Contextual Identity in Practice

Attribute-based Authentication and Signing on Smartphones

Brinda Hampiholi

ISBN: 978-94-92801-80-7
NUR: 980

Typeset using LaTeX 2_ε (TeXLive 2017)
Based on template by Pim Vullers (`https://github.com/pimvullers/phd_thesis`)
Cover design by Brinda Hampiholi and Manoj Kumar Sharma
Printed by Proefschrift All In One (AIO) (`https://www.proefschrift-aio.nl`)

The author was employed at the Radboud University Nijmegen. This research is funded by the Netherlands Organization for Scientific Research (NWO) and supported by KPN as project ‘Own Your Own Identity (OYOI)’.

Contextual Identity in Practice

Attribute-based Authentication and Signing on Smartphones

Proefschrift

ter verkrijging van de graad van doctor
aan de Radboud Universiteit Nijmegen
op gezag van de rector magnificus prof. dr. J.H.J.M. van Krieken,
volgens besluit van het college van decanen
in het openbaar te verdedigen op **dinsdag 28 mei 2019**
om **14.30 uur** precies

door

Brinda Badarinath Hampiholi

geboren op 20 juni 1989
te Bangalore, India

Promotor:

Prof. dr. B.P.F. Jacobs	

Copromotor:

Dr. G. Alpár	Open Universiteit

Manuscriptcommissie:

Prof. dr. L. Batina	
Prof. dr. S. Fischer-Hübner	Karlstad Universitet, Zweden
Dr. A. Lehmann	IBM Research Zürich, Zwitserland
Dr. S. Gürses	KU Leuven, België
Prof. dr. A. Sasse	University College London, VK

Contextual Identity in Practice

Attribute-based Authentication and Signing on Smartphones

Doctoral Thesis

to obtain the degree of doctor
from Radboud University Nijmegen
on the authority of the Rector Magnificus, prof. dr. J.H.J.M. van Krieken,
according to the decision of the Council of Deans
to be defended in public on **Tuesday, May 28, 2019**
at **14.30 hours**

by

Brinda Badarinath Hampiholi

born on June 20, 1989
in Bangalore, India

Supervisor:

Prof. dr. B.P.F. Jacobs

Co-supervisor:

Dr. G. Alpár	Open University

Doctoral Thesis Committee:

Prof. dr. L. Batina	
Prof. dr. S. Fischer-Hübner	Karlstad University, Sweden
Dr. A. Lehmann	IBM Research Zürich, Switzerland
Dr. S. Gürses	KU Leuven, Belgium
Prof. dr. A. Sasse	University College London, UK

Acknowledgements

Back in 2014, when I had just finished my Masters at University of Twente, four years of PhD had seemed very long to me. But, time flew so fast that it was almost unbelievable when my stint at Radboud University as a PhD student was over in 2018. Looking back, I truly feel that it has been an incredible journey filled with many learning experiences, motivating discussions, collaborations, travels, and celebrations. I would like to thank all the people who inspired and supported me during my PhD.

First of all, I would like to thank my supervisor, Bart for suggesting this PhD project and being a really good mentor to me. You have supported me in my research and have always been encouraging. I appreciate your quick response to emails, prompt and critical reviews on my work and the enthusiasm you showed for new ideas and initiatives. I will always remember your strategic and intuitive questions during our meetings. You not only inspired me but also made me more diligent and responsible towards the project. Thank you, Bart.

Next I would like to thank my co-supervisor, Gergely for motivating me and sincerely guiding me throughout my PhD. You helped me in writing the articles and in honing my scientific writing and analytical skills. I feel happy and grateful for all the stimulating and productive discussions we have had over the years. I cannot thank you and Bart enough for reviewing all the chapters of my thesis and providing me with a lot of helpful feedback that improved the final result.

I would like to extend my gratitude to the members of my thesis committee for reading and positively evaluating my thesis. In particular, I thank Simone and Anja for their valuable feedback on specific parts of the thesis. Thank you Lejla for coordinating with the committee members during the thesis evaluation and helping me to organise the defense.

I cherished the time spent in the Digital Security group at Radboud University. It was wonderful because of the nice colleagues I had there. Thanks to Fabian and Freek for being really great office-mates. Be it about research, travel, daily activities or about our common interest: food and cooking, our conversations were always enjoyable. Fabian, thank you for all your support during these years including several interesting brainstorming sessions and discussions we have had on IRMA-related topics. My heartfelt thanks also goes to Wouter for being an insightful colleague and a great co-author. You have patiently answered my questions, presented me with thought-provoking excercises during our discussions, worked very hard on our joint-paper and have inspired me with your determination even when things did not work as we had expected.

During my PhD, I worked very closely with Bart, Gergely, Fabian and Wouter and I learnt a lot from them. You have been kind, inspirational and contructively critical for which I am ever grateful. I enjoyed being a part of the IRMA team (now known as the Privacy by Design foundation). Thanks to Fabian, Sietse, Koen, David, Hanna and Bart for all the interesting discussions we had on the secure implementation and usability aspects of the IRMA application.

I also thank Irma Haerkens for being such a sweet person and helping me in many ways since the beginning of my PhD.

Furthermore, I would like to thank Peter, Veelasha, Ko, Pedro, Joost Renes, Joost Rijneveld and Freek for being super-cool colleagues who flew all the way from the Netherlands to India to attend my wedding. It was such a great pleasure to have you all there. The time we spent together on several occasions, such as department outings, Croatia summer school, Real World Crypto was so much fun and memorable. Other dear colleagues with whom I had many interesting conversations while I was at the university were Ronny, Engelbert, Louiza, Anna, Jeroen, Paulus, Merel, Kostas, Erik Poll and the post-quantum crypto group members. All of you contributed to the good times that I had as a PhD student. Thank you very much.

I had an opportunity during my PhD to pursue an internship for three months at Telekom Innovation Laboratories (T-Labs), Berlin. I thank Martin Kurze for the supervision and all the colleagues there who made my stay invigorating and memorable. When the four years of my PhD ended, I started working at Philips Research, Eindhoven. It has been a pleasure being a part of the security group in the Data Science department at Philips. I would like to thank my colleagues for the research-oriented and amicable work atmosphere. Special thanks to Peter van Liesdonk for helping me with the Dutch translation of my PhD thesis summary and improving it.

I would like to thank my dear friends: Aicha, Anvitha, Federico, Claudia, Calle, Ulrike, Andreas, Cristian, Ági, Nilkanth, Namrata, Harsh, Abhishek, Amit, Shweta with whom I have spent many joyous times, travelled together, and shared vibrant and colourful experiences outside PhD and work contexts.

Final words of gratitude are dedicated to my family for always being there for me. Especially my mother and grandmother who have supported each and every endevour in my life since my birth and Manoj, my husband who has been a constant source of love, inspiration, strength and happiness to me for all the years we have been together.

Brinda Hampiholi
Eindhoven, April 2019

Abstract

Context plays a big role in how people behave and present themselves to others. A context can be defined by where, when and with whom they are. For instance, an individual introduces herself with her name and designation at an office meeting whereas at a friend's party, she may introduce herself to the other guests by saying her nickname and that she is a close friend of the host. In the physical world, a person can consciously choose her *contextual identity*, which is the identity information she reveals about herself based on the context. In principle, a person should be able to do the same in the digital world. However, the existing digital systems and services do not support it. On the contrary, many of them collect uniquely identifying and/or excessive personal information from individuals irrespective of the transaction context. This results in an overspill of users' personal information, which in turn has many undesirable privacy consequences for users, such as user profiling, tracking, unintended identity disclosures, identity fraud. Clearly, there is a need for adopting technologies that can put users back in control over the disclosure of their own identity information in different digital contexts.

Attributes of a person, for instance, 'a student', 'older than 18 years', 'name', 'social security number' provide a natural mechanism to achieve contextual identities. A human user can obtain attribute-based credentials (ABCs) from an authorised issuer that can vouch for the validity (for this user) of the attributes contained in the credential. An ABC is a cryptographic container of attributes that are bound to the user via a user-specific secret key. The user can authenticate by selectively disclosing attributes from a credential to an online service provider and get access to the service. Depending on the nature of the disclosed attributes, the user can be anonymous, pseudonymous or uniquely identifiable to a service provider. ABCs are more flexible than traditional identity management technologies that invariably use unique identifiers of users. The versatility of ABCs motivates the research work done in this thesis.

The overall objective of the thesis is to enable practical and secure use of contextual identities via ABCs in digital transactions. In particular, the thesis focuses on how to better utilise ABCs on smartphones and tackles some of the challenges influencing their real-world adoption. Traditionally, ABCs have been considered mostly for authentication purpose. The first part of the thesis, consisting of Chapters 3 and 4, describes real-world applications of ABCs such as digital signing and carrying out multi-step transactions (online shopping). This demonstrates that ABCs are useful for more purposes than authentication. Along with usefulness, ABCs need

to provide strong security and trust assurances to users and service providers. The second part of the thesis, consisting of Chapters 5 and 6, discusses the techniques to enhance security and trust in the ABC applications without reduction in the privacy, and several bootstrapping methods which facilitate the deployment of ABCs in real-world use cases.

Chapter 3 shows that the utility of ABCs can be extended from only authentication to other important user actions including digital signatures. It presents how to realise timestamped attribute-based signatures (ABSs) in practice along with attribute-based authentication using the same set of attributes in a secure way. ABSs are great alternatives to the prevalent public-key signatures as they provide the same security level while offering flexibility and privacy benefits to signers. The benefits include the ability to include only a subset of their personal attributes in the signatures and be unlinkable from their signatures. For example, when a signer includes only her citizenship attribute in an ABS, then she cannot be uniquely linked to this ABS and neither can this ABS be linked with her other ABSs.

Chapter 4 explores how attribute-based credentials can be used in multi-step transactions, in particular in shopping transactions consisting of cart creation, payment and delivery phases. The main aim is to enforce that a minimum amount of data is collected from the user in every step of the transaction. This chapter describes privacy-preserving protocols for carrying out attribute-based webshopping transactions in which attribute-based signatures without timestamps are used. Such an application of ABSs shows that they can be used for more purposes than conventional digital signing of documents (e.g. contracts, agreements).

Using ABCs on smartphones is user friendly and efficient but there are also concerns about the security of cryptographic secrets when stored on insecure phones. The security of a user's secret key that is bound to her ABCs is crucial for secure and trustworthy operation of the entire ABC system. Chapter 5 considers a basic key-sharing scheme between a user's phone and a central server to secure the user's secret key. This enforces instantaneous blocking of the key when the phone is lost or stolen. However, if the key-sharing server cannot be fully trusted, it may collude with service providers and compromise users' privacy provided by ABCs. This collusion exploits the presence of side channel information, such as time of key access at the server and its use at a service provider. To counter this, a new scheme called Tandem is proposed. Tandem is a set of protocols that augment threshold cryptography to ensure the security of the users' secret keys while maintaining users' privacy towards a malicious server.

User enrolment is the first step in deploying any new technology. Chapter 6 deals with secure, trustworthy and user-friendly methods for user enrolment in an ABC ecosystem. It discusses secure self-enrolment protocols through which users can enrol themselves online and obtain trustworthy ABCs onto their phones. It is expected that this way of bootstrapping ABCs will increase the trust assurance provided by ABCs and bring the attribute-based technology closer to being adopted by service providers and users.

To conclude, users can authenticate, sign and carry out attribute-based transactions with ABCs by disclosing only the personal attributes that are relevant to the

context. This thesis demonstrates the practical applicability of ABCs and ensures that their use is not only convenient and privacy-friendly but also secure and trustworthy. In the current digital arena where end users as well as privacy regulations such as, the European GDPR are demanding privacy and user control over personal data, it would be beneficial to businesses if they adopt attribute-based technology for their operations concerning user identities.

Samenvatting

Context speelt een grote rol in hoe mensen zich gedragen en presenteren aan anderen. Een context kan worden gedefinieerd door waar, wanneer en met wie ze zijn. Bijvoorbeeld: een individu introduceert zichzelf met haar achternaam en functie op een vergadering, terwijl ze zich op het feest van een vriend aan de andere gasten voorstelt met haar voornaam en vertelt dat zij een goede vriend van de gastheer is. In de fysieke wereld kan een persoon bewust kiezen voor haar *contextuele identiteit*, de identiteitsinformatie die ze over zichzelf onthult op basis van de context. In de digitale wereld zou een persoon in principe hetzelfde moeten kunnen doen. De bestaande digitale systemen en diensten ondersteunen dit echter niet. Integendeel, veel van dit soort systemen verzamelen unieke identificerende en/of overmatig persoonlijke informatie van individuen, ongeacht de context van de transactie. Dit resulteert in een surplus aan persoonlijke gegevens van gebruikers, wat op zijn beurt veel ongewenste gevolgen heeft voor de privacy van gebruikers, zoals gebruikersprofilering, tracking, onbedoelde openbaarmaking van identiteit en identiteitsfraude. Het is duidelijk dat er technologieën nodig zijn die controle over de openbaarmaking van identiteitsinformatie in verschillende digitale contexten in de handen van de desbetreffende gebruiker leggen.

Kenmerken van een persoon (attributen) bieden een natuurlijk mechanisme om contextuele identiteiten te bereiken; bijvoorbeeld 'een student', 'ouder dan 18 jaar', 'naam', 'sofinummers'. Een menselijke gebruiker kan op attributen gebaseerde inloggegevens (attribute-based credentials, of ABC's) verkrijgen van een geautoriseerde uitgever. Deze uitgever staat vervolgens in voor de geldigheid van de kenmerken voor deze gebruiker. Zo'n ABC is een cryptografische container met attributen die aan de gebruiker zijn gebonden via een gebruiker-specifieke geheime sleutel. De gebruiker kan zichzelf identificeren door selectief attributen aan een online serviceprovider bekend te maken en toegang te krijgen tot de service. Afhankelijk van de aard van de openbaar gemaakte attributen, kan de gebruiker anoniem, pseudoniem of uniek identificeerbaar zijn voor een serviceprovider. ABC's zijn flexibeler dan traditionele technologieën voor identiteitsbeheer die steevast unieke identificatiegegevens van gebruikers gebruiken. De veelzijdigheid van ABC's motiveert het onderzoekswerk in dit proefschrift.

Het doel van dit proefschrift is om praktisch en veilig gebruik van contextuele identiteiten via ABC's mogelijk te maken in digitale transacties. In het bijzonder richt het proefschrift zich op het verbeteren van het gebruik van ABC's op smartphones, en op het aanpakken van enkele van de uitdagingen die van invloed zijn op

hun toepassing in de echte wereld. Van oudsher worden ABC's meestal beschouwd voor authenticatiedoeleinden. Het eerste deel van het proefschrift, bestaande uit Hoofdstukken 3 en 4, beschrijft praktische toepassingen van ABC's zoals digitale ondertekening en het uitvoeren van meerstap transacties (online winkelen). Dit toont aan dat ABC's nuttig zijn voor meer doeleinden dan alleen authenticatie. Daarnaast moeten ABC's sterke beveiligings- en vertrouwenszekerheid bieden aan gebruikers en serviceproviders. Het tweede deel van het proefschrift, dat bestaat uit de hoofdstukken 5 en 6, bespreekt technieken om de veiligheid en het vertrouwen in ABC-applicaties te verbeteren zonder dat dat ten koste gaat van de privacy, en verschillende bootstrappingmethoden die de inzet van ABC's in praktijkgevallen vergemakkelijken.

Hoofdstuk 3 laat zien dat de toepassing van ABC's kan worden uitgebreid van puur authenticatie naar andere belangrijke acties van gebruikers, waaronder digitale handtekeningen. Hierin wordt beschreven hoe tijd-gestempelde attribuut-gebaseerde handtekeningen (attribute-based signatures, of ABS's) in de praktijk kunnen worden gerealiseerd. ABS's zijn geweldige alternatieven voor de gangbare handtekeningen voor openbare handtekeningen, omdat ze hetzelfde beveiligingsniveau bieden maar tegelijkertijd ondertekenaars flexibiliteit en privacy bieden. De voordelen omvatten de mogelijkheid om alleen een subset van hun persoonlijke attributen op te nemen in de handtekeningen en om deze attributen los te koppelen van de handtekeningen. Wanneer een ondertekenaar bijvoorbeeld alleen haar burgerschapsattribuut in een ABS opneemt, kan zij niet op unieke wijze aan dit ABS worden gekoppeld en kan dit ABS evenmin aan haar andere ABS's worden gekoppeld.

Hoofdstuk 4 onderzoekt hoe op attributen gebaseerde legitimatiegegevens kunnen worden gebruikt in meerstaptransacties, met name bij winkelfuncties die bestaan uit het maken, betalen en afleveren van wagens. Het belangrijkste doel is om ervoor te zorgen dat er bij elke stap van de transactie een minimale hoeveelheid gegevens van de gebruiker wordt verzameld. Dit hoofdstuk beschrijft privacybeschermende protocollen voor het uitvoeren van op attributen gebaseerde webwinkeltransacties, waarbij attribuut-gebaseerde handtekeningen zonder tijdstempels worden gebruikt. Een dergelijke toepassing van ABS's laat zien dat ze voor meer doeleinden kunnen worden gebruikt dan conventionele digitale ondertekening van documenten (bijvoorbeeld contracten, overeenkomsten).

Het gebruik van ABC's op smartphones is gebruiksvriendelijk en efficiënt, maar er zijn ook zorgen over de beveiliging van cryptografische geheimen wanneer deze worden opgeslagen op onveilige telefoons. De veiligheid van de geheime sleutel van een gebruiker die aan haar ABC's is gebonden, is cruciaal voor een veilige en betrouwbare werking van het volledige ABC-systeem. Hoofdstuk 5 beschrijft een basisschema voor het delen van sleutels tussen de telefoon van een gebruiker en een centrale server om de geheime sleutel van de gebruiker te beveiligen. Dit zorgt voor onmiddellijke blokkering van de sleutel wanneer de telefoon zoekraakt of wordt gestolen. Als deze server voor het delen van sleutels echter niet volledig kan worden vertrouwd, kan deze server samenwerken met serviceproviders om inbreuk op de privacy van de gebruikers te plegen. Deze collusie maakt gebruik van de aanwezigheid van nevenkanaalinformatie, zoals de tijd van belangrijke toegang op de server en het gebruik ervan bij een serviceprovider. Om dit tegen te gaan wordt een nieuw schema met de naam Tandem voorgesteld. Tandem is een reeks protocollen die drempel-cryptografie toepast om de veiligheid van de geheime sleutels van gebruikers te waarborgen, terwijl de privacy van gebruikers tegenover een kwaadwillende server wordt behouden.

De eerste stap bij het inzetten van nieuwe technologie is gebruikersregistratie. Hoofdstuk 6 behandelt veilige, betrouwbare en gebruiksvriendelijke methoden voor gebruikersregistratie in een ABC-ecosysteem. Het bespreekt veilige protocollen voor zelfinschrijving waardoor gebruikers zich online kunnen inschrijven en betrouwbare ABC's op hun telefoons kunnen krijgen. De verwachting is dat deze manier van bootstrapping van ABC's de vertrouwenszekerheid van ABC's zal vergroten en de attribuut-gebaseerde technologie dichter bij de acceptatie door serviceproviders en gebruikers zal brengen.

Tot slot kunnen gebruikers authenticatie, ondertekening en uitvoering van op attributen gebaseerde transacties met ABC's uitvoeren door alleen de persoonlijke kenmerken die relevant zijn voor de context openbaar te maken. Dit proefschrift demonstreert de praktische toepasbaarheid van ABC's en zorgt ervoor dat het gebruik ervan niet alleen handig en privacy-vriendelijk is, maar ook veilig en betrouwbaar. In de huidige digitale arena waar eindgebruikers en privacyregels (zoals de Europese GDPR) privacy- en gebruikerscontrole over persoonsgegevens eisen, zou het in het voordeel van bedrijven zijn als zij op attributen gebaseerde technologie toepassen voor hun activiteiten met betrekking tot gebruikersidentiteiten.

Contents

Chapter 1

Introduction

In the recent years, people have embraced the digital world for carrying out many activities such as, gathering information, social interaction, banking and shopping. So they regularly communicate with many online service providers. Typically, a user needs to authenticate herself to gain access to most services. For example, a user authenticates with her employee credential (e.g. username-password) to access an official document. This action convinces her employer (resource owner) that she is an employee. This is similar to showing the employee ID card to the security guard before entering the office building. The authentication step allows the service provider to check if a user is really who she says she is; this step is required to set up a trust relation between the user and the service provider during the communication.

The information about a user that a service provider needs to know to authenticate her differs from context to context. This is directly related to the notion of *contextual identity*, that is, an individual reveals different aspects of herself depending on the context [1]. At any given time, an individual may act as an employee, a service subscriber, a citizen or in some other role. Specifically, for the authentication purpose, an employer might need to know if the user is an employee, a video streaming service might need to know if the user is an active subscriber, a tax assessment service might need the user's social security number. These examples show that the information required to authenticate a user may or may not identify her. It is crucial for the user's privacy that only minimum required information about the user for the context is known to a service provider.

However, in the current digital world, many service providers (SPs) collect some personally identifying information (PII) from users during registration such as email address, and mostly use username-password mechanism for authenticating users. The username-password combination is associated with the PII of the users. The SPs uniquely identify users from their authentications and log all their actions. The unique identifiers of users known to SPs may be used to link users' actions across services, thus removing the context barriers on the user-information that is disclosed online and creating privacy risks for the users [2].

For example, a user Alice provides some bits of personal information to a video streaming service (e.g. name, email address, credit card number), sets up an authentication method (mostly, username-password) and starts using the service. The streaming service identifies Alice every time she logs in to her account to watch a video and archives the watched videos. It may analyse Alice's data (e.g. video choices, watching frequency) for improving her experience while using the service, for instance, to provide video suggestions. But, at the same time, such analysis might provide some insights into Alice's personality, likes, dislikes, daily routines etc., to the service. If this service combines its data about Alice with some health forum which is also used by Alice, it can for instance correlate the type of videos she watches and queries she has posted on the forum to deduce her mental health condition. This example is just a glimpse of how a user's privacy can be compromised by online services.

When different service providers collude and combine the user data they have collected, then almost all the online activities of individual users can be monitored and analysed without the knowledge of the users. The user data could be used for unintended purposes such as, pervasive surreptitious tracking, profiling, manipulative behavioral advertising etc. Consider the case of Cambridge Analytica[1]. It abused personal information from Facebook user profiles to build a system that could profile individual voters, in order to target them with personalised political advertisements. This system was used to predict and influence choices of voters during elections in several countries. Such privacy scandals have demonstrated the breadth of user-data misuse and privacy violations that affect not only users but also society in general. Furthermore, the service providers (in general, the corporate actors) that collect and process users' personal information often run the risk of getting hacked[2]. The hacked or stolen user information can be used by the hackers for committing identity fraud, sending spam or phishing messages to users etc. As a consequence of the above security and privacy incidents, we have seen a constantly growing public concern regarding unlimited data collection and processing and loss of privacy in the digital world[3].

Our concept for privacy

As a first step towards solving prevalent privacy issues, we need to choose a way of looking at privacy. Instead of viewing privacy as merely a form of secrecy, we consider privacy as freedom of choice, awareness and control over one's own personal data flows [3, 4]. That is, users must be able to

- clearly see which personal information is being disclosed to which entity, for what purpose and if this disclosure is minimum;
- regulate the disclosure of their information based on the context;

[1] https://www.theguardian.com/news/series/cambridge-analytica-files [last accessed: August 4, 2018]

[2] https://en.wikipedia.org/wiki/List_of_data_breaches [last accessed: July 26, 2018]

[3] http://www.pewinternet.org/2014/11/12/public-privacy-perceptions/; http://databreachlegal.com/2016/05/popular-concern-about-online-privacy-is-still-growing/; https://www.businessinsider.nl/facebook-trust-collapses-after-cambridge-analytica-data-scandal-2018-4/ [last accessed: August 14, 2018]

- be unlinkable based on their transactions.

