attributs fournis par différents fournisseurs d'attributs, pourrait également être étudiée.

- des évaluations expérimentales supplémentaires pour Spot sur des dispositifs avec des capacités matérielles avancées. En effet, il serait intéressant d'évaluer l'ordre de grandeur de certains paramètres du système. Par exemple, le temps consommé par le serveur pour attendre la réception de deux exemplaires identiques du message de contact, pourrait être évalué par rapport au nombre de connexions acceptées par le serveur et au nombre de connexions demandées.

- une évaluation des coûts de communication de Spot, tout en faisant varier des paramètres tels que la distance entre l'utilisateur et le proxy chargé de transmettre ses informations de contact. En se basant sur cette évaluation, il serait intéressant

de définir l'étendu des zones géographiques et de déterminer le nombre de proxies nécessaires.

DOCTORAL SCHOOL

Titre : Des technologies malléables préservant la vie privée pour des systèmes de gestion des identités respectueux de la vie privée

Mots clés : sécurité, identités numériques, gestion des identités, cryptographie, protection des données, utilisabilité

Résumé :
Cette thèse vise à trouver le meilleur compromis entre sécurité, vie privée et utilisabilité pour les systèmes de gestion des identités, en s'appuyant sur des primitives cryptographiques. Les deux premières contributions s'intéressent à la gestion des identités pour le contrôle d'accès et considèrent des identités et attributs réels qui contiennent des informations personnelles (ex : âge) et sensibles (ex : caractéristiques biométriques).
Pour répondre à cette préoccupation, la première contribution propose un système de gestion des identités centré sur l'utilisateur et respectueux de la vie privée, appelé PIMA, dans lequel les utilisateurs gardent le contrôle sur leurs attributs. Un utilisateur, qui reçoit des attributs certifiés par un fournisseur d'identité, peut interagir de façon pseudonymisée avec un fournisseur de services et lui prouver l'authenticité des attributs présentés tout en minimisant le nombre de ces attributs. Cette solution s'appuie sur un nouveau schéma de signature malléable qui permet aux utilisateurs de transformer le certificat issu sur ses attributs de façon restreinte et contrôlée. Elle préserve aussi la vie privée en satisfaisant les propriétés de non-associabilité entre des fournisseurs de services curieux qui tenteraient d'associer différentes transactions à un même utilisateur.
La deuxième contribution porte sur un nouveau schéma d'authentification biométrique, appelé PUBA, qui offre des garanties de robustesse et de respect de la vie privée. Trois étapes sont nécessaires. Tout d'abord, l'utilisateur se rend physiquement chez le fournisseur d'identités qui pousse le modèle biométrique chiffré et certifié sur son smartphone. Puis il s'enregistre à distance auprès d'un fournisseur de services, de façon anonyme. Enfin, il s'authentifie hors ligne auprès du fournisseur de services qui capture la modalité biométrique, cette modalité étant vérifiée localement via le smartphone. En s'appuyant sur des signatures malléables, la solution proposée empêche l'utilisation de fausses identités biométriques et garantit la fiabilité de l'authentification. La non-associabilité et l'anonymat, sont aussi préservées. La troisième contribution apporte une solution au besoin de partager des données dans un système de gestion des identités, et en particulier étudie la gestion des attributs éphémères des utilisateurs dans le contexte du traçage de proximité pour les systèmes d'e-santé. La solution proposée, appelée SPOT, assure la cohérence et l'intégrité des données et préserve la vie privée des utilisateurs qui partagent leurs informations de contact avec les personnes à proximité. Des alertes sont émises vers les personnes ayant été en contact avec des personnes infectées. L'architecture hybride utilisée qui repose sur un serveur centralisé et des proxies décentralisés empêche les utilisateurs malveillants d'injecter de fausses alertes, et empêche de relier toute information de contact à un même utilisateur et de réidentifier les utilisateurs impliqués dans un contact avec une personne infectée.

Title : Malleable Privacy-Enhancing-Technologies for Privacy-Preserving Identity Management Systems

Keywords : security, digital identities, identity management, cryptography, data protection, usability

Abstract :
This thesis aims at finding the best trade-off between security, privacy and usability for identity management systems, based on cryptographic primitives. The first two contributions focus on identity management for access control and consider real identities and attributes that contain personal (e.g., age) and sensitive (e.g., biometric traits) information.
The first contribution proposes a user-centric and privacy-preserving identity management system, named PIMA, in which users keep control over their attributes. A user, that receives attributes certified by an identity provider, is able to interact, in a pseudonymized manner, with a service provider and prove the authenticity of the provided attributes while ensuring that he discloses only the minimum number of attributes. This solution is based on a new malleable signature scheme that allows users to modify the certificate issued on his attributes in a restricted and controlled manner. It also preserves privacy by satisfying the unlinkability property between curious service providers that try to link different transactions to the same user.
The second contribution presents a new biometric authentication scheme, named PUBA, that offers robustness and privacy guarantees. Three steps are required. First, the user physically visits the identity provider that pushes an encrypted and certified biometric template onto his smartphone. Then he remotely enrolls at a service provider, in an anonymous manner. Finally, he authenticates offline to the service provider that captures a new biometric template in order to be locally verified via the smartphone. By relying on malleable signatures, this solution prevents the use of fake biometric identities and guarantees the authentication soundness. Unlinkability and anonymity are also preserved, even when considering a collusion between curious identity and service providers.
The third contribution provides a solution to meet the need of data sharing in an identity management system. In particular, it studies the management of users ephemeral attributes in the context of proximity tracing for e-healthcare systems. The proposed solution, named SPOT, ensures data consistency and integrity and preserves the privacy of users who share their contact information with people in proximity. Alerts are issued to users who have been in contact with infected persons. The use of a hybrid architecture, which relies on a centralized server and decentralized proxies, allows to prevent malicious users from injecting false alerts, and to prevent the linkability of contact information to the same user and the re-identification of users involved in contact with an infected person.

Institut Polytechnique de Paris
91120 Palaiseau, France