We approach privacy from the perspective of contextual integrity [2]. Contexts can be broadly defined as the spheres of life such as education, health, workplace, marketplace etc., or as finely drawn as the conventional routines such as visiting a doctor, training for a sport, searching for a job etc. Figure 1.1 shows several contexts in a person's life and how her identity varies across different contexts.

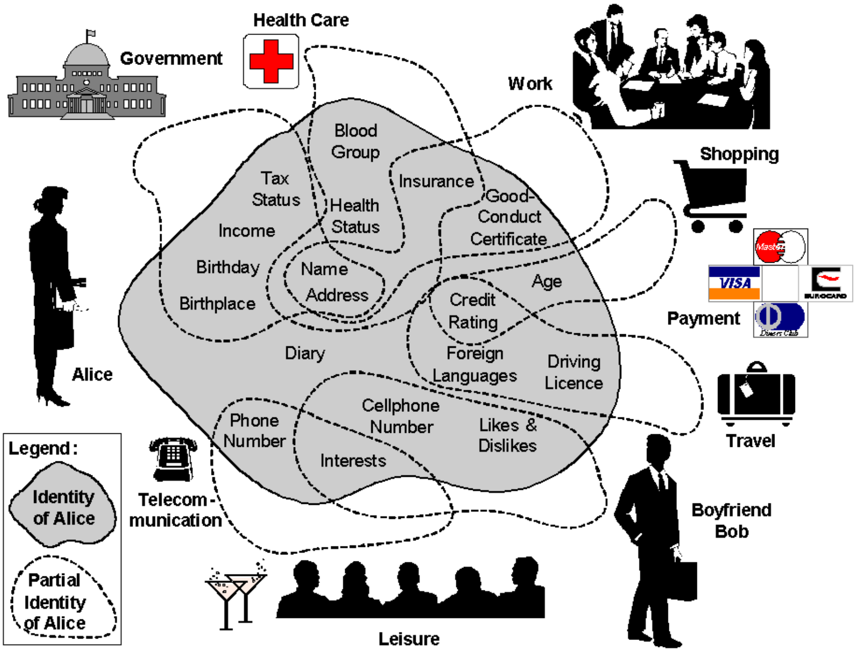


Figure 1.1: An example to show how Alice's identity is made up of pieces of personal information that form partial identities in different contexts. These partial identities are also called contextual identities. (Image Source: FIDIS [5])

According to the theory of contextual integrity, it is crucial to know the context – who is collecting the information, the nature of the information, for which purpose the information is collected, who is analysing it, with whom is it shared, the relationships among the various parties, and even larger institutional and social circumstances [2]. Contextual integrity is said to be upheld only if the collection of personal information is *appropriate* to the context and *norms of the information flow* or *distribution* are obeyed. For example, an online grocery store maintains customer records with some personal information of its customers such as name and address that are needed for delivery of ordered groceries. It may seem appropriate if the store analyses the type of purchased products and shopping patterns of customers solely to provide discounts and to stock up the items in the store. But if the store bombards its customers with questions about their lifestyle choices or their professions, then it definitely breaches the norms of appropriateness. If the store shares the customer information with magazine vendors or advertising agencies without informed consent from the customers, then it not only breaches the norms of appropriateness but also the norms of flow. Then we say, the privacy of the customers is violated by the grocery store.

Diaz et al. [6] classify the landscape of privacy technologies into three privacy paradigms: (i) 'privacy as confidentiality' that focuses on preventing data disclosure altogether and minimising the need to trust others with appropriately handling identifiable and linkable data; (ii) 'privacy as control' paradigm that focuses on ensuring acceptable data collection and usage through user-defined and organisation-defined policies; (iii) 'privacy as practice' paradigm that focuses on making both data disclosure and data usage transparent to users.

Our concept of privacy is a combination of all the above privacy paradigms. That is, we believe that a technology needs to prevent default disclosure of all user data irrespective of the context (privacy by confidentiality). But at the same time preventing data disclosure altogether may not be practical in many real-world use cases. So a technology may have to rely on the policies defined by organisations or service providers for context-dependent data disclosure from users[4] (privacy as control). However, it is crucial for the technology to provide transparency to users about data disclosures so that users themselves can make up their own mind as to whether they are comfortable with which data about them is being collected and how it is being used (privacy as practice).

According to a study on user perceptions about privacy [7], the ability to disclose information selectively, depending on perceived risks and benefits and the degree of trust in the information receiver, is key to users 'feeling in control' in cyberspace. Furthermore, there are many legitimate and beneficial interactions, where anonymity or the ability to adopt multiple personae (contextual identities) is seen as an essential aspect. So, in our view, a user-centric privacy technology needs to embed the above aspects in its design.

Law and technology for privacy

Privacy can be achieved through two complementary aspects: one is law and another is technology. Governments in some countries have recognised the importance of preserving people's privacy and protecting the personal data flowing into the Internet. Thus, there have been efforts in bringing new data protection regulations into effect on a multi-national scale. Here we refer to the new general data protection regulation (GDPR) that has recently come into effect [8]. It aims to protect the right to privacy of data subjects[5] in several ways. For example, the GDPR requires that the data belonging to data subjects is processed lawfully, fairly and in a transparent manner, that the data are only used for the purpose for which they have been collected ('purpose limitation'), that they are only stored for as long as this purpose requires ('storage limitation') and that only the minimal amount of data required for the processing should be collected from users ('data minimisation'). The GDPR also imposes hefty fines and penalties for the entities that illegally collect and pro-

[4] Typically, policies are driven by compliance with existing regulations such as data protection law. So, the law should require a service provider to collect only the minimum data from users.

[5] A data subject is any natural person whose personal data is being collected, held or processed.

cess personal data. Upon non-compliance, the fines can be as high as 20 million euros, or 4% of the worldwide annual revenue of the prior financial year, whichever is higher. Definitely, these are the measures taken in the right direction to regulate the collection and protection of the users' personal data, thereby, ensuring users' privacy. However, only legal policies do not suffice.

For example, ridesharing company Uber may be complying with the law when it stores and analyses personal information of its customers[6] for its core functionality and also to enhance user experience. However, it is difficult or rather impossible for Uber users to meaningfully exert control over their own personal information and preserve their privacy towards Uber. The long and obscure privacy policies that are posed to the users with an 'accept or leave' condition make the task of achieving transparency, user control and contextual integrity only harder. Basically, users are obliged to fully trust Uber to not misuse or share their information with the vendors and third parties who have contractual agreements with Uber. With the enforcement of GDPR, users need to trust the European data protection agencies to detect if Uber violates their privacy as soon as it happens and penalise Uber. We expect that companies like Uber will comply with the GDPR rules out of fear of penalisation. However, it still seems like users are dependent on external entities regarding their privacy and cannot proactively control the disclosure of their personal data. That is why, in addition to strict privacy law, we need technologies that complement such law.

The technologies have to be coded in a way that supports users' privacy. From Lessig's "code is law" viewpoint, computer code regulates; that is, coders (people who write the code) embed laws in the cyberspace [9]. The architecture and the design chosen for technologies decide to protect privacy or promote monitoring, to give users the freedom to choose which information they disclose to whom or be oblivious to this value etc. Moreover, application designers and organisations must bear in mind that, even though privacy may initially not be an important concern for some users, they may react strongly when they see that their privacy has been invaded [10]. Conclusively, the technology architects, designers and coders need to consider privacy as a core societal value and build on the foundational principles of privacy by design [4]. Thus, privacy-enhancing technologies, which proactively embed privacy into their design, have great potential to solve the online privacy crisis.

Privacy-enhancing technologies (PETs) are coherent systems that are designed to protect privacy by eliminating or reducing personal data or by preventing unnecessary and/or undesired processing of personal data, all without losing the functionality of the information system [11]. Using PETs, security and privacy are not necessarily a trade-off, both can be attained simultaneously. In this thesis, we consider one such privacy-enhancing technology – attribute-based credentials.

[6] This information includes customers' name, contact information, payment information, device location, profile photo (if uploaded by the customer), device manufacturer and model, mobile operating system, pick-up location, destination, trip history, contact information for those with whom customers wish to share information, and information about how customers interact with the company's interfaces (e.g., browser types, IP addresses, device identifiers, the areas of Uber's services that a user visits, and the length and frequency of visits).

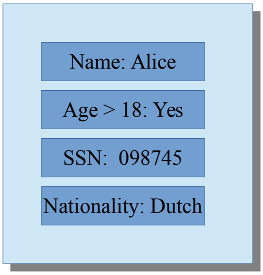


Figure 1.2: An example for an attribute-based identity credential

1.1 Attribute-based Credentials

Attribute-based credentials (ABCs) are an example of PETs which have been mostly considered for privacy-preserving authentication of users. In this thesis, we consider ABCs to achieve contextual identities in practice. In this section, we describe the concept of ABCs and their properties on a high level, then we introduce an operational identity platform that has implemented ABCs on smartphones. In the subsequent chapters, we will look at how can we expand the applicability domain of ABCs, how we can strengthen the security of ABCs on smartphones while ensuring user-friendlness and how to bootstrap ABCs such that they provide trust assurance when applied in real-world use cases.

Authentication of a user can be accomplished based on *what* she is rather than *who* she is. 'What a user is' can be characterised by her personal attributes. A piece of information belonging to a user such as 'age over 18', 'is a student', 'name=Alice' is represented as an *attribute*. Some attributes identify a user (e.g. name, bank account number) and some attributes do not identify the user (e.g. age over 18, gender, nationality) i.e., such attributes hold for many users. An attribute-based credential is a cryptographic container of a set of attributes. Figure 1.2 shows a typical identity credential. ABCs can allow a user to login to a website or access an online resource by disclosing attributes. The nature of the disclosed attributes determines if the user is identified or not identified by the service provider (authenticating party). For example, a user can anonymously log into a chatroom for adults by disclosing her 'age>18' attribute and she is identified to the chat-service-provider (and to other chatroom users) if she discloses her name in addition to 'age>18" attribute during authentication.

Attribute-based credential systems have been developed in the past in a way that they can be trusted, like normal cryptographic certificates, while they protect the privacy of their holder at the same time. A user obtains credentials from a credential issuer, that can vouch for the validity (for this user) of the attributes contained in the credential. Every credential is bound to a user-specific cryptographic key and carries the digital signature of the respective issuer. To authenticate at a service provider (SP), the user discloses few or all attributes from this issued credential and proves their validity. The SP accepts the attributes if they are issued by an issuer

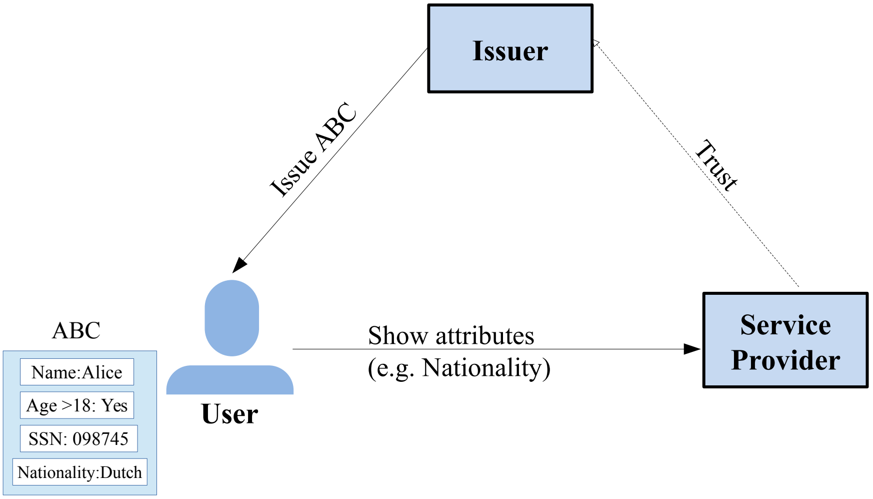


Figure 1.3: An ABC system where User receives ABCs from Issuer and shows selected attributes (at a later time) to Service Provider for the purpose of user authentication.

that it trusts and verifies the user's proof before allowing access to the service. We note that the credential issuer is not involved in the authentication session between the user and the SP. Figure 1.3 depicts an ABC system with its main actors User, Issuer and Service Provider and the interactions between them.

ABCs allow users to achieve both security and privacy simultaneously. The cryptographic nature of the credential-as-container concept includes the following security aspects:

- Authenticity: The credential originates from the issuer, and this issuer asserts that the attributes hold for the user (attribute owner) by digitally signing the credential. The issuer's digital signature ensures authenticity of the attributes contained in the credential.
- Integrity: The issuer's signature on the credential guarantees integrity. That is, it ensures that the attributes contained in the credential have not been altered since they were issued.
- Non-transferability: A credential is non-transferable as it is bound to the secret key of the person involved in the issuance protocol.
- Confidentiality: It is possible for the user to hide all the attributes contained in the credential and prove only their validity to a verifier.

Furthermore, a credential protects the privacy of its owner through the following cryptographic properties.

- Selective disclosure: An ABC allows a user to selectively disclose the attributes stored within a credential to a verifier (service provider) while hiding the other attributes.

- Issuer unlinkability: This property ensures that any information gathered during issuing cannot be used to link a verification of the credential to its issuance.
- Multi-show unlinkability: This property guarantees that when a credential is verified multiple times, these sessions cannot be linked.

When no identifying attributes are disclosed, authentications by a user cannot be linked to her i.e., the unlinkability properties hold even if the credential issuer and all verifiers collude.

Traditional identity documents versus ABCs

An attribute-based credential can be compared to a traditional passport that consists of a person's identity attributes such as name, date of birth, nationality, address and passport number. But as we shall see later, an ABC is much more flexible and privacy-friendly than a passport. Typically a passport is issued to a citizen by a government organization and the passport is checked at various places such as airline companies at the airport, border security control and sometimes by other commercial organizations to verify the name or the age of the passport holder. Similarly, an ABC is signed and issued by a certified *issuer*. After a successful issuance, the user can authenticate with this ABC at any online or offline service provider (also referred to as the *verifier*) that trusts the issuer. ABCs are authentic, non-transferable and non-modifiable (even by the user herself), just as passports.

Let us look at an example scenario to differentiate the utility of a passport-like identity and an attribute-based identity credential. Alice wishes to buy alcohol at a store which requires her to prove that she is over 18 years of age. Here, it would be excessive to present her passport to the storekeeper (acting as the verifier) just to prove her age because the passport contains way more information about Alice than needed in the current context. Especially if it is an electronic passport, then all of Alice's data could be extracted from the chip and stored at the verifier[7]. In such a situation, ABCs allow Alice to show only the 'age>18' attribute to the storekeeper while hiding other attributes within the credential. If Alice buys alcohol from the same store again and shows her 'age>18' attribute, the storekeeper cannot recognise that it was Alice who had bought alcohol from his store in the past. This example illustrates how ABCs protect the user's privacy with the selective disclosure and unlinkability properties. Apart from privacy-protection, ABCs also protect users against identity fraud: as in the above example, if Alice's identifying attributes such as her name and date of birth are not revealed at all, they cannot be abused. There are several other examples where attributes are useful, for instance, only a subscriber attribute might suffice to post a question on an online health forum, only a transaction number attribute might be sufficient in an interaction with a webshop etc.

[7] The verifier might most probably prefer to store the personal information of the users thinking that this information might turn out to be useful in the future. This leads to a so-called *function creep* when data are used for other purposes than the ones they were collected for.

In comparison to traditional identity documents such as passports, driving licenses or even public-key certificates (PKCs)[8], ABCs are privacy-friendly. ABCs allow users to selectively disclose their attributes according to the context and be unlinkable based on their transactions whereas passports and PKCs do not support such privacy features. As we will describe in the next paragraph, the selective disclosure and unlinkability properties of ABCs also make them more flexible than PKCs and passports.

Attribute-based credentials can support non-identifying, partially identifying or fully identifying authentication. Consider the context of an online medical forum where a user wishes to ask some sensitive questions related to her medical condition without identfying herself or linking her transactions on this forum. She can achieve this non-identifying, non-linkable authentication using ABCs at the forum by only disclosing her *forum-membership* attribute. Let us call this attribute a_1. If the context is online shopping, the user can choose to disclose a pseudonymous loyalty number attribute (a_2) to the online store in order to gain points, or alternatively, shop anonymously without disclosing any attribute. If the context is to declare taxes, a user must provide her identifier, e.g., social security number attribute (a_3), to the tax authority so that the authority can use her identifer to find out if she is paying her taxes regularly and also inform her if she will receive some tax-return benefits. Figure 1.4 illustrates this flexibility of ABCs to support three flavors of authentication. This flexibility is a major contrasting factor between ABCs and public-key certificates (PKCs). Authentication using PKCs, as used currently in the digital world, is always identifying and linkable to a user, thus it does not provide privacy.

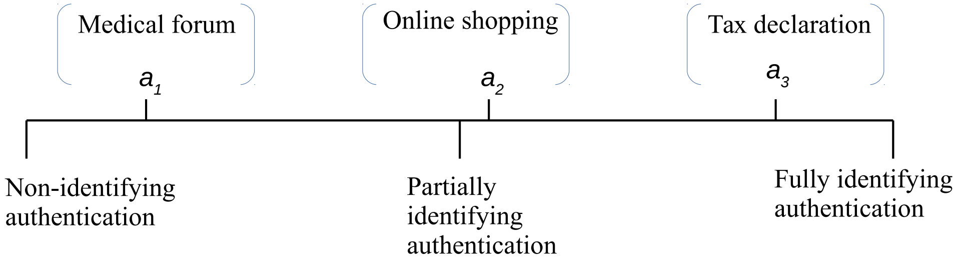

Medical forum
a_1
Online shopping
a_2
Tax declaration
a_3
Non-identifying authentication
Partially identifying authentication
Fully identifying authentication

Figure 1.4: Context-specific authentication with ABCs

Centralised versus decentralised storage of ABCs

Now that we have seen some aspects of the functionality and privacy benefits of ABCs, we get to an important architectural question: which entity stores and performs the computation of such ABCs on behalf of a human user? The two main candidates are: (1) a central server that is not under the direct physical control of the user and (2) user's personal device that is under direct physical control of the

[8] `https://en.wikipedia.org/wiki/Public_key_certificate` [last accessed: August 20, 2018]

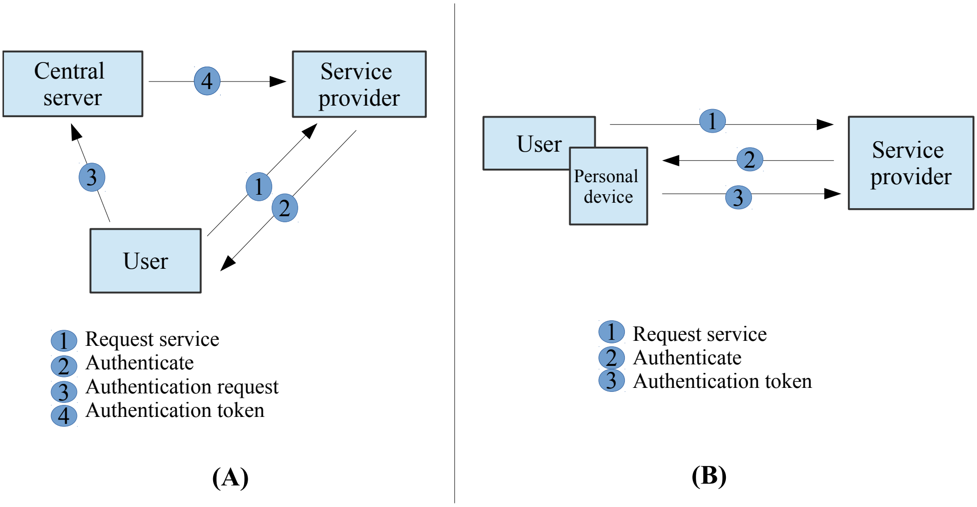


Figure 1.5: (A) Centralised ABC authentication: Central ABC server stores a user's ABCs and creates an authentication token using the ABCs on the user's behalf. (B) Decentralised ABC authentication: A user's personal device (e.g. smart phone) stores her ABCs and creates an authentication token for the service provider on her behalf.

user. The architecture of the ABCs system is *centralised* if the central server is used for storing and computing over ABCs and *decentralised* if the user's personal device is used.

In the centralised architecture, when a service provider asks a user to authenticate, the user forwards the authentication request to the central server that holds her ABCs. Alternatively, the service provider directly contacts the central server to send an authentication token based on which it can authenticate the user. The server creates an authentication token using the user's credentials and sends this token to the service provider. In contrast, the decentralised approach allows the user to directly use the ABCs stored on her personal device to create an authentication token for the service provider. Figure 1.5 illustrates both the approaches.

The centralised approach gives enormous power to the central server because it can monitor and track every transaction involving usage of a user's credentials. Even more, the server can act, in principle, on behalf of users and thus completely take over someone's identity. Both threats are analysed by Brandão et al. in the context of nation-scale brokered-identification systems [12]. They show how a central hub that acts as the broker between users, identity providers and service providers can impersonate users, link users' transactions across different service providers, and also learn personally identifiable information of users. Furthermore, in the centralised approach, the central server becomes a single point of failure. That is, by compromising the server, the attacker may get access to the valuable user-information stored on the server (for example, if the information is stored as plaintext). Moreover, if the attacker takes down the server completely then users will have no way to authenticate to any service provider.

Attribute-based credentials inherently provide privacy benefits to users, however, deploying them in a centralised setting diminishes these benefits, at least towards the central server. Thus, in the interests of both privacy and security, it is a better move to keep ABCs under direct physical control of individuals, so people can "own their own identity". To enable decentralised attribute-based authentication, we can make the users' personal devices such as smart phones, tablets store the user's ABCs (either on hardware or software) and generate cryptographic authentication tokens using these ABCs on behalf of users. The tokens generated by users' devices are directly sent to the authenticating parties i.e. service providers. Another advantage of the decentralised approach is that users can obtain ABCs on their devices from several issuers and use them as many times as they want at various service providers without consulting issuers. Basically, their devices act as ABC wallets from which users can selectively disclose attributes based on the authentication context. The decentralised setup for the use of ABCs is shown in Figure 1.5 (B).

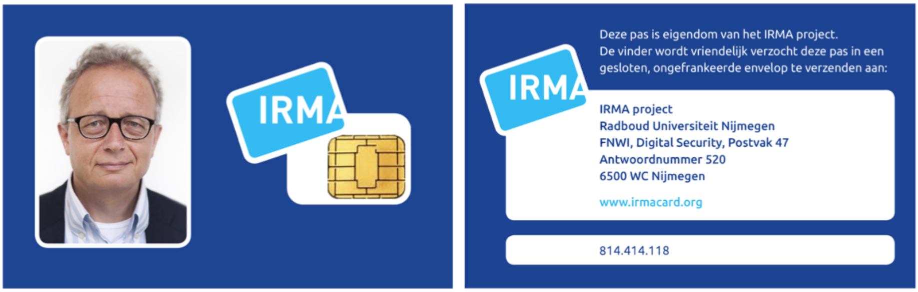
I Reveal My Attributes (IRMA)

Cryptographic techniques that enable secure and privacy-friendly attribute-based authentication have been around for more than a decade, see [13, 14, 15]. Although provably secure and privacy enhancing in a cryptographic sense, ABCs have not been adopted in mainstream applications and their actual uptake in practice remains low. This may be due to the fact that little effort has been spent

- to make the ABC system (that consists of complex concepts and the cryptographic mechanisms, multitude of protocols and features) simple to understand and use even by non-specialists and,
- to create ABC systems that better align with modern users' workflows that often involve gadgets like mobile devices (such as smart phones and tablets).

Recognising the true potential of ABCs in the real world applications, the IRMA (I Reveal My Attributes) project started in 2012 within the Digital Security group in Nijmegen, the Netherlands, as a research effort into the practical realisability of attribute-based authentication.

Initially, three attribute-based credential systems were scrutinised: U-prove [16], self-blindable attributes [17], and Idemix (Identity Mixer) [18], to assess the weaknesses, strengths and opportunities of these three approaches. Idemix came out as the most flexible and provided advanced privacy properties such as multi-show unlinkability. As smartcards are secure and also widely used for many digital applications (e.g. identity management, payments, loyalty points collection), they became the first choice of platform for the IRMA implementation of ABCs. Consequently, the IRMA team developed a low level prototype implementation of Idemix on a MULTOS smart card [19]. This implementation followed a decentralised architecture in which the user's device that stores ABCs is a smartcard. Figure 1.6 shows how a user's IRMA card looked like.



Deze pas is eigendom van het IRMA project.
De vinder wordt vriendelijk verzocht deze pas in een gesloten, ongefrankeerde envelop te verzenden aan:

IRMA project
Radboud Universiteit Nijmegen
FNWI, Digital Security, Postvak 47
Antwoordnummer 520
6500 WC Nijmegen

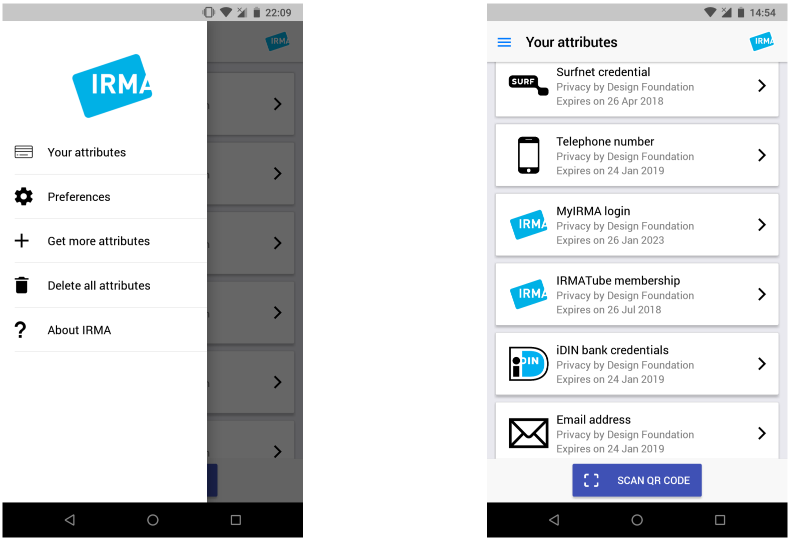
www.irmacard.org

814.414.118

Figure 1.6: A typical IRMA card from 2014

The IRMA implementation of ABCs on smartcards had two benefits. First, it achieved better performances on smartcards for issuance and selective disclosure of attributes when compared to other smartcard implementations at that time [19]. Second, it was much simpler than Idemix itself. In contrast to the Idemix cryptographic library, the IRMA implementation only focused on a subset of attribute-based functionality: credential issuance and selective disclosure of attributes from credentials. This simplicity does not restrict the system much in practice [20]. For instance, IRMA does not support range proofs as the original Idemix, but these can often be simulated using simple attributes. When a user has to prove that she is over 18 years of age, Idemix creates a range proof based on the user's date of birth that proves that the user's age lies between two values. However in IRMA, the 'age>18','age>65' etc., are encoded as attributes whose values are either *yes* or *no*. To prove that a user is over 18 years, the user just discloses 'age>18' attribute and creates a selective disclosure proof. Restricting IRMA to only credentials and attributes makes the system simpler as a whole and more efficient performance-wise, and it has the additional benefit that users always have to make the same decision: Do I want to disclose this attribute or not?

Although IRMA's efficient smartcard-implementation was a major step in realising ABCs in practice, the uptake of ABCs remained low among users and service-providers. The IRMA team considered the fact that the uptake depends on more than just the perceived security of the underlying platform. Other factors such as, portability and usability play an important role too. In comparison with smart cards, smart phones have more advanced storage and computation capabilities, they support a wider range of cryptographic applications, and most importantly, they provide interactive user interfaces and connectivity for users. In an obvious sense, smart phones are more convenient as they are better aligned with modern users' workflows, they increase the user's control over the outgoing personal attributes and expand the range of applications for which ABCs could be used. So, to turn the tide in the adoption of ABCs, IRMA shifted its focus from ABCs on smartcards to ABCs on smartphones. Consequently, the IRMA team began the research and implementation of a practical, decentralised, and user-centric attribute-based identity platform on smartphones.



Surfnet credential
Privacy by Design Foundation
Expires on 26 Apr 2018

Telephone number
Privacy by Design Foundation
Expires on 24 Jan 2019

MyIRMA login
Privacy by Design Foundation
Expires on 26 Jan 2023

IRMATube membership
Privacy by Design Foundation
Expires on 26 Jul 2018

iDIN bank credentials
Privacy by Design Foundation
Expires on 24 Jan 2019

Email address
Privacy by Design Foundation
Expires on 24 Jan 2019

Figure 1.7: Screenshots of IRMA app that shows the main and the user attributes screens.

In 2016, the IRMA project spurred an independent, non-profit spin-off called *Privacy by Design (PbD) Foundation*[9]. The aim of this foundation is to create and maintain free open-source software that primarily focuses on the privacy of the user and its main project is the IRMA system at present. Furthermore, it is also involved in establishing formal connections with other societal organisations to further the development and roll-out of the IRMA technology. On the technical development front, the PbD foundation has developed an open-source smartphone application called the *IRMA app*. This app extends IRMA's smartcard implementation of Idemix ABCs [14, 15] with an interactive user interface. The IRMA app acts as an ABC-wallet in which users can collect ABCs from several issuers and use them to authenticate themselves based on the context. The IRMA app is available both on the Android and iOS platforms and some screenshots of the app are provided in Figure 1.7. In sum, IRMA offers a decentralised, privacy-friendly, user-centric platform for proportional and contextual use of personal identity attributes. By this, IRMA realises many objectives described in [21, 8] within the European legal framework. The context-dependence concept used in IRMA is related to Helen Nissenbaum's interpretation of privacy as contextual integrity [2]. Furthermore, the concept of privacy by design and by default which is important in the GDPR is embedded in IRMA [4].

The ultimate goal of the IRMA project is to demonstrate the applicability of attribute-based credentials and make their use practical and trustworthy. IRMA aspires to reach this goal both by academic research and implementation efforts. The work in this thesis is a part of this academic research of the IRMA project and focuses on the following goals:

[9] `https://privacybydesign.foundation/en/`

- to explore areas in the digital realm where the application of the ABC technology can have benefits to all the stakeholders (credential issuers, service providers and end users).
- to investigate the trust and security challenges that ABCs face due to being implemented on smart phones and to find technical solutions to counter them.

With these goals we want to achieve the final objective of putting privacy-friendly contextual identities via ABCs in practice. Some of the proposals made in the thesis have been integrated into the IRMA app by the PbD foundation. These efforts within the IRMA project is bringing research and practice as close as possible in the field of attribute-based identity management. Several research projects including IRMA, ABC4Trust[10] have been working towards the advancement of attribute-based technologies. It is only through such dedicated and collective efforts that we can further the chances of these PETs to flourish and get widely adopted in the real world.

1.2 Research Questions and Contributions

The main research question of the thesis is:

How can ABCs be better utilised in practice?

This central question is divided into the following sub-questions that are addressed by the chapters of the present thesis. The first two questions focus on the real-world applications where ABCs can be used other than their traditional application: authentication. The last two questions focus on preparing ABCs for deployment.

RQ1. How can we extend attribute-based authentication to digital signatures in practice?

RQ2. How can we use ABCs in transactions such as webshopping?

RQ3. What technical approaches can increase the security of ABC-keys stored on users' personal devices (e.g. smart phones) without relying on the secure hardware?

RQ4. How can we increase the trustworthiness of ABCs on users' smartphones so that service providers (verifiers) can trust the ABCs to reflect the real identities of the users?

The summary of main contributions of the thesis are listed below. The structure of the thesis and individual contributions of the author in every chapter are described in the subsequent section.

Contribution C1. ABCs are not just versatile in terms of supporting different forms of authentication (fully, partially or non-identifying authentication). Their utility can be extended to support other important user actions such as *digital signatures*. The thesis presents how to realise attribute-based signa-

[10] https://abc4trust.eu/

tures (ABSs) in practice along with attribute-based authentication using the same set of attributes in a secure way. Under our proposal, users can easily create timestamped ABSs on messages using a selected set of their personal attributes. We list many use-case scenarios that can benefit from the flexibility provided by the ABSs. We also expect ABSs to be *the* application that can spur wide use of attribute-based credentials in the real world.

Contribution C2. We explore how attribute-based credentials can be used in *multi-step transactions* (e.g. a shopping transaction consisting of cart creation, payment and delivery steps) to enforce data minimisation and purpose limitation. We devise privacy-preserving protocols for carrying out attribute-based web-shopping transactions. We apply ABSs without timestamps in our protocols. Such applications of ABSs show that they can be used for more purposes than conventional digital signing of documents (e.g. contracts, agreements).

Contribution C3. Using ABCs on phones is user friendly and efficient but there are also concerns about the key security when stored on phones. The security of a user's secret key that is bound to her ABCs is crucial for secure and trustworthy operation of the entire ABC system. In this chapter, we first consider a basic key-sharing scheme between a user's phone and a central server to secure the user's secret key. This enforces instantaneous blocking of the key when the phone is lost or stolen. However, if the key-sharing server cannot be fully trusted, it may collude with service providers and compromise users' privacy provided by ABCs via timing attacks. To counter this, we propose TANDEM, a set of protocols that augment threshold cryptography to ensure the security of the users' secret keys while maintaining users' privacy towards a malicious server.

Contribution C4. It is important to bootstrap a technology in a secure, trustworthy and user-friendly manner. We aim to address this issue for ABCs at user enrolment, which is a first step in deploying any new technology. We develop secure self-enrolment protocols through which users can enrol themselves online and obtain trustworthy ABCs onto their phones. We expect that this way of bootstrapping ABCs will increase the trust assurance provided by ABCs and bring the attribute-based technology closer to being adopted by the service providers and also by the users.

The Structure of the Thesis

Figure 1.8 depicts the structure of the thesis. **Chapter 1** gives the introduction to the thesis. **Chapter 2** describes the preliminaries consisting mainly of cryptographic building blocks used in attribute-based credentials and, Idemix protocols for credential issuance and disclosure of attributes. The core of the thesis is divided into two parts. **Part 1** describes ABC applications. It consists of **Chapters 3 and 4** that covers the first two contributions of the thesis. The reason for describing the applications of ABCs in the first part of the thesis is to show that ABCs are useful not only for authenticating users i.e. as better alternatives for username-passwords or public-key certificates, but also for other purposes such as digital signing and

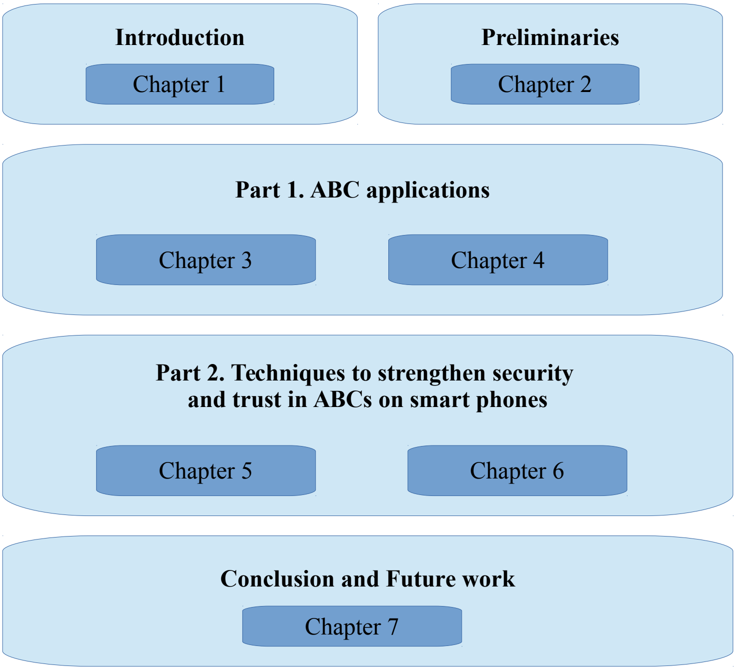


Figure 1.8: The structure of the thesis

carrying out multi-step transactions (e.g. online shopping). As ABCs have such important real-world applications, they need to provide strong security and trust assurances to verifiers. **Part 2** discusses the techniques to enhance security and trust in the ABC applications without reduction in the privacy and some bootstrapping methods which take ABCs off the ground and bring them closer to getting deployed in real-world use cases. This part consists of **Chapters 4 and 5** and covers the last two contributions mentioned in the previous section. **Chapter 6** presents conclusions and recommendations for future work.

Below we briefly describe the core chapters and the author's personal contributions.

Chapter 3 describes how ABCs can be manifested as attribute-based signatures (ABSs), compares these with public-key signatures, discusses the practical issues that arise due to the introduction of the signature functionality to an existing attribute-based authentication scheme, and finally, it proposes possible cryptographic and infrastructural solutions.

> This chapter is based on the paper "Towards Practical Attribute-based Signatures" [22] by myself, Gergely Alpár, Fabian van den Broek, and Bart Jacobs.

Chapter 3 addresses the research question RQ1 as summarised in Contribution C1. In particular, my contribution in the chapter is analysing the value propositions of ABSs by comparing them with the existing digital signatures, designing a practical ABS scheme with trusted timestamps and assessing its security, presenting a practical instantiation of timestamped ABSs in IRMA and a discussion on revoking ABSs and use cases that can benefit from ABSs, and writing the paper. The ideas proposed in the chapter have resulted from many brainstorming sessions and helpful suggestions by the coauthors. The IRMA signatures described in this chapter have been successfully implemented by the Privacy by Design Foundation so that IRMA users can create attribute-based signatures in a secure and an easy way.

Chapter 4 describes how real-world transactions such as online shopping can be turned into data minimised transactions using attributes. The attribute-based shopping scheme proposed in this chapter benefits all the parties involved in the transaction.

> This chapter is an updated version of the paper "Privacy-Preserving Webshopping with Attributes" [23] by myself and Gergely Alpár.

Chapter 4 addresses the research question RQ2 as summarised in Contribution C2. My contribution includes analysing how to model the stages in an online shopping transaction using ABCs, designing the protocols for each stage, assessing the security and privacy of the scheme, performing a comparison of attribute-based webshopping with existing webshopping models and writing the paper. Discussions with Gergely greatly helped to improve the protocols and the overall quality of the paper.

Chapter 5 introduces Tandem – a set of protocols that augments threshold cryptography to secure cryptographic keys (e.g. secret keys that are bound to users' ABCs) stored on users' (insecure) devices while maintaining users' privacy.

> This chapter is an updated version of the paper "Tandem: Securing Keys by Using a Central Server While Preserving Privacy" [24] by myself, Wouter Lueks, Gergely Alpár and Carmela Troncoso. The paper is under submission at the time of publishing this thesis.

Chapter 5 addresses the research question RQ3 as summarised in Contribution C3. The initial idea to use threshold cryptography to secure user keys stored on smart phones (on the IRMA app), using a central server was conceived after some brainstorming sessions with some of the IRMA team members (including the first three authors of the above article). Tandem was later developed by the authors to counter the privacy issues that arise by using traditional threshold cryptography when the central server is not trusted to protect users' privacy. My contribution includes analysing the application of a basic key-sharing scheme in the IRMA context, motivating the need for Tandem: a key-sharing solution that ensures users' privacy under stricter (aka. weaker) trust assumptions w.r.t. the key-sharing server, establishing an overview of the state-of-the-art solutions in threshold cryptography and how these solutions differ from Tandem, applying Tandem to ABCs (especially to the Idemix credentials). The development of Tandem protocols and writing of the paper were a joint work with the co-authors.

Chapter 6 presents several ways of bootstrapping ABCs i.e., mechanisms for users to obtain ABCs remotely but securely. With these mechanisms, the service providers, who are the potential verifiers of the users' ABCs, can have reasonable trust assurance in the presented credentials. These methods fall within our self-enrolment framework and they make ABCs more practical and enable quick adoption by service providers.

> This chapter is an updated version of the paper "Securely Derived Identity Credentials on Smart Phones via Self-enrolment" [25] by myself, Fabian van den Broek and Bart Jacobs.

Chapter 6 addresses the research question RQ4 as summarised in Contribution C4. The concept of secure self-enrolment for ABCs was jointly developed by the authors. My contribution was performing a preliminary assessment of IRMA ABCs to assign assurance levels based on the eIDAS regulation requirements, and writing the full paper. Some of the SSE methods described in the chapter are implemented by the Privacy by Design Foundation to enable secure and trustworthy issuance of IRMA credentials to users.

Chapter 2

Preliminaries

IRMA implements a subset of features supported by IBM's Idemix technology [14, 15, 14]: credential issuance and selective disclosure of attributes from credentials. It uses the cryptographic building blocks such as the zero-knowledeg proofs, Camenisch-Lysyanskaya signatures and Idemix protocols for issuing and showing attributes from the ABCs. As some of the contributions in the thesis involve modifying the cryptographic details of IRMA, we briefly describe here the primitives and protocols used in it.

2.1 Zero-knowledge Proofs

A frequently used cryptographic concept in attribute-based credentials is a *proof of knowledge*. The goal of such a proof is for a user, or *prover*, to convince a *verifier* of a given statement. For example, in a challenge-response construction, a user proves that she knows the private key corresponding to her public key by signing or decrypting a challenge sent by the verifier.

To describe such proofs of knowledge we use the notation introduced by Camenisch and Stadler [26]. For example,

$$PK\{(\alpha) : h = g^{\alpha} \mod p\}$$

denotes a proof of knowledge of a value α such that $h = g^{\alpha} \mod p$, that is, a proof of knowledge of the exponent (α) in a discrete logarithm problem.

A *zero-knowledge protocol* is a way to convince the verifier that the user has the knowledge of some secret without giving any further information to the verifier than what he already knows. More precisely, the term *zero-knowledge* refers to the fact that whatever information the verifier learns from the user, that information could have been generated by the verifier on its own, without the assistance of the user. However, a verifier who actually carried out the protocol will be convinced that the user has the specified knowledge (e.g. the private key).

A well-known example of a zero-knowledge protocol is Schnorr's protocol [27], Common information: $(p, q, g), h = g^x$

User knows x		Verifier
$t \in_R \mathbb{Z}_q$		
$u = g^t \mod p$	$\xrightarrow{\quad u \quad}$	
	$\xleftarrow{\quad c \quad}$	$c \in_R \mathbb{Z}_q$
$r = t + cx \mod q$	$\xrightarrow{\quad r \quad}$	check $u \overset{?}{=} g^r h^{-c} \mod p$

Figure 2.1: Schnorr's protocol [27] in which User proves knowledge of x such that $h = g^x$ to the Verifier.

which proves knowledge of a discrete logarithm. We describe Schnorr's protocol which works in a cyclic group $\mathbb{G}$ whose description is (p, q, g) is known to the public. Here p, q are primes where q divides $(p - 1)$, q is the order and g is the generator of the group $\mathbb{G}$. The user's private key is the discrete logarithm x and her public key is $h = g^x \mod p$. To prove that the user knows the private key x, she first commits to a random value t and sends the commitment $u = g^t \mod p$ to the verifier. The verifier then generates a challenge c at random and sends this to the user, who computes the response r based on the challenge. Finally, the verifier checks whether $u = g^r h^{-c} \mod p$. This protocol is depicted in Figure 2.1. The triple (u, c, r) from the Schnorr's protocol constitutes a transcript that proves the user's knowledge of the private key x.

An honest-verifier zero-knowledge proof has to satisfy three main properties:

- Completeness: A prover who knows x can convince the verifier. That is, the verification equation $u \overset{?}{=} g^r h^{-c} \mod p$ holds for such a prover.
- Soundness: If the prover does not know x, then she cannot convince the verifier. This property guarantees the verifier that the user actually knows the secret.
- Zero-knowledge: The verifier does not learn any other information except the fact that the prover knows x, since, he could have computed such a triple (u, c, r) himself by choosing c and r at random and computing $u = g^r \cdot h^{-c} \mod p$.

The Schnorr's zero-knowledge protocol shown in Figure 2.1 works for groups with known order. Here, modular reduction $\pmod q$ is applied by the user during the computation of the response in order to ensure that the response is distributed uniformly and it hides the private key x. However, a similar protocol can be constructed for groups in which the order of the (sub)group is not known to all parties. This is for instance the case in an RSA setting where the order of the group is only known by the party that knows the primes p and q. As a result the user cannot perform the modular reduction using the order of the group when computing the response. This means that the response no longer hides the secret x as it is not distributed uniformly. Therefore, the user must choose a significantly larger[1] random

[1] For instance, the length of the random value t should be 128-bits longer than the combined lengths of the private key x and the challenge c. This is in contrast to the zero-knowledge protocol

value t such that u is distributed statistically close to uniform over the subgroup generated by the generator g and the secret x is statistically hidden in the response r. Hence, this protocol is called *statistical zero-knowledge* [28].

Fiat-Shamir Heuristic. The Schnorr's protocol depicted in Figure 2.1 is interactive in nature. That is, there is active communication between the user and the verifier during the protocol in which the response is computed over the challenge sent by the verifier. However, in practice, the zero-knowledge protocols that are often implemented are non-interactive in nature. The Fiat-Shamir heuristic [29] can be used to transform a zero-knowledge protocol into a non-interactive zero-knowledge proof. This is often used to translate a zero-knowledge protocol into a signature scheme, or to reduce the communication overhead of the interactive protocols.

To make a zero-knowledge protocol non-interactive, the challenge c is not retrieved from the verifier but computed as

$$c \leftarrow \textsc{Hash}(u)$$

where Hash is a cryptographic hash function and u is the commitment that the user computed in the previous step. This non-interactive proof of knowledge can be used as a signature when the challenge is computed as

$$c \leftarrow \textsc{Hash}(msg, u)$$

where msg is some message to be signed. Both the commitment u and the response r are calculated as in the interactive Schnorr's proof. The transcript (u, c, r) in this case is a signature on msg. This can be verified by checking whether the following holds:

$$c = \textsc{Hash}(msg, \hat{u})$$

where $\hat{u} = g^r \cdot h^{-c}$. If the proof (c, r) is valid, this holds since:

$$\hat{u} = g^r \cdot h^{-c} = g^{t+c\cdot x} \cdot h^{-c} = g^{t+c\cdot x} \cdot (g^x)^{-c} = g^t = u \mod p.$$

Such a non-interactive zero-knowledge proof is often called a signature proof of knowledge, because of the message that is included in the proof.

2.2 Camenisch–Lysyankaya Signature

We recall that an attribute-based credential is a container of attributes which carries the credential issuer's signature. The signature scheme used to construct an ABC in Idemix/IRMA is the Camenisch–Lysyanskaya (CL) signature scheme [18, 30]. Our description of the CL signature scheme and the notation are based on the specification of the Identity Mixer cryptographic library [14].

for known-order groups in which the length of t is just the length of the prime order q.

Keys used in CL signature scheme. The Camenisch–Lysyanskaya signature works in the quadratic residue subgroup QR_n of $\mathbb{Z}_n^*$ in which the strong RSA assumption holds. The public key of the issuer consists of the RSA modulus n and some random generators from the quadratic residue group: $S, Z, \{R_i\}_{i \in M}$ where M is the maximum number of attributes that this public key can support. The modulus n is a product of two safe primes i.e., primes p and q such that $p' = {}^{(p-1)}/_{2}$ and $q' = {}^{(q-1)}/_{2}$ are also primes. The private key of the signer is p, q. The order of the QR_n group is given by $|QR_n| = p'q'$ and it is known only to the signer. The signer is the issuer in an ABC setting.

CL Signature Generation. To sign a collection of attributes $\{a_i\}_{i \in M}$, these a_i first have to be aggregated into a single group element Q according to the following equation:

$$Q = \frac{Z}{S^v \cdot \prod_{i \in M} R_i^{a_i}} \mod n, \tag{2.1}$$

where v is a random number. This value v is used in Section 2.3 for blinding the attributes that have to remain hidden (e.g. secret key of the user), and in Section 2.4 to randomise the signature.

The actual signature generation process is similar to the RSA signature scheme. The first step is the generation of a random prime e which is used as the ephemeral RSA public key for this signature. Next, the RSA private key $d = e^{-1} \mod (p'q')$ corresponding to the public key e is computed. Finally, an RSA signature is put over the aggregated messages as follows:

$$A = Q^d \mod n \tag{2.2}$$

As a result the Camenisch-Lysyanskaya signature over the set of attributes $\{a_i\}_{i \in M}$ is the triple (A, e, v).

CL signature verification. In order to verify such a Camenisch-Lysyanskaya signature (A, e, v) over the set of attributes $\{a_i\}_{i \in M}$, the verifier has to check the following equation:

$$A^e = \frac{Z}{S^v \cdot \prod_{i \in M} R_i^{m_i}} \mod n \tag{2.3}$$

It is similar to verifying an RSA signature. The above verification equation can be rearranged as follows so that it is not necessary to compute the inverse:

$$Z = A^e \cdot S^v \cdot \textstyle\prod_{i \in M} R_i^{m_i} \mod n \tag{2.4}$$

The unforgeability of CL-signature scheme relies on the *Strong-RSA assumption* which is stated as follows: Given an RSA modulus n and an element $u \in \mathbb{Z}_n^*$, it is hard to compute values A and $e > 1$ such that $A^e = u \mod n$.

The CL signature scheme is an ideal building block for privacy-preserving technologies such as ABCs as it has the following properties.

1. A signer can issue signatures on committed values without knowing the signed value. Basically, these signatures are blind signatures [31]. This is useful in an ABC setting because every credential has the user's secret key as an attribute that should be hidden from the issuer (signer) when he signs the credential during issuance.
2. A signature owner (or a user who owns attributes) can prove the knowledge of a signature on a committed value. In an ABC setting, the attribute owner can prove the validity of the issuer's signature on the credential from which attributes are disclosed to the verifier.
3. A signature owner or any party (not necessarily the signer) can modify a CL-signature without changing the message that it signs. The resulting randomised (modified) signature can be verified against the original public key of the signer. In an ABC setting, randomisability of CL signature allows the user to be unlinkable based on the issuer's CL signature during successive attribute disclosures to verifiers.

2.3 Idemix Credential Issuance

As described in Section 1.1, an attribute-based credential contains a set of attributes and all the attributes are bound to a user's secret key. An example ABC with four attributes is shown in Figure 2.2 in which sk denotes the user's secret key sk, $a_1, .., a_4$ denote the attributes that hold for the user and (A, e, v) denotes the credential issuer's signature. This signature is computed as follows:

$$A = \Big(\frac{Z}{S^v R_0^{sk} R_1^{a_1} R_2^{a_2} R_3^{a_3} R_4^{a_4}}\Big)^{1/e} \pmod n$$

In the above signature, $(n, S, Z, R_0, R_1, R_2, R_3, R_4)$ is the public key and the factors of modulus n: p and q form the private key of the issuer, respectively. We have combined the equations (2.1) and (2.2) in the above example credential. We emphasise that the user's secret key sk (exponent of R_0 in the above equation) is bundled along with the attributes $a_1, .., a_4$ into a credential that carries the issuer's signature. The issuer creates the credential and puts a CL signature on it during issuance. We explain how the credential issuance takes place in Idemix. As the issuance protocol from the Idemix's cryptographic library is implemented in IRMA, the same protocol description holds for the issuance of credentials in the IRMA system. We provide the algorithms for every stage in this protocol because they are required to understand the extensions and modifications that we make to these algorithms to support our proposals in the chapters 3, 4 and 5.

A credential issuance is an interactive protocol between a user and an issuer. During the issuance protocol, the issuer creates a new credential for the user that is bound to the user's secret key sk and blindly signed with its private key p, q. This protocol consists of three main steps.

- *Commitment phase* where the user generates a commitment U to her secret key sk as $U \leftarrow S^{v'} R_0^{sk} \mod n$ where v' is a randomly chosen blinding value.

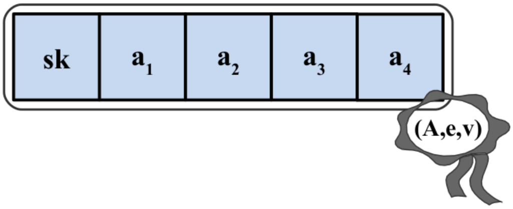


Figure 2.2: A visual representation of an IRMA credential

Algorithm 1 Prepare for a blind Camenisch-Lysyanskaya signature by creating the commitment U to the user's secret key sk [32].

```
function CL-BLIND-COMMIT($sk, (n, S, Z, \{R_i\}_{i \in M})$)
    $v' \leftarrow$ RANDOM( )
    $U \leftarrow S^{v'} \cdot R_0^{sk} \mod n$
return $(U, v')$
```

Algorithm 1 shows how the commitment U is created. Then the user proves to the issuer the knowledge of sk and correctness of her commitment U with the proof:

$$PK\{(\nu, \mu) : U = S^{\nu} \cdot R_0^{\mu} \mod n\}$$

Algorithm 2 shows how we can construct the above proof. This proof is similar to a non-interactive Schnorr proof of knowledge. The freshness of this proof is guaranteed by a nonce n_U provided by the issuer (aka the signer).

The issuer verifies the proof above as per Algorithm 3. The verification succeeds if the signer can successfully reconstruct the commitment $\tilde{U}$ on U. This reconstruction works because of the following equation. We use the notation $\equiv_n$ to denote $\mod n$ at some places in the thesis for conciseness, for instance in the equation below.

$$\begin{aligned} \hat{U} &\equiv_n S^{\hat{v}'} \cdot U^{-c} \cdot R_0^{\hat{sk}} \equiv_n S^{\tilde{v}'+c \cdot v'} \cdot U^{-c} \cdot R_0^{\tilde{sk}+c \cdot sk} \\ &\equiv_n S^{\tilde{v}'} \cdot U^{-c} \cdot U^{c} \cdot R_0^{\tilde{sk}} \equiv_n S^{\tilde{v}'} \cdot R_0^{\tilde{sk}} \\ &\equiv_n \tilde{U} \end{aligned}$$

Upon successful verification of the proof, the issuer proceeds to sign.

- *Issuer's signature phase*: This phase consists of the issuer blindly signing the user's commitment U along with other attributes $\{a_i\}_{i \in M}$ where M is the maximum number of attributes in the credential. This step consists of generating a blind Camenisch-Lysyanskaya signature as shown in the Algorithm 4. The random prime e acts as an ephemeral public key for this signature and the corresponding private key is d which is calculated as $d \leftarrow e^{-1} \mod (p'q')$.

  Furthermore, the issuer who is the signer provides the following proof of knowledge to prove to the user that it knows the private key d and the CL signature

Algorithm 2 Generate a proof of correctness for the user's commitment U [32].

```
function CL-PROVE-U($(U, v'), n_U, sk, (n, S, Z, \{R_i\}_{i \in M})$)
    $\tilde{v}' \leftarrow$ RANDOM( )
    $\tilde{sk} \leftarrow$ RANDOM( )
    $\tilde{U} \leftarrow S^{\tilde{v}'} \cdot R_0^{\tilde{sk}} \mod n$
    $c \leftarrow$ HASH($U, \tilde{U}, n_U$)
    $\hat{v}' \leftarrow \tilde{v}' + c \cdot v'$
    $\hat{sk} \leftarrow \tilde{sk} + c \cdot sk$
return $(c, \hat{v}', \hat{sk})$
```

Algorithm 3 Verify the proof of correctness for U [32].

```
function CL-VERIFY-U($U, (c, \hat{v}', \hat{sk}), n_U, (n, S, Z, \{R_i\}_{i \in M})$)
    $\hat{U} \leftarrow U^{-c} \cdot S^{\hat{v}'} \cdot R_0^{\hat{sk}} \mod n$
    if $c \neq$ HASH($U, \hat{U}, n_U$) then return INVALID
    return VALID
```

has been constructed correctly.

$$PK\{(\delta) : A = \left(\frac{Z}{U \cdot S^{v''} \cdot \prod_{i \in M} R_i^{a_i}} \right)^{\delta} \mod n\}$$

The user can validate the issuer's CL-signature based on the above proof. If the proof verifies correctly, then the user proceeds to complete the credential in the next step. The issuer's task of an issuance ends with the creation of this proof and it can actually throw away the ephemeral values e and d as they were generated only for this CL signature.

- *Credential completion phase*: Finally, the user completes the credentials i.e.

Algorithm 4 Generate a blind Camenisch-Lysyanskaya signature [32].

```
function CL-BLIND-SIGN($U, \{a_i\}_{i \in M}, (n, S, Z, \{R_i\}_{i \in M}), (p, q)$)
    $v'' \leftarrow$ RANDOM( )
    $U \leftarrow U \cdot S^{v''} \mod n$
    for all $i \in M$ do
        $U \leftarrow U \cdot R_i^{a_i} \mod n$
    $Q \leftarrow Z \cdot U^{-1} \mod n$
    $e \leftarrow$ RANDOMPRIME( )
    $d \leftarrow e^{-1} \mod (p'q')$
    $A \leftarrow Q^d \mod n$
    return $(A, e, v'')$
```

the issuer's signature by combining the blinding values of the user and signer, v' and v'' respectively as

$$v \leftarrow v' + v''$$

to create the final randomisation value v of the Camenisch-Lysyanskaya signature (A, e, v). As we will see in the next section, the user proves the knowledge of this signature but she never shows the credential itself to a verifier during a credential verification .

2.4 Idemix Credential Verification via Selective Disclosure

A user can authenticate to a verifier using her ABCs that were issued as described in the previous section. This is cryptographically realised by a selective disclosure ($\mathcal{SD}$) protocol. In an $\mathcal{SD}$ protocol, the user discloses a subset of attributes from her ABCs and, then proves – using a zero-knowledge proof – to the verifier the validity of the attributes. D denotes the set of disclosed attributes and H denotes the set of hidden attributes which includes the secret key attribute. Therefore, H and D are disjoint sets of attributes: $H \cap D = \emptyset$ and the number of elements in the union set $H \cup D$ equals the total number of attributes inside the credential.

Technically, the $\mathcal{SD}$ protocol in the Idemix and in IRMA consists of two steps: (i) randomisation of the issuer's signature on the ABC from which the attributes will be disclosed to a verifier and, (ii) creating a zero-knowledge proof to prove the knowledge of the issuer's signature and the hidden attributes in the ABC. These two steps are described in detail below.

- *Randomise issuer's signature.* When a user reveals non-identifying attributes (e.g. age>18) from an ABC to a verifier during an authentication, the only possibility for the verifier to link subsequent authentication of the same user is by linking her authentications based on the issuer's signature on the ABC. That is why the CL signature of the issuer on the ABCs needs to be randomised to prevent this linkability. The CL signature scheme allows the user to randomise the signature without modifying the attributes inside the credential. The randomisation is done using Algorithm 5. First, a randomisation value r is generated to randomise the signature component A. Next the value v is adjusted such that the signature remains valid, that is, it still satisfies (2.3):

$$\begin{aligned} A'^e &\equiv_n (A \cdot S^r)^e \equiv_n A^e \cdot S^{e \cdot r} \\ &\equiv_n \frac{S^{e \cdot r} \cdot Z}{S^v \cdot \prod_{i \in M} R_i^{a_i}} \equiv_n \frac{S^{-e \cdot r} \cdot S^{e \cdot r} \cdot Z}{S^{-e \cdot r} \cdot S^v \cdot \prod_{i \in M} R_i^{a_i}} \\ &\equiv_n \frac{Z}{S^{v - e \cdot r} \cdot \prod_{i \in M} R_i^{a_i}} \equiv_n \frac{Z}{S^{v'} \cdot \prod_{i \in M} R_i^{a_i}} \end{aligned}$$

  Basically, the issuer's signature (A, e, v) becomes (A', e, v') after randomisation where $A' = A \cdot S^r \mod n$ and $v' = v - e \cdot r$.

- *Create a zero-knowledge proof.* The issuer's signature randomisation operation only effectively randomises the A value of the signature. Hence it is required to hide the e and v' values using a zero-knowledge proof when revealing this randomised CL signature and the attributes $\{a_i\}_{i \in D}$ to the verifier. Furthermore, the following proof hides the attributes $\{a_i\}_{i \in H}$ that the user does not wish to disclose to the verifier. We call this proof Selective Disclosure ($\mathcal{SD}$) proof.

$$PK\{(\epsilon, \nu, \{\mu_i\}_{i \in H}) : Z = A'^{\epsilon} \cdot S^{\nu} \cdot \prod_{i \in H} R_i^{\mu_i} \cdot \prod_{i \in D} R_i^{a_i} \mod n\}$$

Algorithm 6 describes the operations that have to be performed to generate the above proof of knowledge. The $\mathcal{SD}$ proof is in fact a non-interactive zero-knowledge proof in which the user hashes the randomised signature component A', aggregated commitment $\tilde{Z}$ and a *nonce* sent by the verifier. The *nonce* binds the $\mathcal{SD}$ proof to the authentication session with the verifier, thereby preventing replay attacks. The values output by the Algorithm 6 are challenge c, randomised CL signature A' and responses $\hat{e}, \hat{v}, \hat{a}_{i \in H}$ computed by the user for all the hidden values. These values constitute the transcript of an $\mathcal{SD}$ proof. This proof is sent along with the disclosed attributes $\{a_i\}_{i \in D}$ to the verifier for verification.

The verifier can verify the $\mathcal{SD}$ proof using Algorithm 7 using the issuer's public key $(n, S, Z, \{R_i\}_{i \in M})$ and the disclosed attributes $\{a_i\}_{i \in D}$. Basically, the proof proves the correctness of the issuer's signature over the disclosed attributes. The verification relies on the reconstruction of the commitments, which in this case is possible because of the equation:

$$\hat{Z} \equiv_n \tilde{Z} \tag{2.5}$$

For clarity, we show below how the equality in the above equation holds.

$$\begin{aligned}
\hat{Z} &\equiv_n Z^{-c} \cdot A'^{\hat{e}} \cdot S^{\hat{v}} \cdot \textstyle\prod_{i \in D} R_i^{c.a_i} \cdot \prod_{i \in H} R_i^{\hat{a}_i} \\
&\equiv_n Z^{-c} \cdot A'^{\tilde{e}+c \cdot e} \cdot S^{\tilde{v}+c \cdot v'} \cdot \textstyle\prod_{i \in D} R_i^{c \cdot a_i} \cdot \prod_{i \in H} R_i^{\tilde{a}_i + c \cdot a_i} \\
&\equiv_n Z^{-c} \cdot A'^{\tilde{e}} \cdot A'^{c \cdot e} \cdot S^{\tilde{v}'} \cdot S^{c \cdot v'} \cdot \textstyle\prod_{i \in D} R_i^{c \cdot a_i} \cdot \prod_{i \in H} R_i^{\tilde{a}_i} \cdot \prod_{i \in H} R_i^{c \cdot a_i} \\
&\equiv_n Z^{-c} \cdot (A'^{e} \cdot S^{v'} \cdot \textstyle\prod_{i \in M} R_i^{a_i})^{c} \cdot A'^{\tilde{e}} \cdot S^{\tilde{v}'} \cdot \prod_{i \in H} R_i^{\tilde{a}_i} \\
&\equiv_n (A'^{e} \cdot S^{v'} \cdot \textstyle\prod_{i \in M} R_i^{a_i})^{-c} \cdot (A'^{e} \cdot S^{v'} \cdot \prod_{i \in M} R_i^{a_i})^{c} \cdot A'^{\tilde{e}} \cdot S^{\tilde{v}'} \cdot \prod_{i \in H} R_i^{\tilde{a}_i} \\
&\equiv_n A'^{\tilde{e}} \cdot S^{\tilde{v}'} \cdot \textstyle\prod_{i \in H} R_i^{\tilde{a}_i} \\
&\equiv_n \tilde{Z}
\end{aligned}$$

Note that Eqn. (2.5) uses Eqn. (2.4) and hence depends on the validity of the signature used to generate this proof. The proof will verify correctly only if the triple (A, e, v) is a valid signature, that is (2.4) holds.

Algorithm 5 Randomise a Camenisch-Lysyanskaya signature [32].

```
function CL-RANDOMISE($(A, e, v), (n, S, Z, \{R_i\}_{i \in M})$)
    $r \leftarrow$ RANDOM( )
    $A' \leftarrow A \cdot S^r \mod n$
    $v' \leftarrow v - e \cdot r$
     return $(A', e, v')$
```

Algorithm 6 Generation of IRMA selective disclosure proof that proves the knowledge of a Camenisch-Lysyanskaya signature and hidden attributes [32].

```
function CL-PROVE-D($\{a_i\}_{i \in D}, (A', e, v'), nonce, (n, S, Z, \{R_i\}_{i \in M})$)
    $\tilde{e} \leftarrow$ RANDOM( )
    $\tilde{v} \leftarrow$ RANDOM( )
    $\tilde{Z} \leftarrow A'^{\tilde{e}} \cdot S^{\tilde{v}} \mod n$
    for all $i \in H$ do
        $\tilde{a}_i \leftarrow$ RANDOM( )
        $\tilde{Z} \leftarrow \tilde{Z} \cdot R_i^{\tilde{a}_i} \mod n$
    $c \leftarrow$ HASH($A', \tilde{Z}, nonce$)
    $\hat{e} \leftarrow \tilde{e} + c \cdot e$
    $\hat{v} \leftarrow \tilde{v} + c \cdot v'$
    for all $i \in H$ do
        $\hat{a}_i \leftarrow \tilde{a}_i + c \cdot a_i$
     return $(c, A', \hat{e}, \hat{v}, \hat{a}_{i \in H})$
```

Algorithm 7 Verification of the IRMA selective disclosure proof [32].

```
function CL-VERIFY-D($(c, A', \hat{e}, \hat{v}, \{\hat{a}_i\}_{i \in H}, \{a_i\}_{i \in D}), nonce, (n, S, Z, \{R_i\}_{i \in M})$)
    $\hat{Z} \leftarrow Z^{-c} \cdot A'^{\hat{e}} \cdot S^{\hat{v}} \mod n$
    for all $i \in D$ do
        $\hat{Z} \leftarrow \hat{Z} \cdot R_i^{c \cdot a_i} \mod n$
    for all $i \in H$ do
        $\hat{Z} \leftarrow \hat{Z} \cdot R_i^{\hat{a}_i} \mod n$
    if $c \neq$ HASH($A', \hat{Z}, nonce$) then return INVALID
    return VALID
```

Part I

ABC applications

Chapter 3

Attribute-based Signatures

3.1 Introduction

A signature is a proof of the signer's intent to acknowledge the contents of what is signed. In the physical world, signatures have mainly existed as handwritten signatures. A handwritten or a wet signature is usually a depiction of the signer's name or a mark made with hand and pen on the message being signed. It is considered as the signer's uniquely-personal and undeniable physical evidence that is permanently affixed on the signed message or a document. For example, a person's signature on a consumer contract provides evidence of her identity and informed consent to the terms of the contract. In many countries, handwritten signatures maybe witnessed and recorded in the presence of a public notary to carry an additional legal force[1]. The system works as long as there are strict laws to punish forgery.

Handwritten signatures are translated to the digital world as electronic signatures or eSignatures. eSignatures can be achieved in many ways. Some of them could be as simple as checking a text box or typing the signer's name at the end of an electronic document. In such cases, eSignatures can be easily forged without leaving any trace. To ensure security of signatures in the digital world, cryptography is applied.

Digital signatures are a form of eSignatures that use cryptography to demonstrate authenticity of digital messages. A valid digital signature provides the following security guarantees to the recipient of the signature: (i) authenticity: the message was signed by a known signer, (ii) integrity: the message has not been modified since it was signed, and (iii) non-repudiation: the signer cannot deny that the message was signed by her. Digital signatures are now accepted as legally binding in many countries and increasingly used for certifying contracts, notarising documents, authenticating individuals or corporations, and as components of more complex systems [33].

[1] `https://en.wikipedia.org/wiki/Signature` [last accessed: August 28, 2018]

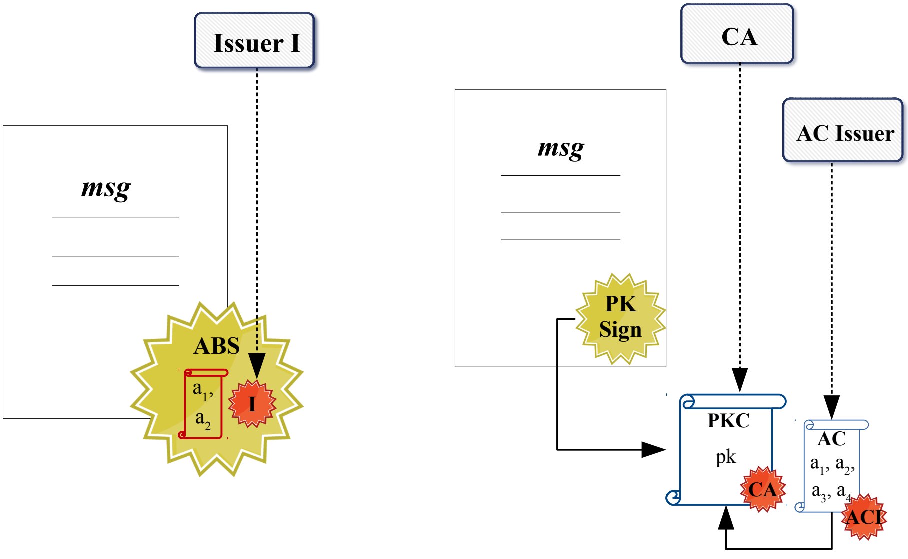
3.1.1 Public-key signatures

A digital signature scheme is typically used by a *signer* and a set of potential *verifiers*. The most widely used digital signature scheme is based on asymmetric cryptography which requires a signer to sign with her private key. The corresponding public key is used for verifying this signature. Such a signature is called *Public key (PK) signature*. To prove the authenticity of the link between the signer and her public key, the PK signature scheme relies on an underlying public key infrastructure (PKI). In the case of the most popular PKI standard X.509 [34], there are two types of certificates: public key and attribute certificates. A public key certificate (PKC) proves the signer's ownership of the public key and is signed by a Certificate Authority (CA). It contains the public key associated to the signer's private key, the signer's distinguished name and other metadata (serial number, a validity period, etc.). An attribute certificate (AC) contains arbitrary attributes of the signer and is signed by an AC issuer (possibly different from the CA which has signed the PKC). This AC is tied to the public-key certificate of the signer. In other words, an AC without an associated PKC is not valid. The signer's distinguished name field in both AC and PKC must be the same to facilitate a successful AC verification [35].

A PK signature protocol flow is described in the following steps. First, a signer generates her public-private key pair and sends a certificate request that includes the generated public key to a CA. The CA first checks the identity of the signer by some means[2]. If correct, then the CA creates, signs and returns the PKC to the signer. During signing, the signer signs the message with her private key and sends the signature along with her CA-signed PKC to a verifier. The verifier verifies the signature using the signer's public key and validates the PKC. The PKC validation involves checking the CA's signature on the contents of the PKC, whether the CA is a trusted issuer (i.e. if it is linked to a trusted root CA) and making sure that the PKC has not expired or been revoked. Additionally, if an AC was presented to the verifier by the signer along with the PKC, then the verifier checks if both AC and PKC belong to the signer and verifies the AC issuer's signature over the AC contents, along with the AC issuer's entire PKC certification. In summary, a successful verification of the PK signature, the PKC and the AC proves to the verifier that the signed message was not altered since signing, it originated from an authentic signer and the signer cannot repudiate signing the message later.

Although PK signatures provide considerable security guarantees, they are inflexible for both signers and verifiers in the following ways. First, a PK signature persistently identifies a signer based on her public key. So it cannot be used when the signer does not wish to be identified based on her signature. Second, it does not allow a signer to select which of her attributes should be included in the signature. Finally, a PK signature itself does not bear the attribute information. The attributes within an attribute certificate are only linked to a separate public key certificate. The verification of a PK signature requires only the public key in the PKC and not the attributes in the AC. So a verifier cannot verify the authenticity of the attributes

[2] Guidelines by CA/Browser Forum for CAs on how to verify the identity of web certificate applicants can be found at `https://cabforum.org/wp-content/uploads/CA-Browser-Forum-EV-Guidelines-v1.6.8.pdf` [last accessed: August 28, 2018].



(a) An ABS that includes a set of selected attributes a_1, a_2 along with the issuer's signature.

(b) A PK signature with an associated PKC with public key pk and AC with attributes a_1, a_2, a_3, a_4.

Figure 3.1: Comparison between an ABS and a PK signature.

as a part of the signature verification but has to rely only on the PKC-AC link.

Besides inflexibility, PK signatures are also privacy unfriendly for signers. In fact, these two aspects are inter-related such that the first two reasons for inflexibility leads to the following privacy issues. The public key acts as a unique identifier (both with PKC and AC) that follows a signer across all of her signatures. This makes the signer's signatures linkable irrespective of the signing context. Then, a PK signature always reveals all the information encoded in the PKC and the AC to the verifiers even if a subset of the information would have been sufficient in a given signing context. There is no possibility for minimum information disclosure. Due to the above reasons, there is a need to investigate other practical digital signature primitives that are secure, flexible, and privacy preserving.

3.1.2 Attribute-based signatures

Attribute-based signatures (ABSs) are digital signature primitives that are designed with privacy and security as main goals. Maji et al. [36] introduced the concept of ABSs which allow a person to sign a message using her secret key with fine-grained control over her identifying information. In ABSs, a signer, who possesses a set of attributes from an issuing authority, can sign a message with a predicate that is satisfied by her attributes. For example, a signer who holds attributes 'Dutch citizen' and 'doctor' can sign a message while satisfying a "Dutch doctor" predicate. The signature reveals no more than the fact that a signer, with a specific set of certified attributes satisfying a predicate has attested to the message. This can protect the privacy of the signer. An ABS is in fact a non-interactive zero-knowledge (NIZK) proof that is bound to the message, the signer's attributes and her secret key. Secret key binding achieves signer authentication. An ABS ensures the integrity of the signed message as any changes to the message or the attributes after signing invalidates the signature. Furthermore, as each attribute is bound to the secret key of a signer, different signers cannot collude to pool their attributes together into an ABS.

We consider the above notion of ABSs but we focus on specific ABSs that stem from attribute-based credentials (ABCs). An ABC contains a set of attributes belonging to an individual, is bound to a secret key and is signed by a certified issuer (similar to a CA in PKI). For example, a *student credential* which is issued to a student by her university may consist of student number, name, course identifier etc., a *doctor credential* issued to a person by a medical authority after an identity-verification step may consist of her name, role, license number, specialty, address etc. ABCs allow a user to selectively disclose and prove the authenticity of her personal attributes via NIZK proofs to a verifier (See Section 2.4 for details). In this chapter, we bind messages to the NIZK proofs over attributes and use them as attribute-based signatures. Our main idea is detailed in Section 3.1.3.

To understand the main differences between an ABS and a PK signature, we consider a simple example. A signer with four attributes a_1, a_2, a_3, a_4 wishes to sign a message msg without revealing any other information except her attributes attributes a_1 and a_2. She creates an ABS shown in the Figure 3.1a over the message and an ABC with the four attributes issued by an issuer $\mathcal{I}$. For the sake of comparison, she also creates a PK signature that is shown in Figure 3.1b. This PK signature is associated to a public key certificate that contains the signer's public key pk and an attribute certificate that contains her attributes a_1, a_2, a_3, a_4. We state the differences between an ABS and a PK signature below.

- As we can see from the Figure 3.1, the ABS bears the attributes and the attribute issuer $\mathcal{I}$'s signature in itself instead of requiring an additional PKC and an AC as in the PK signature. The verification of the signer's ABS is sufficient to assure a verifier that the attributes belong to the signer because the attributes are needed to verify the ABS. But in case an AC is presented to the verifier, then the verifier has to rely on the link between the AC and the PKC and on the validity of the PKC to be assured that the attributes in the AC indeed belongs to the signer. This is because a signer's PK signature requires only the public key of the signer and not the attributes in the AC for its verification.
- The ABS can selectively disclose a subset of attributes (e.g. only a_1 and a_2) in a signature while an AC always discloses all the attributes that it contains.
- In contrast to a PK signature, an ABS does not require a unique public key of the signer for its verification. A verifier needs to use $\mathcal{I}$'s public key and the signer's disclosed attributes to verify the validity of an ABS on a message. So

the absence of a unique identifier such as the signer's public key allows two ABSs to be unlinkable.

In summary, ABSs are great alternatives to PK signatures as they provide the same security level as in PKI while offering flexibility and privacy benefits to signers, such as ability to reveal only a subset of their personal attributes and be unlinkable.

With a real-world example, we further explain the privacy features of ABSs. Alice, who is a doctor wishes to sign a medical statement. As we have already mentioned earlier, Alice can avail selective disclosure feature with ABSs. That is, she can sign the statement msg_1 using only her profession related attributes "role=doctor" and "speciality=Orthopaedics" and hide other attributes, such as name, address and medical license number. As this signature only proves that Alice is a doctor without identifying her, it may protect her in certain situations, for instance, in reporting child abuse cases. The second privacy feature is signature unlinkability. That is, no two attribute-based signatures made by Alice can be linked to her or to each other by either the verifier or by the attribute issuer. This holds only if the disclosed attributes or the messages do not identify Alice. For instance, Alice signs a second message msg_2 with her sports club director attribute, then her signatures on msg_1 and on msg_2 are unlinkable. Furthermore, in a separate event, Alice can create an ABS over msg_3 with her "role=doctor" *and* "role=sports club director" attributes. Thus, ABSs allow Alice to combine attributes received from different issuers (in this case, medical authority and sports club) in a single signature. We note that the signatures over msg_1, msg_2 and msg_3 are unlinkable to Alice and to each other.

One might wonder why Alice cannot include her chosen attributes in the message that she wishes to sign. This would achieve the same selective disclosure and unlinkability as ABSs. Of course, she can do that. Nevertheless the authenticity of the attributes cannot be ensured in this way. That is, Alice can simply add attributes that she does not possess, in the message. Whereas in ABSs, all the attributes are bound to a signer's secret key and signed by the attribute issuer. This guarantees that a malicious signer cannot include a non-existing attribute or some other signer's attributes in her ABS. Only the authentic attributes belonging to the signer can be used to create a valid ABS.

The selective disclosure feature provides a great deal of flexibility to ABSs. This is because ABSs can be used as *context-dependant or contextual signatures.* That is, these signatures include contextual identities of a signer. We assume that there are *signature policies* that contain the list of attributes to be disclosed by the signer in an ABS. If a verifier requires a signature from a signer, then the verifier drafts a signature policy that states which attributes have to be included in the signature. The policies must be carefully designed to ask for the minimum required attributes for that particular context from the signer. However, if a signer initiates the signing by herself, then she chooses which attributes are to be included in her signature. The disclosed attributes could be identifying (e.g. name, license number), pseudonymous (e.g. an alias, a hash value) or non-identifying (e.g. student, doctor, age>18). In fact, ABSs provide privacy to signers to the extent of the disclosed attributes. In the above example, Alice's ABS over msg_1 with role and speciality attributes is an anonymous signature. Anonymous ABSs are especially useful in the scenarios such as anonymous voting, anonymous petitions, leaking secrets etc. In the scenarios in which a verifier wishes to keep track of a signer's signatures without identifying her, pseudonymous ABSs can be used. For example, Alice signs with a pseudonymous attribute from her doctor credential (e.g. the hash derived from her medical license number). More often in the real world, we come across applications for signatures which require unique identification of a signer. For example, a legal contract requires identifiers of the signing parties. In the case of Alice, a verifier may ask her to disclose a uniquely identifying attribute (e.g. her medical license number) in her ABS for accountability reasons. In sum, ABSs can be flexibly used in the applications that require full privacy (anonymity and unlinkability) as well as in the applications that require unique identification of signers.

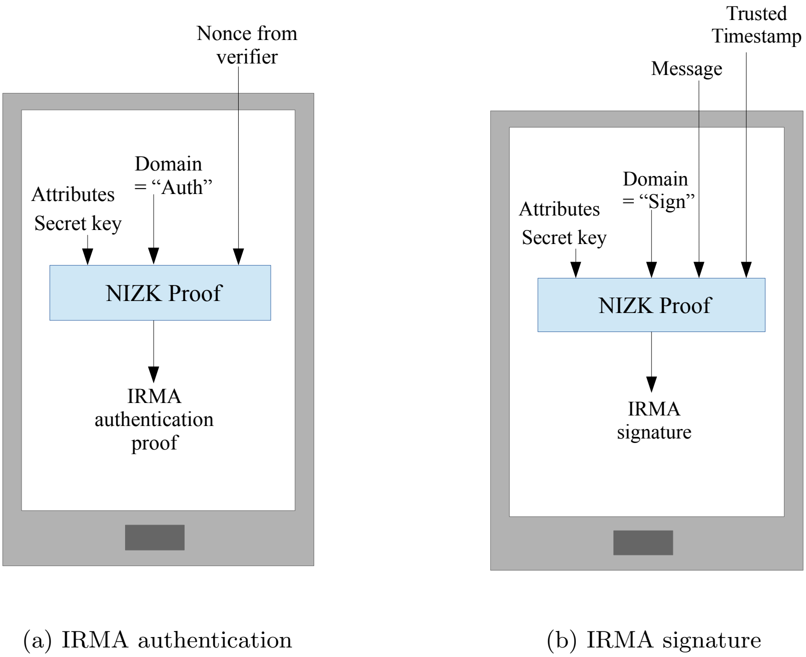
In all the above scenarios, a signer always sees which attributes are being disclosed. Therefore, prior to signing, she can assess if the disclosure is relevant and proportional in the signing context or not. Such flexiblity and signer's control over the attribute disclosure cannot be achieved with PK signatures. In summary, ABSs achieve security (message integrity, signer authenticity and non repudiation), privacy (minimum data disclosure and signature unlinkability) as well as flexibility (support for anonymous, pseudonymous or non-anonymous signatures, easy validation of the attributes and the message).

3.1.3 The idea behind ABSs in IRMA

Attribute-based signatures are powerful cryptographic primitives that provide viable and privacy-friendly alternatives to the currently widely used public-key signatures. We focus on using NIZK proofs over messages and attributes as ABSs. Although this concept is not new, there are no ABS schemes that are being applied in real-world applications. Our goal is to realise a secure and a privacy-preserving ABS construction that can be put into practice. To get there we take three steps: (i) realise an ABS construction from an operational attribute-based authentication technology – IRMA [37] (Section 3.2); (ii) propose a secure way to distinguish authentication and signatures (Section 3.3) and, (iii) propose a trusted timestamping scheme to include authentic timing information in ABSs (Section 3.4). We call the ABS construction that we get after these three steps as an *IRMA signature*. Figure 3.2 compares the authentication and signature functions in the IRMA system.

Within the IRMA project, authentication with attribute-based credentials is efficiently implemented on smart phones in the form of the *IRMA app*. This authentication is accomplished with selective disclosure of attributes and an accompanying authentication proof. This proof is an NIZK proof that proves the validity of the disclosed as well as the hidden attributes in the ABCs. We use the Fiat–Shamir heuristic [29] to create an attribute-based signature from the authentication proof. So, an ABS within IRMA is essentially a non-interactive proof of knowledge that binds the message and the signer's authentic attributes (see Section 3.2 for details). A device that carries the signer's attributes, secret key is called an *IRMA device*. In practice, the IRMA device is the signer's smart phone.

The above approach achieves our goal of putting ABSs into practice in a short span of time as extending the IRMA app's authentication functionality to support



(a) IRMA authentication

(b) IRMA signature

Figure 3.2: Comparison between an authentication proof and a signature on an IRMA device.

attribute-based signatures involves relatively little work. Then the IRMA app can perform both authentication and signature on the same device with a set of user's attributes. For example, Alice can authenticate with her age attribute at a liquor store and sign a payment message with her bank account attribute with her IRMA app. This approach is user friendly as the user can both authenticate and sign using her attributes on her smartphone. The potential ability of the IRMA app to support both functions also enables fast realisation and roll-out of the technology.

We list our main contributions below and note that our contributions are in the engineering and design side rather than in cryptography.

- We describe how to achieve both attribute-based signature and authentication functions on smartphones from the IRMA implementation of ABCs.
- We use domain separation to cryptographically separate IRMA authentication and signature functions on the smartphones and analyse the security of the system after the separation.
- The validation of signatures must be possible at any time after signing, irrespective of the status of the attributes at validation time. The only condition is that the attributes must be valid at the time of signing. Taking this into consideration, we design a trusted timestamping scheme that suits ABSs within the IRMA system and analyse its security.
- We present a practical instantiation of IRMA signatures with IRMA-Idemix

implementation and finally, describe some use cases that benefit from IRMA signatures.

3.1.4 Related work

The digital signatures involving X.509 attribute certificates [35] are comparable to ABSs but we have already mentioned the main differences between the two types of signatures. However, one could argue that PK signatures could provide unlinkability just like ABSs if domain-specific key pairs are used. This means that a signer will have to generate dedicated key pairs for each domain (e.g. for each verifier), obtain PKC for each public key and associate a fresh PKC with each of her attribute certificate. This approach removes the linkability based on a single public key and makes the resulting domain-specific PK signatures unlinkable across domains. However, this leads to complicated key management issues for signers. Moreover, even when attribute certificates (ACs) are available, it is not possible for a signer to reveal a subset of attributes from each AC to a signature verifier because the AC-issuer's signature is over all the contents of the AC and it will not verify correctly if some attributes are missing.

NIZK proofs have previously been used in the construction of signature primitives [38, 39, 40]. In particular, there have been some theoretical works on attribute-based signature schemes based on bilinear pairings [36, 41, 42, 43]. However, the proposed schemes focus mostly on the core cryptography and formally proving the security and privacy. To the best of our knowledge, none of the above schemes is put into practice.

Group and ring signatures are digital signature primitives that are designed to achieve relative anonymity of the signers by allowing them to sign as members of a certain group or a ring. In group signatures [44], a dedicated entity (group manager) runs a setup protocol and an explicit join protocol for every group member to create the respective member's signing key. Furthermore, the group manager is able to open signatures issued by group members to identify the respective signer. A group signature generally works well when a signer is acting on behalf of an organisation. But this does not work well for the group of all "over 18 years of age" worldwide. Group signatures have been incorporated in real-world applications, such as direct anonymous attestation for trusted platform modules (TPMs) and vehicle safety communications [45]. In contrast to group signatures, ring signatures do not have a dedicated group manager, they do not constrain signers to be a part of an organisation (i.e. a signer can join an "ad-hoc" ring) and they also guarantee unconditional anonymity to the signers. The main differences between these forms of signatures and ABSs are as follows.

- Group/Ring signatures only prove that the signer is a member of a group. As ABSs, they do not let the signer prove the possession of some personal attributes (e.g. name = Alice, employee ID = 56890) in the signature.
- A group signature does not allow a signer to sign as a member of multiple groups in a *single* signature. For example, to sign with two attributes, e.g. as an employee of organisation X and a member of a sports club, a signer will have

to use different signing keys for each group and generate two group signatures. Therefore, a signer has to manage n signing keys for n groups (or attributes). Moreover, the verification algorithm is run as many times as the number of attributes in the signature, thus compromising efficiency. In contrast, an ABS lets the signer create a single signature with a conjunction of attributes issued by different issuers. Thus, ABSs do not create key management issues for signers and are more efficient than group signatures.

- A verifier's signature policy may require a signer to sign with certain attributes, in other words, prove that they are members of different groups in the signature. With group signatures, it is easy for different signers who do not satisfy the signature policy to collude and create a valid signature if jointly they could satisfy the policy. For example, a signer from organisation X's sales team and her friend who is a senior manager can together create a valid group signature on a message. This attribute pooling attack does not work in ABSs because all the attributes required to create a valid ABS must be bound to a single signer's secret key [46].
- Group signatures always allow a group manager to identify the signer from her signature and ring signatures never allow any entity to learn the identity of the signer. ABSs attain a middleground by flexibly allowing the signer to be identifiable or be unconditionally anonymous based on her signatures according to the signing context.

Khader [47] proposes a notion called attribute-based group signatures (ABGS) in which only the attributes of a signer used to satisfy the predicate are revealed while hiding the signer's identity. Although this notion is similar to the ABSs that we consider, Khader's ABGS allows a group manager to identify the signer of any signature (which is similar to the semantics of group signatures [44]) as mentioned above in the last bullet point. In our view, the flexible nature of ABSs, the control over attributes that they provide to signers and more importantly, the ease with which we can realise ABSs following our approach (See Section 3.1.3) make them more attractive than the other existing privacy-preserving digital signature schemes.

Quite closely related to how we realise ABSs, the authors of ABC4Trust deliverables [48, 49] suggest the possibility of including an application-specific message as an optional input to the protocol that authenticates a user based on her attributes. Camenisch et al. [50] present a PABS scheme based on CL-signatures as a secure instantiation of the core building block: attribute-based signature. We use the same building block in our instantiation of ABS in Section 3.5. The authors provide a formal description of the cryptographic realisation of PABSs and their security and privacy properties. In this chapter however, we focus mainly on the *practical set-up* of ABSs that can be used as digital counterparts of handwritten signatures on messages or documents (e.g. forms, agreements). Such signatures embed timestamps and can be verified at any time after their creation by any verifier. Under our proposed setup, one can securely authenticate and sign messages with the same set of attributes on their smartphones. We will furthermore see in the next chapter how attribute-based signatures can be used during real-world transactions in the presence of an immediate verifier.

3.2 IRMA's Selective Disclosure Proofs as Digital Signatures

The attribute-based authentication in IRMA incorporates the data minimisation principle by means of selective disclosure of attributes. As explained in the Preliminaries (Chapter 2), selective disclosure is accomplished with a non-interactive zero-knowledge proof that proves the validity of all the attributes within a credential (both disclosed and hidden attributes) during a user authentication. As will be shown here, an IRMA selective disclosure ($\mathcal{SD}$) proof can also be used for signing a message or a document using selected attributes on her IRMA device. The key idea is to apply the Fiat–Shamir heuristic so that a *non-interactive zero-knowledge proof* (or so-called signature of knowledge) signs a message [29, 27]. When an $\mathcal{SD}$ proof is used for signing purposes, it becomes an ABS and it is written as

$$\mathcal{SD}\big\{\{a_i\}_{i\in D}\big\}\big(h_1(msg)\big) \tag{3.1}$$

where $\{a_i\}_{i\in D}$ denotes the disclosed attributes (D is the set of disclosed attributes) from the signer's credential(s) and $h_1(msg)$ is the hash of the message. We use the hash function h_1 to convert the varied-length messages to fixed-length hashes of those messages. It is simpler for the cryptographic functions to handle hashes rather than lengthy messages. An ABS proves the authenticity and integrity of the message represented by $h_1(msg)$, and the signer's possession of the attributes and of the secret key involved in the proof generation.

On a high level, $h_1(msg)$ is included in the $\mathcal{SD}$ proof in the following way. Like any zero-knowledge proof, the $\mathcal{SD}$ proof has three steps: commitment, challenge and response. In an $\mathcal{SD}$ proof, the signer first commits to her private inputs: secret key and the set of hidden attributes. This commitment hides the private values from the verifier, while it binds the signer i.e. the committer to these values, so that she cannot change them later. Then she computes a challenge by hashing the commitment and the message to be signed ($h_1(msg)$ in our case). Conceptually, the challenge is computed as

$$challenge = h_2(commitment, h_1(msg)). \tag{3.2}$$

We use the hash function h_2 to compute challenge in the $\mathcal{SD}$ proof. We use two hash functions h_1 and h_2 for messages and challenges respectively to separate the domain and the range of these hash functions [51]. As a last step in the proof computation, the signer computes the response to the challenge using her secret key and hidden attributes. The tuple ⟨commitment, challenge, response⟩ makes up an ABS.

The $\mathcal{SD}$ proofs used for IRMA authentication and signatures are different from each other. In the $\mathcal{SD}$ proof for IRMA authentication, due to its non-interactive nature, a verifier sends a nonce in order to bind the proof to the ongoing authentication session. The nonce helps the verifier to check the freshness of the proof and to prevent replay attacks. In Figure 3.2a, we can see that the NIZK proof for authentication takes a nonce as one of its inputs. However, a digital signature differs from an authentication in the following ways: signature creation can be an non-interactive

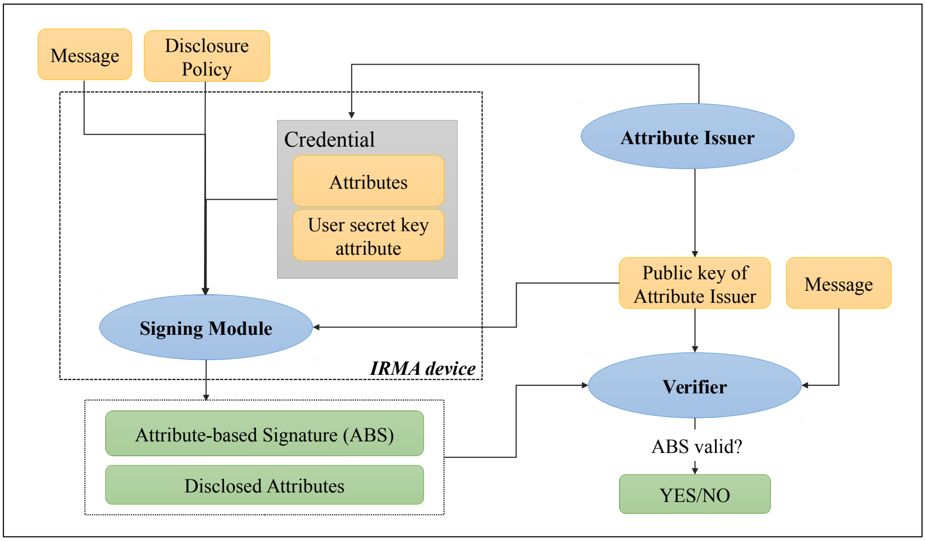


Figure 3.3: IRMA system for ABS generation and verification

operation, that is, it need not be an active session between a signer and a verifier. The verifier might even be unknown to the signer at the time of signing. Therefore, including a verifier-specific nonce is not suitable for signatures. So, we adapt the IRMA authentication approach for signatures in a simple manner. If the hash of a message is used during an $\mathcal{SD}$ proof generation instead of a nonce, then the $\mathcal{SD}$ proof becomes the user's signature on the message. In sum, the main functional difference between an $\mathcal{SD}$ proof in authentication and signatures is the way the *nonce* is defined. As a nonce and a hash value look alike, we discuss a method to distinguish between authentication and signature instances in Section 3.3. However, when there is active interaction between the signer and verifier, the verifier's nonce and the hash of the message can both be included in the $\mathcal{SD}$ proof and it would still be an ABS. Such an ABS can immediately be verified by the verifier just like an authentication proof. In the next chapter, we will apply this concept of ABSs in web transactions (e.g. online shopping) that usually consist of active interactions between a user and a verifier.

3.2.1 ABS scheme description

In this section, we will describe a scheme for realising ABSs in the IRMA system. Figure 3.3 depicts the IRMA-ABS system on a high level. The system consists of (i) an attribute issuer, who issues attributes to the IRMA device of the signer; (ii) the IRMA device that contains the signer's attributes, secret key and, a signing module; (iii) the verifier who verifies the ABS and the disclosed attributes of the signer. The signing module is capable of processing a signature policy (also called a disclosure policy)[3] and selecting the credential(s) that contains the attributes requested by the policy, prior to the signature generation. So, the module knows which attributes should be disclosed from the credential(s) and which attributes must remain hidden. Then it generates an ABS by taking the message to be signed, attributes from the credential(s) and the secret key as inputs.

The IRMA system has a common setup for both authentication and signatures. The initial setup phase consists of the following algorithms for key generation and attribute issuance.

$\mathsf{KeyGen}() \rightarrow sk_U$. When the user installs the IRMA app and uses it for the first time, KeyGen is run to generate a secret key sk_U for the user on her IRMA device. The secret key is needed for attribute issuance, authentication and signing. Thus, the user's attributes, authentication proofs and signatures are bound to sk_U and hence to the IRMA device[4] that contains the secret key.

$\mathsf{AttrIssue}(sk_{\mathcal{I}}, pk_{\mathcal{I}}, sk_U) \rightarrow C$. By running AttrIssue with an authorised attribute issuer $\mathcal{I}$, a user obtains a signed credential C with attributes $\{a_i\}_{i \in M}$ from $\mathcal{I}$ onto her IRMA device where M is the maximum number of attributes that can be issued by I. This set of attributes in C includes the secret key of the user sk_U as an attribute which is always hidden during a selective disclosure. The issuer $\mathcal{I}$ signs[5] the credential with its private signing key $sk_{\mathcal{I}}$; the corresponding public key $pk_{\mathcal{I}}$ is used by verifiers to check both authentication proofs and signatures that are created by the user's IRMA device. The attributes coming from the issuer and the attribute sk_U are bound to the credential C by the issuer's signature.

In particular, the ABS scheme in IRMA consists of the following algorithms for signature generation and verification.

$\mathsf{IRMASignGen}(C,\ msg,\ pk_{\mathcal{I}},\ \mathit{DisclosurePolicy}) \rightarrow \{a_i\}_{i \in D}, \sigma$. A signer runs IRMASignGen on the IRMA device to generate an ABS on a message msg. DisclosurePolicy determines the attributes that need to be included in the σ. The credential C contains the user's secret key attribute and the full set of attributes $\{a_i\}_{i \in M}$ from which a subset $\{a_i\}_{i \in D}$ mentioned in the DisclosurePolicy will be disclosed. IRMASignGen algorithm is at the core of the signing module shown in the Figure 3.3. It consists of two sub-algorithms:

1. $\mathsf{CRandomise}(C) \rightarrow \mathcal{C}$. CRandomise randomises the issuer $\mathcal{I}$'s signature on a credential C without changing the attributes that are contained inside the credential. The randomisation prevents the issuer's signature from becoming the unique identifier that links all the ABSs that use this credential. This achieves signature unlinkability. CRandomise is also used while computing authentication proof in IRMA to prevent linking of two proofs of the same user (See Section 2.4 for details). The output of

[3] The disclosure policy can be implemented as a JSON data type and contains the list of attributes that the user has to disclose to the service provider, along with an optional nonce and a context string.

[4] The device is further bound to the user depending on the user authentication method to access the device (e.g. PIN, security pattern or fingerprint).

[5] Here, a special credential signature scheme that allows blind signing and randomisation of the final signature is used. For e.g. CL-signature scheme (Section 2.2).

CRandomise is a randomised credential $\mathcal{C}$ that consists of the randomised issuer-signature and the same attributes as in the credential C. For simplicity, we refer to the randomised issuer's signature on the credential as the randomised credential in the rest of the chapter. When attributes from multiple credentials are to be disclosed in an ABS, then all those credentials are randomised by CRandomise before generating the ABS. The output of CRandomise is the set of randomised credentials and it is denoted by $\mathcal{C}$.

2. IRMASign($\mathcal{C}$, $\{a_i\}_{i \in M}$, msg, $pk_{\mathcal{I}}$, DisclosurePolicy) $\rightarrow$ $\{a_i\}_{i \in D}$, σ. IRMASign takes the randomised credential $\mathcal{C}$, $\{a_i\}_{i \in M}$ which is the set of all attributes in C, message msg[6], the issuer's public key parameters and the DisclosurePolicy as inputs. The parameters in $pk_{\mathcal{I}}$ are used while creating the ABS. DisclosurePolicy determines the set of attributes that should be disclosed (or rather included) in this signature. This ABS is an $\mathcal{SD}$ proof that proves the validity of all attributes in the credential and their binding to the message. The outputs are the attributes $\{a_i\}_{i \in D}$ where D is the set of disclosed attributes and an ABS. The ABS over msg resulting from IRMASign denoted as

$$\sigma = \mathcal{SD}\big\{\{a_i\}_{i \in D}\big\}\big(h_1(msg)\big) \tag{3.3}$$

IRMASignVerify(σ, msg, $\{a_i\}_{i \in D}$, $pk_{\mathcal{I}}$) $\rightarrow$ *Accept/Reject*. Any verifier who wants to verify the signature σ on msg runs IRMASignVerify. This algorithm takes the signer's signature σ, disclosed attributes $\{a_i\}_{i \in D}$, public key of the attribute issuer $pk_{\mathcal{I}}$ and the message msg as inputs. If σ is valid, then it outputs accept; otherwise reject.

3.2.2 Privacy and security assurances provided by ABSs

The ABS construction in the IRMA system is in fact an $\mathcal{SD}$ proof ensuring that "the signer has signed the message msg on her IRMA device and possesses the attributes $\{a_i\}_{i \in D}$ issued by the issuer $\mathcal{I}$". It provides all the security and privacy guarantees that we have discussed in the Section 3.1.2. For expositional clarity, we specifically state them below.

The ABSs provide privacy in terms of anonymity and signature unlinkability as long as the disclosed attributes and the signed message do not contain identifying information of the signer. Anonymous ABSs make it is impossible for a verifier or an issuer to identify the signer or link signatures to a particular signer. This aspect holds even if the signer signs the same message multiple times. The selective disclosure capability of ABSs enables context-based disclosure of signers' information. This aspect makes even the non-anonymous ABSs more privacy friendly than public-key signatures.

In terms of security, the ABSs in IRMA guarantee the following.

- Message integrity: The message is not altered after signing;

[6] In fact, a signature is always computed over the hash of the message $h_1(msg)$.

- Message authenticity: The message is signed by a signer who actually possesses the disclosed attributes at the time of signing. This is because the message is bound to the attributes by the signature.
- Attribute authenticity: The attributes involved in the signature on the message are authentic because they are signed by an authorised issuer.
- Signer authentication: The signer is authenticated based on the signature, as the message and the attributes are bound to the signer's secret key.

Non-anonymous ABSs provide two additional security guarantees:

- Signer identification: The uniquely identifying attributes included in the signature reliably identify the signer to verifiers.
- Non-repudiation: The signer cannot deny having signed the message if her unique identifier is disclosed in the signature.

The disclosed attributes in ABSs act as the public key of the signer. The attribute issuer takes the place of a CA. As in PK signatures, if the verifiers trust the issuer, then they also trust that the attributes are authentic and bound to the signer's secret key. So, if a signature contains a uniquely identifying attribute of a signer, then this signer cannot repudiate having signed the message.

As our goal is to deploy ABSs in the real world, we consider two more practical aspects that need to be handled to make our ABS construction fully secure.

1. As we wish to achieve IRMA authentication and signatures on the same device with the same secret key, we need to avoid mixing up authentication and signature instances. Therefore we discuss a standard cryptographic method to distinguish the two functions in Section 3.3.
2. An ABS needs to include a trusted timestamp that denotes the time of signing. This timestamp allows a verifier to ensure that the attributes were valid (not expired or revoked) at the time of signing. It makes signature validation possible even if the attributes included in the signature have become invalid by the time of verification. We propose a scheme to obtain authentic timestamps for our ABSs in Section 3.4.

We will call our ABSs with the above two technical constructions *IRMA signatures.*

3.3 Distinguishing IRMA Authentication and Signatures

Our goal is to use both the signature and authentication functions with the same set of attributes on the same IRMA device. An $\mathcal{SD}$ proof is used either for authentication with a fresh nonce or for signature generation with the hash of a message as input. With minimum changes to the IRMA authentication algorithm, the IRMA app can support both the functions.

However, as the hash of a message and a random nonce look alike, the following attack becomes possible. An adversary sends the hash of a message posing as a random nonce during an authentication session and makes the user unknowingly sign this hash with the $\mathcal{SD}$ proof. This is a potential attack scenario in which an authentication session is misused to get a signature of the user without the user being aware of it. In this section, we describe a standard method to inherently distinguish signature and authentication protocol runs, in order to prevent the above attack.

Although two secret keys could be used for authentication and signing to separate the two functions on the IRMA device, this would result in more complicated credential management for users. All the credentials would then have to be issued twice on that device, corresponding to both secret keys. This is because each credential on the IRMA device is bound to the secret key. Using two dedicated keys would be very similar to having authentication and signature functions on two different IRMA devices. This contradicts our original goal. In order to avoid the duplication of all attributes on the device for authentication and signing purposes, we intend to use the *same secret key* for both purposes.

Domain separation [52] is an efficient means to construct different function instances from a single underlying function. If the underlying function is secure, the derived functions can be considered as secure and independent functions. One can implement domain separation by appending or prepending different constants to the input for each of the function instances. We propose to apply *domain separation* for securely diversifying IRMA authentication and signing instances.

Basically, we reserve a variable d whose value is set to *'auth'* for authentication instances and *'sign'* for signing instances. The IRMA app can be programmed such that it takes user consent as the basis while deciding the value of d. If the user gives her consent to sign a message msg, then $d =$ '*sign*' and $\mathsf{IRMASignGen}$ function is called. Alternatively, the value of the domain separator $d =$ '*sign*' is hardcoded in the signature request that is sent to the user by a verifier. Based on the value of d, the user's IRMA app would either create an authentication proof or a signature. The app and its interface lets the user easily notice the difference between sign and authentication sessions. It is important to note that we rely on a correct device implementation. Within $\mathsf{IRMASign}$, the value of d is prepended to the inputs of the hash function h_2 while computing the challenge. We denote the modified challenge (i.e. the challenge with domain separation) as $\mathfrak{c}$. The challenge computation previously denoted by (3.2) now becomes,

$$\mathfrak{c} = h_2(d = \text{`}sign\text{'}, commitment, h_1(msg)), \tag{3.4}$$

If a valid signature is knowingly created by a signer, then during verification, any verifier can successfully check the validity of the signature by reconstructing the challenge with $d =$ '*sign*'.

3.3.1 Brief security analysis of IRMA system after domain separation

In this section, we informally analyse the security of the IRMA system after domain separation by considering the possible ways in which a passive adversary can undermine the system. We assume that the adversary $\mathcal{A}$ has access to a polynomially bounded set of IRMA authentication transcripts denoted by AT and signature-message pairs denoted by SM. First let us consider an impersonation attack in which the adversary attempts to spoof an authentication using one of the transcripts from the set AT. $\mathcal{A}$ cannot successfully authenticate to a verifier using one of transcripts from AT unless same nonce is used in the current authentication and in the old transcript. Furthermore, unforgeability of credentials and authentication proofs in IRMA (and Idemix) hold under the strong RSA assumption.

When we introduce signatures in IRMA, three more possibilities arise for the adversary to undermine the security of the IRMA system:

1. spoof an IRMA authentication by using the signatures from SM;
2. forge a new signature using the authentication transcripts from AT;
3. forge a new signature using the signature-message pairs from SM.

Case 1: Using signatures to impersonate a user during authentication. $\mathcal{A}$ attempts to authenticate with one of the attribute-based signatures from SM. We show that this is possible if she successfully finds either of the following two collisions. We also mention how we deal with such scenarios.

(i) Collision between the hash of a signed message and an authentication nonce.

$$h_1(msg) = nonce$$

where $h_1(msg)$ is the hash of a signed message msg from SM that $\mathcal{A}$ has and $nonce$ is a random number sent by the verifier during an authentication session. If $\mathcal{A}$ finds a collision between $h_1(msg)$ and a verifier's $nonce$, then she can maliciously authenticate to the verifier using her SM.

However, because of the domain separation (see Section 3.3), $\mathcal{A}$ cannot make the verifier accept this signature as a valid authentication proof.

(ii) Collision between the hash functions used for computing challenge in signature and authentication instances.

$$h_2(d = \text{`}sign\text{'}, commitment_s, h_1(msg)) = h_2(d = \text{`}auth\text{'}, commitment_a, nonce)$$

where h_2 is the hash function used for computing the challenge within an $\mathcal{SD}$ proof. The inputs for h_2 in a signature from SM and an authentication transcript from AT are different. The attack succeeds if $\mathcal{A}$ finds a collision between these two h_2 instances. This implies that both hash instances output the same hash value in spite of different inputs. If $\mathcal{A}$ manages to find the above collision then she wins – she can then authenticate with a signature. We note that $\mathcal{A}$'s chances of winning in this scenario depends on the collision resistance of the hash function h_2 that is used in IRMA. If hash functions with no known collision attacks, such as SHA-2 or SHA-3 are used then the above attack is highly improbable.

Case 2: Using IRMA authentication transcripts to forge a new signature. $\mathcal{A}$ eavesdrops on many IRMA authentication sessions and collects authentication transcripts AT. Then she tries to forge an ABS out of an authentication transcript in the set AT. She is successful if she finds a collision in the two scenarios detailed in Case 1.

Case 3: Using signature-message pairs to forge a new ABS. As we said before, $\mathcal{A}$ possesses ABSs for several messages of a user. In this case, the domain separator d is *'sign'* for all signatures that $\mathcal{A}$ already has. Let us call the ABS that is created by signer S over msg_1 as σ_{old}. Assume that $\mathcal{A}$ wishes to forge a new ABS σ_{new} from σ_{old}. The new ABS σ_{new} can be of three types.

1. A signature on a new message msg_2 but with the same attributes as in σ_{old}.
2. A signature on the same message msg_1 as in σ_{old} but with different attributes belonging to the signer S.
3. A signature on a new message msg_2 and also with different attributes of S.

In the first type of σ_{new}, if $\mathcal{A}$ manages to find a collision between $h_1(msg_1)$ and $h_1(msg_2)$, then she can forge a valid signature. Here the unforgeability of an ABS depends on the strength of the underlying hash function h_1 to be collision resistant. In the second and third σ_{new}, it is not possible for $\mathcal{A}$ to forge an ABS without having the knowledge of S's attributes and the secret key. Unforgeability of ABSs is ensured by the same security assumptions underlying the IRMA (Idemix) technology. So we conclude that the adversary will not succeed in forging a new, valid ABS over attributes different from the ones used in the given signature-message pairs, even if those pairs are of adversary's choice. Thus, an ABS within the IRMA system is *existentially unforgeable* under a chosen-message attack.

3.4 Timestamps in Attribute-based Signatures

Similar to identity documents and public key certificates, IRMA credentials carry expiry dates. In IRMA, the expiry date of a credential is encoded as an attribute within the credential. It marks the expiry of all the attributes contained in the credential. This attribute is bound to the $\mathcal{SD}$ proof and always disclosed by default. If attributes from n credentials are disclosed then, n expiry dates are disclosed to the verifier. As IRMA authentication is usually done in an active session between a user and a verifier, the verifier can immediately check the expiry date(s) of the attributes disclosed with the $\mathcal{SD}$ proof. Furthermore, the verifier is certain that the authentication proof is fresh and bound to the current authentication session as the verifier's nonce is included in the proof. However, in the case of a signature, there may not be an active session between a signer and a verifier during which a freshness nonce is exchanged; in other words, a signer may sign a message without knowing when the signature will be verified by which verifier. Nonetheless, at any time after the signature is generated, a verifier must be able to check that the signer's attributes included in an ABS were valid at the time of signing.

Apparently, there are two time elements in our ABS construction:

- Expiry date of the signer's attributes included in an ABS.

- Signature timestamp that denotes the date and time at which the ABS was generated.

A signature timestamp is crucial for ABSs for several practical reasons. First, the timestamp allows an ABS-verifier to verify if the attributes were valid when the signer signed the message. Then, the timestamp provides an unequivocal proof that the signed message existed at a point-in-time and has not changed since then, regardless of the status of the attributes at the time of validation (i.e. if they are expired or revoked). Furthermore, timestamps play an important role in maintaining the security guarantees provided by digital signatures in general. Without a timestamp, we can not trust a signed message when the signer's private key was lost, stolen, or compromised. We neither can solve the cases when the signer herself repudiates the signing, claiming that has accidentally lost her private key, since we cannot know when the message was actually signed.

In the case of ABSs that involve the possibility of delayed verification, in addition to verifying the validity of the signature, a verifier checks if the expiry date of the disclosed attributes is greater than the signature timestamp. If this check fails, then the verifier rejects the ABS. When attributes from n credentials are disclosed with a signature, then the verifier checks whether the earliest of the n disclosed expiry dates is greater than the signing date. In this section, we denote the expiry date of the credential involved in the ABS (the least expiry date, if multiple expiry dates are disclosed) by ε and the signing time in the timestamp as t.

There are two ways to determine the timestamp t for a signature in IRMA. First, the signer can include the local time of her IRMA device as the timestamp t in the message to be signed. This is the most easily implementable solution for ABS-timestamps. But it requires the verifiers in the IRMA system to trust all IRMA devices to include accurate time information in ABSs. If verifiers require a more secure and a trusted timestamp than the ones included by the IRMA devices, there is a second solution: a signer can obtain a timestamp t signed by a trusted timestamp authority. In both the above solutions, during ABS verification, verifiers need to check if the attribute expiry date ε of the disclosed attributes is greater than the signature timestamp t. If the attributes were already expired at the time of signing, then the signer's ABS is rejected. In the next subsection, we elaborate a scheme for the second timestamping solution.

3.4.1 Timestamping scheme description

In this section, we describe in detail how a signer can acquire trusted timestamps for attribute-based signatures from a Timestamp Authority (TSA). In fact, it is not the signer herself but her IRMA device that interacts with the TSA. For simplicity, we use the term 'signer' instead of the IRMA device throughout this section. A TSA is a trusted third party that offers evidence that specific data existed at a certain point in time and guarantees the correctness of the time parameter. In our timestamping scheme, we assume that the TSA is honest, but curious. That is, the TSA is trusted to include correct times in the timestamp but it is is curious to learn about a signer and her messages. Basically, the TSA can work alone or collude with verifiers to

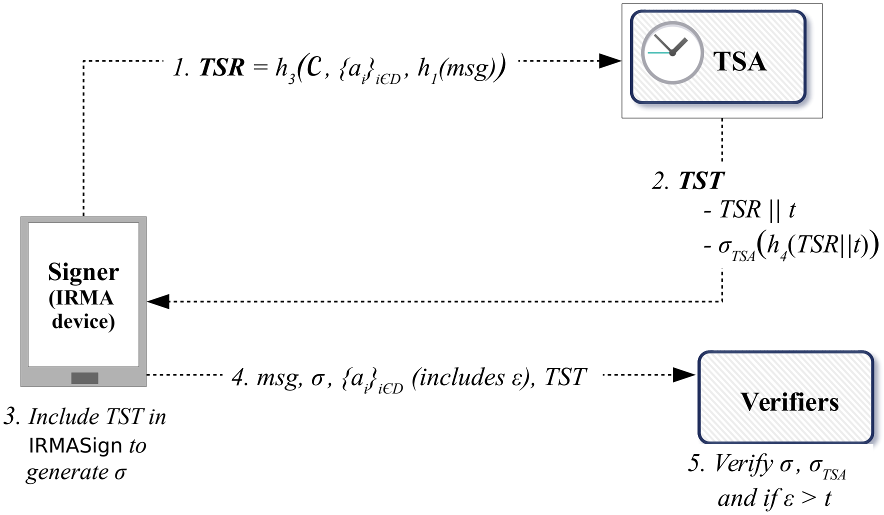

1. TSR = $h_3(C, \{a_i\}_{i \in D}, h_1(msg))$
TSA
2. TST
- TSR || t
- $\sigma_{TSA}(h_4(TSR||t))$
Signer (IRMA device)
3. Include TST in IRMASign to generate σ
4. msg, σ, $\{a_i\}_{i \in D}$ (includes ε), TST
Verifiers
5. Verify σ, σ_{TSA} and if ε > t

Figure 3.4: Timestamping scheme for ABSs.

The timestamp token TST that includes the signer's request TSR, the time t and the TSA's signature σ_{TSA} is included in the ABS σ. In addition to verifying σ and σ_{TSA}, the verifier checks if the disclosed attributes' expiry date ε is greater than the time t in token TST. The fine dashed arrows in the figure indicates that the communication between the signer, the TSA and the verifier is carried out over an anonymous network such as Tor.

compromise a signer's privacy. It is imperative for us that the timestamping scheme is secure but also privacy preserving i.e., it does not weaken the inherent privacy guarantees provided by ABSs. Essentially, this means that the TSA should neither identify a signer nor link two signatures made by the same signer based on the timestamp requests or the final signatures (provided, they are anonymous ABSs).

The timestamping scheme consists of a signer, a TSA and an ABS verifier. The TSA holds a private signing key sk_{TSA} and a corresponding public key pk_{TSA}. We assume that the TSA's public key is known to all the signers and the verifiers in the IRMA-ABS system. On a high level, our timestamping scheme works as follows. The TSA adds an authentic timestamp to a timestamp request that it gets from a signer, combines both into a timestamp token and signs the token with its signing key sk_{TSA}. The signer includes this signed token in her ABS, thus creating a timestamped ABS. An ABS verifier verifies the validity of both the ABS and the timestamp token at verification time.

To achieve privacy against a curious TSA, the signer has to remain unidentified from the TSA both on network and data levels. So in the timestamping scheme, the signer connects to the TSA over an anonymous network (e.g. Tor) through her IRMA device. This prevents the TSA from learning any identifying information about the IRMA device or of the signer that is revealed by the network (e.g. device IP address). Then, the signer creates a timestamp request such that it does not reveal any identifier or the message over which the ABS will be generated. In our scheme, the signer's request is simply a hash value that is bound to the message and the credential that will be used for creating an ABS. The TSA just adds the timestamp to this hash value, signs and sends it back to the signer. Our timestamping scheme ensures that the TSA cannot deanonymise the signer even if it sees the anonymous ABS generated by that signer at a verifier (in the case of collusion).

In the IRMA system, we envision the timestamping functionality to be embedded within the signing module. This function is executed as a part of $\mathsf{IRMASignGen}$ – after $\mathsf{CRandomise}$ and before $\mathsf{IRMASign}$ (See Section 3.2.1 for description of the algorithms). All the credentials that are required by a signature policy are randomised by $\mathsf{CRandomise}$ before generating the signature. The set of randomised credentials involved in an ABS (output of $\mathsf{CRandomise}$) is denoted by $\mathcal{C}$ and the attributes that will be disclosed in the ABS by $\{a_i\}_{i \in D}$. Furthermore, we denote the hash functions used in the timestamping scheme by h_3 and h_4. By this, we separate the domains and ranges of the hash functions used in timestamping from the hash functions h_1 and h_2 used in the previous sections for computing message digests and challenges in the zero-knowledge proofs. The timestamping for ABSs in the IRMA system takes place as described below and also in Figure 3.4.

1. **Timestamp request.** A signer constructs a timestamp request TSR by computing the hash of the randomised credential(s) $\mathcal{C}$, hash of the message to be signed $h_1(msg)$, the attributes that will be disclosed in the ABS $\{a_i\}_{i \in D}$. The set of disclosed attributes include their expiry date(s) ε. Then the signer sends the request $TSR = h_3(\mathcal{C}, \{a_i\}_{i \in D}, h_1(msg))$ to the TSA. Basically, this request aims to bind a run of $\mathsf{IRMASign}$ algorithm to the timestamp. Thus, the timestamp will be valid only for the ABS that results from that particular run of $\mathsf{IRMASign}$.

2. **Timestamp token calculation.** Upon receiving a timestamp request, the TSA concatenates it with the current timestamp t. Then the TSA digitally signs the hash of $TSR||t$ as $\sigma_{TSA} = \mathsf{Sign}_{sk_{TSA}}(h_3(TSR||t))$ where Sign is a generic algorithm that generates a digital signature (e.g. PK signature, ABS). Finally, the TSA sends to the signer a timestamp token TST that consists of two values: $TSR||t$ and its signature σ_{TSA}.

3. **Timestamped ABS generation.** The signer verifies the TSA's signature using pk_{TSA}. If correct, she provides the timestamp token TST as an additional input to $\mathsf{IRMASign}$ algorithm. Now, during the challenge computation, TST is hashed along with the domain separator d, commitment and hash of the message. The equation 3.4 now becomes

$$\mathfrak{c} = h_2(d = \text{‘}sign\text{’}, commitment, h_1(msg), TST). \tag{3.5}$$

   The resulting signature (i.e. output of $\mathsf{IRMASign}$ with the addition of timestamp token) is a timestamped ABS over $h_1(msg)$, timestamp token TST and disclosed attributes $\{a_i\}_{i \in D}$. It is denoted by

$$\sigma = \mathcal{SD}\big\{\{a_i\}_{i \in D}\big\}\big(h_1(msg), TST\big) \tag{3.6}$$

4. **Timestamped ABS verification.** Any verifier who knows the public key of

the attribute issuer $\mathcal{I}$[7] and that of the timestamp authority TSA can verify a timestamped ABS by carrying out the following checks. The verifier validates the ABS σ on $h_1(msg)$ by running $\mathsf{IRMASignVerify}$ protocol with $pk_{\mathcal{I}}$, $\{a_i\}_{i \in D}$, $h_1(msg)$, and TST (timestamp token received along with the ABS from the signer). In addition, the verifier checks (i) if the values of $\mathcal{C}, h_1(msg), \{a_i\}_{i \in D}$ in σ and in the timestamp token TST are the same; (ii) if the disclosed attributes' expiry date $\varepsilon > t$, where t is the time included in TST by the TSA and, (iii) if the TSA's signature included in TST verifies correctly with the TSA's public key pk_{TSA}.

A timestamped ABS provides a cryptographic assurance of the following properties to verifiers.

- The ABS was not generated before the time indicated by the TSA's timestamp t. The inputs for a particular run of the ABS generation algorithm were committed before t and not created or modified after t. As a result, a signer can not repudiate that the timestamped data was in her possession at time t.
- The signature on the message with the enclosed attributes is bound to the timestamp t. That is, the same timestamp cannot be used with a different message and different attributes.
- The message that is signed has remained unchanged since the time t.

Our timestamping scheme for ABSs has a similar construction as one of the timestamping methods described in NIST Special Publication 800-102 [53]. In this method, the signer provides a hash value to the timestamp authority, gets a timestamp token that contains the user-supplied hash value, the timestamp and the TSA's signature on these data, includes this token in her signature, and sends the message, timestamp token and her final signature to the verifier. The differences are that our timestamping scheme is customised for an attribute-based credentials setting and focuses on protecting signer's privacy towards a timestamping authority.

3.4.2 Brief analysis of IRMA timestamping scheme

Our timestamping scheme achieves trusted timestamps that are strongly bound to the message, signer's credential and the ABS. In this section, we informally analyse the security and privacy of our ABS construction when the timestamping functionality is embedded in the ABS generation. We consider a passive adversary $\mathcal{A}$ which has access to all the communications between the signer, the TSA and the verifier. Thus she can correlate the views of the TSA and the verifiers. Further, we assume that $\mathcal{A}$ has access to a polynomially bounded set of timestamped ABS-message pairs.

Our analysis focusses on two basic goals:

1. Security goal – To ensure unforgeability of timestamped ABSs and,
2. Privacy goal – To ensure that the timestamping does not lead to signer identification or linkability in the case of anonymous ABSs.

[7] If many credentials are involved in the ABS then, the verifier needs to know the public keys of all the issuers responsible for the credentials.

The first goal is achieved only if both the timestamps and the ABSs are unforgeable. The second goal is achieved when neither the timestamp nor the ABS reveals any identifying information of the signer. Below we discuss the cases in which the adversary $\mathcal{A}$ can violate the above goals and describe how the design of timestamped ABSs prevents such violations. The first three cases concern the security goal while the last two concern the privacy goal.

Case 1: $\mathcal{A}$ attempts to modify a timestamp on an ABS without the help of the TSA. This attack is not possible because every timestamp carries the TSA's signature. Any change made to the timestamp will invalidate the TSA's signature. This in turn invalidates the ABS.

Case 2: $\mathcal{A}$ attempts to use the same timestamp for multiple messages. This case is not possible because the timestamp in a timestamped ABS is bound to a message $h_1(msg)$, randomised credential $\mathcal{C}$ (unique for every signature) and the attributes $\{a_i\}_{i \in D}$ as orginally intended by a signer. Once the ABS is generated, even the signer cannot reuse the timestamp for a different message; she can only get new timestamps for the message from the TSA and generate new ABSs.

Although the ability to use the same timestamp for multiple messages may seem like a desirable feature in some business cases (e.g. a company requests its employee to sign multiple documents on a specific day), this might also lead to security vulnerabilities that lower the trust in the timestamp on the signature. Because the timestamp is not bound to a specific message, the adversary might use an old timestamp to sign a message in the present (e.g. sign a message in 2018 with a timestamp belonging to the year 2000). This can cause serious problems if for instance, $\mathcal{A}$ puts an older timestamp on her patent application, thereby disqualifying the legitimate patent applications that were signed and submitted on dates later than the date in $\mathcal{A}$'s timestamp. Another example is that organisations can backdate signed reports and documents without detection. To prevent such wrongdoings, we bind the message and attributes to the timestamp. For the same purpose, the ANSI ASC X9.95 standard [54] for trusted timestamping also mandates that timestamps are bound to signers' signatures.

Case 3: $\mathcal{A}$ removes the timestamp from a timestamped ABS and appends a new timestamp. An adversary cannot succeed in carrying out this attack in our timestamping scheme because, by changing the timestamp, $\mathcal{A}$ invalidates the signer's ABS. As the timestamp token TST is embedded into the signature during signing ($\mathsf{IRMASign}$), the signature will not verify correctly without the same token.

In the case of ANSI ASC X9.95 standard, the timestamp token includes the signer's digital signature but vice versa is not true. That is, an adversary (man-in-the-middle or an adversary who controls the verifier) who learns a signer's final timestamped signature can remove the timestamp and get a new timestamp on the same signature from a timestamp authority. Such an attack when translated to the real world can lead to a possible rejection of the signer's signature or have adverse consequences as in the case of patent application example.

Case 4: $\mathcal{A}$ tries to learn the signer's identity from the timestamp request of the signer. This is not possible because the timestamp request (TSR) is a hash value that does not reveal any identifying information to an adversary $\mathcal{A}$ even if it colludes with TSA. Furthermore, TSR is different for every ABS which makes it impossible for the adversary to link any two timestamp requests to a particular signer.

Case 5: In collusion with the TSA and the verifier, $\mathcal{A}$ tries to link an anonymous ABS to a signer based on the timestamp. The time t in the timestamped ABS does not uniquely identify or link the signer because of two reasons: (a) The TSA does not know the signer's identity during timestamp retrieval as both timestamp request and the underlying communication channel between the the signer and the TSA are anonymous; (b) The verifier cannot identify or link the signer due to the inherent privacy guarantees of anonymous ABSs.

Although the above security and privacy analysis of timestamped ABSs is not rigorous, it is sufficient to rule out practical attacks by the adversary that would undermine the IRMA-ABS system. In other words, adding trusted timestamps to our ABS construction within the IRMA system does not compromise its inherent security and privacy guarantees.

3.5 Practical Instantiation of IRMA Signatures

As mentioned in Section 3.1.3, we call the ABS construction within the IRMA system that includes secure domain separation (Section 3.3) and trusted timestamps (Section 3.4) as *IRMA signatures*. IRMA implements ABCs based on IBM's Idemix technology [55]. In this section, we describe the signature generation and verification algorithms with the IRMA-Idemix implementation [32, 37].

Let us consider a signer who owns an IRMA device. Upon the installation of the IRMA app, KeyGen algorithm is run first to generate a secret key sk_U for the signer on her device. Then a credential with attributes $\{a_i\}_{i\in M}$ is issued by issuer $\mathcal{I}$ to the signer using $\mathsf{AttrIssue}$ algorithm. M denotes the set of attribute indices, and hence the maximum number of attributes issued by that issuer. The secret key sk_U is encoded as an attribute in the issued credential and it is never disclosed (not even to the issuer). The expiry date of the credential is also encoded as an attribute and it is always disclosed during every selective disclosure. The conditions about never disclosing the secret key attribute and always disclosing expiry date attribute are hardcoded on the IRMA device. The public key of the attribute issuer $pk_{\mathcal{I}}$ consists of the parameters $(n, S, Z, \{R_i\}_{i\in M})$. The credential is signed by $\mathcal{I}$ using the CL-signature scheme (See Section 2.2). The issuer's CL-signature is denoted by (A, e, v) where A is the signature component, e is a randomly chosen prime and v is a randomly chosen integer.

When a signer uses the same credential multiple times to sign, then her signatures can possibly be linked to the signer based on the issuer's CL-signature. To prevent this linkability, the CL signature is randomised every time before using the credential for signing. During IRMASignGen, the issuer's CL signature (A, e, v) is first randomised as (A', e, v') by Crandomise. The randomised credential $\mathcal{C}$ used in Section 3.2.1 and 3.4.1 is equivalent to (A', e, v') in this section.

After the randomisation, the signer (IRMA device) sends the timestamp request to the timestamp authority TSA and gets a trusted timestamp token TST for the current signature as described in Section 3.4.1.

The IRMASign algorithm in Section 3.2.1 now takes two additional inputs: domain separator d and the timestamp token TST to generate an IRMA signature σ_{IRMA} as shown in Algorithm 8. One of the inputs to the algorithm, the DisclosurePolicy instructs the algorithm which set of attributes need to be disclosed from the credential(s) or rather included in σ_{IRMA}. This set of attributes is denoted by D and the set of attributes that needs to be hidden from the verifier is denoted by H. In IRMASign, the hash function for computing the challenge $\mathfrak{c}$ (Line 8 in Algorithm 8) takes the domain separator $d =$ '*sign*', the randomised CL signature A', the commitment $\tilde{Z}$[8], hash of the message $h_1(msg)$ and timestamp token TST as the inputs. The resulting signature σ_{IRMA} is the set of values that is output by Algorithm 8. The signature can then be verified with IRMASignVerify as shown in Algorithm 9. As described in Section 2.4, the verification relies on the reconstruction of the commitments, which is possible because of the equation (2.5). Only if σ_{IRMA} is valid, the value $\hat{Z}$ computed by the verifier in the Line 6 of Algorithm 9 equals the signer's commitment $\tilde{Z}$ computed in the Line 7 in Algorithm 8. Otherwise, the verification equation in Line 7 in Algorithm 9 fails and the output is that σ_{IRMA} is invalid.

In sum, IRMA's signature generation and verification algorithms 8 and 9 are equivalent to IRMA authentication proof creation and verification algorithms 6 and 7. The only difference is that the input to the hash function h_2 in IRMASign and IRMASignVerify includes the domain separator tag $d =$ '*sign*', the hashed message $h_1(msg)$ and the timestamp token TST.

3.5.1 Hash functions

In the IRMA signature system, hash functions serve four purposes: (i) to create a digest for the message to be signed (msg), (ii) to compute the challenge in an $\mathcal{SD}$ proof, (iii) to compute the timestamp request (TSR) and, (iv) to compute a digest for timestamp request and the timestamp before the TSA signs it. The hash instances (i), (ii) and (iii) are computed by a signer's IRMA device and the hash instance (iv) is computed by the TSA. Instead of using a single hash function, we use four different hash functions h_1, h_2, h_3 and h_4 to achieve the above purposes respectively. Different hash functions h_i are used because, the inputs to the hash functions are in different domains and we want to ensure that the outputs from these hash instances are unrelated to each other. In practice, these hash functions can be derived from a single hash function h using the equation $h_i(x) = h(x||i)$ (where $||$

[8] $\tilde{Z}$ is an aggregated commitment in which the signer commits to her private inputs: secret key and the hidden attributes.

Algorithm 8 IRMA signature generation algorithm [32].

```
function IRMASign($\{a_i\}_{i \in M}, (A', e, v'), h_1(msg), d, TST, (n, S, Z, \{R_i\}_{i \in M})$, DisclosurePolicy)
    $\tilde{e} \leftarrow$ Random( )
    $\tilde{v} \leftarrow$ Random( )
    $\tilde{Z} \leftarrow A'^{\tilde{e}} \cdot S^{\tilde{v}} \mod n$        //commit to the issuer's randomised CL-signature
    for all $i \in H$ do
        $\tilde{a}_i \leftarrow$ Random( )
        $\tilde{Z} \leftarrow \tilde{Z} \cdot R_i^{\tilde{a}_i} \mod n$        //commit to the hidden attributes
    $\mathfrak{c} \leftarrow h_2(d, A', \tilde{Z}, h_1(msg), TST)$  //compute challenge with $d =$ '$sign$', $TST$
    $\hat{e} \leftarrow \tilde{e} + \mathfrak{c} \cdot e$
    $\hat{v} \leftarrow \tilde{v} + \mathfrak{c} \cdot v'$
    for all $i \in H$ do
        $\hat{a}_i \leftarrow \tilde{a}_i + \mathfrak{c} \cdot a_i$
    return $(\mathfrak{c}, A', \hat{e}, \hat{v}, \{\hat{a}_i\}_{i \in H}, TST)$
```

Algorithm 9 IRMA signature verification algorithm [32].

```
function IRMASignVerify($(\mathfrak{c}, A', \hat{e}, \hat{v}, \{\hat{a}_i\}_{i \in H}, \{a_i\}_{i \in D}), h_1(msg), d, TST, (n, S, Z, \{R_i\}_{i \in M})$)
    $\hat{Z} \leftarrow Z^{-\mathfrak{c}} \cdot A'^{\hat{e}} \cdot S^{\hat{v}} \mod n$
    for all $i \in D$ do
        $\hat{Z} \leftarrow \hat{Z} \cdot R_i^{\mathfrak{c} \cdot a_i} \mod n$
    for all $i \in H$ do
        $\hat{Z} \leftarrow \hat{Z} \cdot R_i^{\hat{a}_i} \mod n$        // if $\sigma_{IRMA}$ is valid, then $\hat{Z} = \tilde{Z}$
    if $\mathfrak{c} \neq h_2(d, A', \hat{Z}, h_1(msg), TST)$ then return Invalid      // $d =$ '$sign$'
    return Valid
```

denotes concatenation of bit strings).

3.6 Discussion

In this section, we briefly discuss the revocation in attribute-based signatures and also how the trust relations in ABSs are comparable to those in public-key signatures. Then we describe a few use cases for IRMA signatures and give an estimate of their performance to demonstrate their applicabilty in the real world.

3.6.1 Revocation in ABSs

As we have seen in the Section 3.4, attribute-based credentials that are used in ABSs have expiry dates similar to identity documents or public key certificates. However, there are situations that require the use of these credentials to be suspended well before their expiration. For instance, (i) when the device on which the credentials are stored is lost or stolen, (ii) when the credential no more holds for that user (e.g. an employee credential is suspended when she leaves the company). This is also necessary when the owner of the credential herself abuses it. The suspension action is generally known as *revocation*. In the case of ABSs, it is important to ensure that a signer's secret key and her credentials can be easily revoked in such situations. If an IRMA user loses her IRMA device, then she initiates revocation that prevents the further signing operations on her lost device. This is termed as user-driven revocation. However, if the IRMA system (i.e. issuers or verifiers) performs revocation either because the credential no more holds for a user or if the user misbehaves, then such a revocation is termed as system-driven revocation.

For a general understanding, let us first look at how public-key certificates are revoked in PKI. When a signer's private key is lost or stolen, her keypair is revoked and her public-key certificate is added to a certificate revocation list (CRL). Sometimes, the CRL is also referred as a *blacklist*. During verification of a public-key signature, a verifier checks if the signer's public key is present in the CRL [34]. Alternatively, the verifier queries the revocation status from the PKC issuer using Online Certificate Status Protocol (OCSP) [56]. If the PKC has been revoked, then the signer's PK signature is rejected. Here it is apparent that a unique identifier such as the public key of the signer is required to carry out the revocation check.

In the case of a non-anonymous ABS, the identifying attributes of the signer disclosed in the ABS can be used to check the revocation status of the signer's credentials as in the case of PK signatures. If the signer's attributes are recognisable, then the revocation is simple and fast for both signers and verifiers.

However, revocation no more remains simple in anonymous ABSs. If no unique identifier of a signer is revealed in the signature, a verifier cannot check if the signer's attributes are revoked or not. Revocation has been widely studied in the literature; we refer to, for example, Lapon et al. [57] for an overview of current revocation techniques for attribute-based credentials. Within the IRMA project, Lueks et al. have proposed an efficient revocation scheme [58] for IRMA authentication that avoids identifiers in revocation (such identifiers would enable linking the revocation checks to a single user). This scheme splits the time into epochs and uses epoch-specific and verifier-specific values for generating revocation tokens. During an authentication, a verifier can check if the user's revocation token is present in a revocation list that is specific to the verifier and to the epoch. Thus, verifiers can verify the revocation status of users' credentials based on the revocation tokens without being able to identify or link them across different epochs or domains of other verifiers. As we focus on an ABS construction in the IRMA system, we will analyse if we can use this particular revocation scheme for revoking credentials even in our signature setting.

In the case of signatures, a signer need not know the verifiers in advance, so, the verifier-specific revocation tokens would not work. Also, the per-epoch concept will have to be modified to suit the signature verification scenario. If the signer calculates a revocation token for the current epoch, the verifier has to do a revocation check in that epoch. In the case of a delayed verification in a different epoch, the verifier will have to retrieve the revocation token list corresponding to the epoch in which the signer has signed the message, which is very inefficient. Thus, it is not trivial to adapt the above revocation mechanism designed for IRMA authentication to ABSs. This is a topic for future research.

At present, there are two possible solutions for revocation in the case of anonymous ABSs.

- As suggested by Alpar et al. [59], attribute expiry dates can be used for revocation. This can be achieved by making the expiration times very short and re-issuing of attributes simple. If a security breach or a key compromise is detected then the attribute issuer would just stop re-issuing the attributes to that particular IRMA device.
- We can achieve instantaneous blocking of the signing keys using a central server as described in Chapter 5. In this approach, the signing key is cryptographically shared between a signer's IRMA device and a central server which necessitates participation of both entites in a signature generation. This allows the signer to block her keyshare at the server instantly if the corresponding IRMA device is lost or corrupted. It also allows the system (via the server) to limit the rate of signing operations by a signer per unit time.

3.6.2 Trust relations in IRMA signatures

An IRMA signature is comparable to a timestamped PK signature with respect to the trust relations assumed in IRMA and PKI. In the case of PK signatures, there is a cryptographic relation between the private key (or the secret key) and the public key of the signer. The public key is signed by a CA attesting that the link between the signer and her public key is authentic. The expiry date of the public key is also decided and signed by the CA. A verifier uses the signer's public key to verify her PK signature and the CA's public key to verify the authenticity of the signer's public key. The verifier must trust the CA for the authenticity and expiry date of the user's public key. In the case of an IRMA signature, there is a cryptographic relation between the signer's secret key, attributes and the attribute issuer's public key. There is no unique public key for a signer but the disclosed attributes and the issuer's public key constitutes the verification key for an IRMA signature. A verifier must trust the attribute issuer for the authenticity and expiry dates of the signer's attributes and that these attributes are strongly bound to the signer's secret key. The attribute issuer in IRMA can be compared to the CA in PKI. In both timetamped PK signatures and ABSs, a verifier needs to trust the timestamp authority TSA for the correctness of the time at which a signature is created.

It is important to understand that an IRMA signature comprises multiple logical layers. An IRMA signature embeds the signer's attributes, attribute issuer's signature, message and the TSA's timestamp during its generation. The validity of an IRMA signature implies the validity of the attributes at the time of signing and the issuer's signature on the attributes. Table 3.1 summarises the three signatures that have to be verified during the verification of an IRMA signature. We emphasise that the verification of an IRMA signature integrates the verification of the attribute issuer's signature as it is a proof of knowledge of the issuer's signature and disclosed attributes. So, explicitly, a verifier is required to verify (i) the IRMA signature using the disclosed attributes and the issuer's public key and, (ii) the TSA's signature using the TSA's public key. In this section, our focus is on the signatures. However, in addition to verifying the signatures, the verifier checks other aspects such as, the validity of the attributes w.r.t. timestamp and expiry dates, the revocation status of signer's credentials, the revocation status of the issuer's private key.

Table 3.1: Abstraction of signature layers involved in IRMA signatures.

Signature	**Public key used to verify**	**Signing party**
IRMA signature (σ_{IRMA})	Disclosed attributes	Signer
Credential signature ($\mathcal{C}$)	Issuer's public key	Issuer
TSA's signature (σ_{TSA})	TSA's public key	TSA

In the context of IRMA signatures (ABSs in general), we also discuss the difference in the trust models in the following two cases:

- when a signer's identity information (e.g. social security number, student) is included as normal text in the message to be signed, and
- when a signer's identity information is included as an attribute in the signature.

In the first case, a verifier trusts the signer to have included authentic identity information in the data that is signed and in the second case, the verifier trusts the attribute issuer regarding the authenticity of the signer's identity information that is included in the signer's signature. We explain this with an example. There are two IRMA signatures $\sigma_1 = \mathcal{SD}\{name, student\}\big(\text{“}I\ am\ a\ student\text{”}\big)$ and $\sigma_2 = \mathcal{SD}\{name\}\big(\text{“}I\ am\ a\ student\text{”}\big)$. σ_1 is an IRMA signature on the message "I am a student" and it includes the signer's name and student attributes. Here the signature verifier gets a student attribute from the signer that was issued by a trusted issuer and hence, the verifier can trust the issuer for the authenticity of this attribute. σ_2 is an IRMA signature on the message "I am a student" without student attribute. Now the verifier has to trust the signer's statement that she is a student. So, depending on the application use case and the trust model of the verifier, the signature policy can mention if the signer's identity information can be included in the message that is being signed or it must be included as an attribute in the IRMA signature.

3.6.3 Use case scenarios

First we discuss some use cases which can benefit from the flexibility offered by *role-based* IRMA signatures.

1. In the introduction, we briefly mentioned a medical doctor signing a medical statement about a patient using her own medical license number and specialisation attributes. This can be applied to many professionals signing documents in which their competence is a useful part of the signature. More generally, this leads to what may be called *role-based* signatures.

2. With such role-based signatures one can distinguish professional and personal signatures. For instance, a signature of a notary should be different when she is signing the sale document professionally (as a notary) or privately (as a house buyer). IRMA signatures are ideally suited for making such differences explicit, for all verifiers to see.

3. In the healthcare sector, it is important for professionals to get an informed consent from individual patients before accessing and analysing their medical data. The current mechanism to obtain patient consents is as simple as patients ticking a check box that says 'I agree'. But such a mechanism does not prove that the consent was indeed given by the correct patient as it is not verifiable and the patient can easily repudiate giving the consent at a later time. In this case, ABSs can prove beneficial in terms of verifiability and reliability. A healthcare professional can request a patient to provide a signed consent using her attributes (e.g. name, age). If the patient agrees to give the consent then she creates an ABS over the consent message using her attributes. Now the professional can immedaitely verify the patient's ABS and store it for the future reference. As authentic and identifying attributes issued by a trusted issuer are used to sign the consent, the professional can rely on the consent to be authentic, reliable and non-deniable. There are many other situations in which the method used to obtain signed consents from individuals play a very important role from both ethical and legal perspectives. Thus, ABSs are very relevant in such cases.

4. Attribute-based signatures can be used for generating identifiable or non-identifiable signed posts, bulletin board announcements, signed chats. When used with pseudonymous attributes, the ABCs provide authenticity, integrity, non-repudiation as well as enforce blocking of misbehaving users.

Now we describe some use case scenarios that require signer *privacy/anonymity* and how ABSs can be useful in such scenarios.

1. *Anonymous voting.* In large-scale elections, there are usually two main phases: (i) registration and (ii) vote casting. A crucial difference between these two phases is that the first one should be identifying, whereas the second one should not.

   During the registration phase, a potential voter can authenticate to a voting authority in a properly identifying manner, *e.g.* via her citizen registration number. This identity is needed to check if the person at hand is eligible to vote.

   If the check is successful, then the voting authority can blindly issue a random but deterministic 'voting ID' attribute[9] to her IRMA device. During the election phase, this voter can sign her vote with her IRMA device under her 'voting ID' attribute. The (random) voting IDs of all the voters are stored and if any voter tries to vote for the second time, then the voter ID matches with one of the previously stored voter IDs. This second vote can either be discarded or it can replace the first vote based on the voting authority's policy decision. Thus, we can keep the voters anonymous and also avoid double voting by using IRMA signatures. Here we note that a potential voter can use the same IRMA device for authenticating during registration and for signing anonymously during the vote casting phase.

2. *Anonymous petitions* is another application similar to the anonymous voting that can benefit from IRMA signatures.

3. *Whistle blowing.* IRMA signatures can also be used by confidential sources who want to keep themselves unidentified to a journalist or to the public to whom they reveal information. But still they can include some relevant attributes in the signatures on their statements, in order to provide credibility. Whistle blowing includes the cases of reporting domestic abuse or workplace harassment cases without directly identifying themselves. With IRMA signatures, such a reporter can sign a message elaborating the sensitive issue by including relevant but non-identifying attributes (e.g. residing locality, employee). There are many situations where bringing some unethical or illegal issue into the notice of law and justice authorities is important and *not* the identity of the person reporting it.

3.6.4 Estimating the performance of IRMA signatures

Motivated by the study done in this chapter and the huge potential of ABSs, the PbD foundation [37] has implemented the IRMA signatures. In this implementation, IRMA's selective disclosure proofs are extended with domain separation, trusted timestamps and a new interactive user interface for generating and verifying IRMA signatures. Now, IRMA app offers digital signing as another feature along with user authentication for its users.

The current implementation of the IRMA signatures comprises of obtaining a

[9] This is similar to how the user's secret key is blindly issued as an attribute from the issuer during an ABC issuance. See Section 2.3 for details on the blind issuance of ABCs.

trusted timestamp for the signature from the timestamp authority and then generating the signature ($\mathcal{SD}$ proof). To enable user-driven revocation and to secure users' secret keys (bound to ABCs) that are stored on smart phones, the keys are shared between the users' IRMA apps and a central server (called the key-share server) controlled by the PbD foundation (See Section 5.2.1 for details). Due to key sharing, the IRMA app needs to collaborate with the key-share server to create a $\mathcal{SD}$ proof. So the total time to create an IRMA signature includes the round trip times between the IRMA app, key-share server and the timestamp authority which depends on the efficiency of the underlying network in terms of bandwidth and latency. Nevertheless, as one can observe from an IRMA signature demo on the PbD foundation website[10], the signature creation and verification together takes less than 5 seconds. So we conclude that in terms of user experience, IRMA signature generation and verification works efficiently without long user-perceptible delays.

3.7 Concluding Remarks

We have presented the first practical and easily realisable form of attribute-based signatures by building on top of the IRMA technology. IRMA signatures are created by extending the IRMA's selective disclosure proofs that are used for user authentication. They incorporate domain separation that facilitates the secure use of authentication and signature functions on a single IRMA device using the same secret key and, embed trusted timestamps that are bound to the signer's attributes, message and the timestamp authority's signature on the timestamp token. Furthermore, we discuss practical aspects such as revocation, performance and use cases in which IRMA signatures can be readily used.

In conclusion, IRMA signatures offer much greater functionality and flexibility than traditional PKI-based digital signatures in terms of (i) contextual privacy guarantees to the signers, and (ii) ease for verifiers to recognise the role of the signers based on the attributes included in the signature and to verify the authenticity of the attributes. Furthermore, the implementation of these signatures on the IRMA app provides a simple, easy-to-use, efficient digital signing alternative that users can use to create contextual signatures.

[10] `https://privacybydesign.foundation/demo-en/signature/`

Chapter 4

Attribute-based Online Shopping

4.1 Introduction

Nowadays, it seems unavoidable to give away a lot of personal information while carrying out transactions on the Internet. For example, let us consider online shopping. Almost all webshops obligate us to register and log in in order to shop anything online. While it may seem counterintuitive, such transactions could be made privacy friendly; the technology exists to achieve that! In this chapter, we put forward a privacy-friendly scheme for webshopping transactions using attribute-based credentials. We emphasise that such an approach has advantages not only for the purchasers but also for other stakeholders, namely, the webshops, the banks and the delivery companies.

4.1.1 Current webshopping model

There has been a profound change in the way we shop in the last 20 years. In a traditional, brick-and-mortar shopping scenario, one walked into a store, collected items, went to the counter and paid by cash. Increasingly we obtain things online. Current webshopping includes registering personal details and authentication credentials (typically, username and password), placing products in the 'shopping cart', logging in using the authentication method, paying with the involvement of a financial service (such as, bank, PayPal), and finally, initiating product delivery. Figure 4.1 illustrates the steps carried out in the current webshopping schemes. While in the traditional brick-and-mortar scenario, purchasers can remain anonymous and no personally identifiable information (PII) is stored about them, in the online case, PII is often registered by several companies, including the webshop, the bank and the delivery company. Gradually, our shopping activities have become highly traceable and identifiable. Moreover, the relevant information is stored basically forever, with no transparency regarding where it is stored, how it is used and with which parties it is shared.

This huge amount of information places great technical and legal responsibility on the afore-mentioned companies in terms of protecting personal data. Over the time, many of them have become victims of hacking[1]. Under new data protection regulations including the European General Data Protection Regulation (GDPR)[2], the companies risk paying high penalties in the case of data protection violations on their part or failure to report data breaches. And it is not only security problems that arise. Because purchasers give away a lot of personal information, including permanent data (name, address, credit card number, etc.), dynamic data (e.g. purchased items), meta-data (e.g. the bank knows the location and time of purchase) and derived data (combined information, behavioral patterns, etc.), data collection and processing may result in different kinds of privacy threats (exclusion, aggregation, secondary use, etc.) [60]. For the companies that process customers' personal data, strict compliance with data protection regulations becomes inevitable. Regulations such as, the GDPR gives the rights to people to access (see) all the data stored about them and non-compliance involves fines for the companies. Furthermore, the GDPR makes privacy by design and by default mandatory for such companies. This is why both companies and customers have a common interest in countering the security and privacy issues.

4.1.2 Webshopping without disclosing PII

There are privacy-friendly approaches to webshopping. In contrast to the currently dominant fully identifying webshopping paradigm, anonymous online marketplaces such as Silk Road and Agora maintain anonymity for both sellers and purchasers. However, they often become platforms for black markets [61]. So, in this chapter, we consider a significantly new approach for carrying out online shopping transactions, called *attribute-based webshopping*. This approach focusses on achieving purchasers' privacy, while not hiding sellers and products from the public eye. As a result, we strike a balance between the overly-exposing and the overly-hiding paradigms in webshopping.

In this new way of online shopping, a purchaser does not create a personal account at webshops. In fact, for these webshops, the purchaser can remain anonymous. She reveals the minimum information required to complete a shopping transaction to the various participants: a webshop, a bank and a delivery company. The main idea behind our scheme is that the business stakeholders in a shopping transaction learn as little as possible during each interaction with the purchaser. Whenever it is plausible, purchasers are not identified, and no linkable information are revealed about them. This can be achieved by using attribute-based credentials (ABCs). Section 1.1 describes the concept of ABCs and Chapter 2 provides the

[1] E.g. `https://securityintelligence.com/the-top-5-retail-breaches/` [last accessed: August 29, 2018]

[2] More information about GDPR can be found on `http://www.eugdpr.org` [last accessed: July 1, 2018].

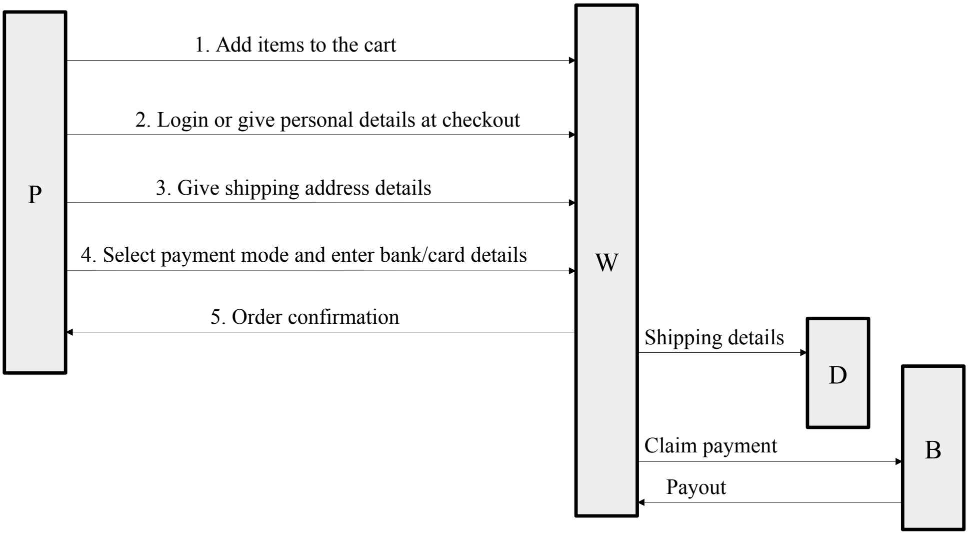

P
W
D
B
1. Add items to the cart
2. Login or give personal details at checkout
3. Give shipping address details
4. Select payment mode and enter bank/card details
5. Order confirmation
Shipping details
Claim payment
Payout

Figure 4.1: Steps carried out between a purchaser P, webshop W, bank B and delivery company D in the current webshopping schemes. We note that the webshop learns the PII, payment and shipping details of a purchaser in such schemes.

necessary cryptographic background information and the protocols for the issuance and verification of ABCs in the Idemix/IRMA system.

We propose to use ABCs for carrying out webshopping transactions. We design most of the information exchanged between the participants within a shopping transaction as attributes and interactions between them in terms of issuing and using ABCs. In particular, a purchaser receives some ABCs in some steps from the webshop and the bank which can be used in other steps within the transaction. During the use of ABCs, a purchaser creates an attribute-based signature (ABS).

In the previous chapter (Chapter 3), we described ABSs and presented a setup for realising ABSs in practice. This setup was specific to an operational ABC-platform IRMA [37] in which ABSs were extended with features such as domain separation and trusted timestamping. The extended ABSs are called *IRMA signatures* and they are intended to function as conventional digital signatures which are verifiable by any verifier at any time after signing. For example, an IRMA signature created by a signer over a property sale document with 'role=notary' attribute is a digital signature that needs to be verified by anybody, even many years after signing. Hence, this signature embeds a trusted timestamp that represents the time of signature creation. Such a timestamp is crucial in situations when the signature creation and its validation take place asynchronously (i.e. in different sessions).

However, in online transactions, including webshopping, there is always an active communication between a user and a verifier. In such situations, ABSs can be used without additional timestamps because the verifier can verify the validity of the signature and the attributes in the same session, similar to authentication.

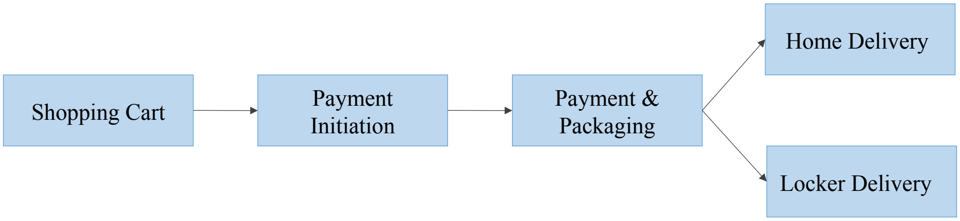


Figure 4.2: Webshopping process

That is why, in our webshopping scheme, we use the basic form of ABS: a non-interactive zero-knowledge proof over a set of selected attributes, a verifier's nonce and a context-specific message. We denote this basic ABS by $\mathcal{ABS}$ in this chapter. Such an ABS is basically an authentication proof (aka selective disclosure proof) but with a context-specific message (e.g. 'Payment signature', 'Delivery acknowledgement') as an additional input. We include such a message because it makes the context of the interaction within the transaction very clear especially to the user (aka purchaser) before signing. In sum, we use ABSs in webshopping transactions because they provide security guarantees of a typical digital signature namely, signer authentication, message integrity, and non-repudiation, while ensuring minimal data disclosure by the purchaser and possible unlinkability. These guarantees are required in the context of online transactions. Nevertheless, one might ask why we cannot use IRMA signatures that include trusted timestamps in the webshopping scheme. We will describe the reason for this in Section 4.3.

The attribute-based shopping scheme retains the main steps: order, payment and delivery, as it also happens in the current webshopping model. See Figure 4.2. However, the steps are more separated; the purchaser controls the initiation of each of them. For instance, the payment step is further divided into payment initiation step (the purchaser initiates payment at the bank) and payment at the webshop (the purchaser provides a payment proof in the form of an attribute-based signature to the webshop before it can proceed to packaging). The technical links between these steps are carefully designed attribute-based credentials.

To give an intuition how our scheme works, we briefly describe the workflow that takes place in an attribute-based shopping transaction. First, the purchaser collects products in her shopping cart, and when she is ready, she closes it. As a result, the purchaser receives the total sum to be paid and a cart identifier as a cart-ABC. Second, the purchaser selects a payment method – possibly independent of the webshop – and contacts the financial service, which we call a bank for simplicity. The actual payment is done with some means, and as a result, the purchaser receives the receipt of payment in the form of a bank-ABC from the bank. Third, the purchaser proves to the webshop that the cart amount has been paid by sending a signature created using the bank-ABC. Then the webshop collects the products according to the cart, packages them and prepares for delivery. Fourth and finally, the purchaser contacts a delivery company of her choice and provides the delivery address. This company collects the purchaser's package from the webshop and dispatches it to the purchaser's address. Alternatively, we describe another option in which a locker facility can be used to hand over the products to purchasers. If the purchaser chooses locker delivery, the webshop sends the purchaser's package to the locker from where she can pick it up. Section 4.3 describes the webshopping scheme using ABCs in detail. Overall, our webshopping scheme follows the same workflow as the current webshopping. Although the internal working of our scheme is different, all the complexity with respect to the underlying cryptography is hidden from the participants involved in the transaction. This ensures that the user experience while shopping online remains unchanged.

The attribute-based webshopping scheme ensures minimal information exchange between the purchaser and each company. The webshop knows only about the purchaser's order details, but it does not know the identity information of the purchaser nor about the payment details. Also, the purchaser's bank learns only how much money has to be debited from the purchaser's account, but it does not learn the identity of the webshop or the purchased items. Lastly, the delivery company learns only the necessary dispatching details (name, address, etc.), and nothing about the purchased items or the payment. (If the locker delivery mode was chosen, then, unlike the delivery company, the locker does not even learn the name and address of the purchaser.)

The main goal of this chapter is to show that achieving basic functionality involved in a typical webshopping transaction: order, payment and delivery, in a secure and data-minimised way is feasible using attribute-based credentials. However, one of the frequent scenarios in a webshopping setting is when a purchaser wishes to return the delivered product back to the webshop in exchange for a replacement or a refund. Although we do not primarily focus on these additional functions in our webshopping scheme, we show in Section 4.4.7 that ABCs can easily support them. However, in online shopping, there are many more aspects such as, dispute resolution, detecting possible fraud, providing reviews about the products or the webshop services, customer relationship management that are out of scope in the proposed webshopping scheme.

In comparison with the widely used, identifying webshopping process, our approach has many benefits for the participants. Most importantly, purchasers do not need to register and authenticate to webshops. This is not only privacy friendly, but also more convenient. Their interaction with companies are more privacy-friendly and they have more control over disclosure and dissemination of their personal data. This technical solution offers advantages not only for the purchasers, but also for companies. Because webshops do not collect and process personal data in our scheme, they do not have to worry about data-protection regulations such as GDPR. As there is no long-term relation between the purchaser and the webshop, our proposal is flexible, which may also help small companies. For instance, a webshop, offering specific products that people purchase only once, can provide a quick service without collecting any superfluous data from its customers. The banks (or financial institutions) can benefit from the new scheme by being able to offer 'shopping-with-privacy' as a new service in the form of anonymous payment credentials. Finally, delivery companies could offer delivery service subscriptions and build a more trusted relationship with their customers by providing prompt delivery and protecting their personal details (e.g. name, address, contact information).

4.2 Preliminaries

The concept of attribute-based credentials is described in the Introduction (Section 1.1), and their issuance and usage (selective disclosure) protocols are described in the Preliminaries (Section 2). For expositional clarity, we explain below the relevant concepts of ABCs along with notations that we will use in this chapter.

4.2.1 Blind issuance

An ABC issuance protocol takes place between a user (actually, a personal device that holds the user's secret key) and an issuer. In this protocol, the user obtains some personalised attributes as a credential from the issuer. See Section 2.3 for a detailed protocol description. The credential issuance takes place such that both the issuer and the user know the values of all the issued attributes (before and after the issuance) except that the issuer does not learn the value of the user's secret key. The user always blinds her secret key (technically, one of her attributes in each of her credentials) from the issuer while the issuer issues other attributes following the user's proof of knowledge of this secret key. However, in some practical cases (e.g. e-voting, e-cash) not only the secret key but other attribute values may also be blinded during issuance. In the attribute-based webshopping scheme proposed in this chapter, we make use of the blind issuance property of ABCs not only for the secret key but also for other attributes. That is, the values of the blinded attributes will only be known to the user and not the issuer at the time of issuance. This ensures that the issuer cannot link the issuance and the use of an attribute-based credential even if it sees the disclosed attributes at the verifier. For example, an issuer I blindly issues a random number r as an attribute to a user U and U discloses r to a verifier V during a selective disclosure protocol. In this case, I cannot recognise the user U even if it learns the value of r from V.

4.2.2 Attribute-based signatures

We briefly recall the main ideas of ABSs for the convenience of the reader. An ABS is in fact a non-interactive zero-knowledge proof over a selected set of attributes and a message. While creating this proof, the signer hashes the commitment to the hidden attributes, a context-specific message, and the verifier's nonce (one-time use random number) to create an unpredictable challenge. The message binds the signature to the specific context and the nonce provides freshness, thereby preventing replay attacks. The nonce also links the signature request by a verifier to the response given by the signer. The meaning of an attribute-based signature conceptually is as follows: "The signer characterised by the disclosed attributes ... signs the message...". The notion of ABSs used in this chapter is similar to the selective disclosure ($\mathcal{SD}$) proofs used for IRMA authentication except that a context-specific message is provided as an additional input in ABSs during the challenge computation. The message explicitly tells the user (aka signer) the context in which the proof is being created. A nonce alone cannot accomplish this. Furthermore, when there is active communication between a signer and a verifier, then the ABS generation takes the form of a challenge-response protocol in which the verifier can immediately verify the validity of the signature on the message and the attributes. These ABSs do not require additional timestamps. We use such ABSs in our webshopping scheme and denote them by $\mathcal{ABS}$. Technically, the main difference between an IRMA signature and an $\mathcal{ABS}$ is that $\mathcal{ABS}$ does not embed a timestamp of signature creation and thus, does not require the timestamp for its verification. In fact, using a unique timestamp within the signature leads to purchaser traceability in the webshopping scheme (See Section 4.3 for details).

ABSs ensure that the minimum data about the signer is revealed to a verifier and the signer can possibly be untraceable based on the signature while providing security guarantees such as, signer authentication based on the disclosed attributes, message integrity, non-repudiation. For example, consider a signer who signs a message “I am paying 10 euros to Fox news website” with a membership ID attribute issued by Fox news. Now Fox news can (i) authenticate the signer based on her revealed membership ID (as the ABS reveals no more information about the signer, it ensures minimum data disclosure); (ii) verify that the signed message has been not been modified since signing; (iii) make sure that the signer who owns the disclosed membership ID cannot deny signing the message. As ABSs are secure and privacy-friendly, we make use of them in the proposed attribute-based webshopping scheme.

4.2.3 Notation for ABCs

ABCs provide many flexible cryptographic operations. We introduce some notation for the operations to make it easier to describe our protocols in the webshopping scheme. Under this scheme, a single shopping transaction will involve several issuances of ABCs and signatures using the issued ABCs. During these operations, some attributes are hidden while some are disclosed from the user's ABCs to an issuer/verifier. To avoid confusion, we consider this notation to be especially useful in this chapter.

- An ABC issuance is an operation carried out interactively by an issuer I and a user. This operation is denoted by

  $$C^I \leftarrow Cred^I(a_1, .., a_k)$$

  where $a_1, .., a_k$ are the attributes in that credential. As a result, the user stores the credential C^I, I's signature over the attributes $a_1, .., a_k$ inside C^I. The $Cred^I(..)$ operation always contains the secret key sk of the user (typically the purchaser in online shopping context). In principle, it is $Cred^I([sk], a_1, .., a_k)$, but we leave it implicit in our notation. In the current context, an ABC with attributes is denoted as

  $$C^I(a_1, a_2, .., a_k)$$

- An attribute-based signature (ABS) is a selective disclosure proof on a message *msg*. It is denoted by

$$\mathcal{ABS}\{C^I(a_1, [a_2])\}(nonce, msg)$$

where the attribute a_1 is disclosed and a_2 is hidden from the verifier. The *nonce* in the above signature is used to add freshness to the proof and it is provided by a verifier. However, when there is no active verifier for a signature, then the nonce input is empty. The message could be a constant data string (like a label specifying the signing context that remains constant for all signers, e.g. *"Payment"*) or a variable data string (e.g. name and address that varies across signers). In the webshopping protocols, when we mention that a purchaser sends an ABS to a verifier (e.g. webshop, delivery company), we mean that the purchaser sends the tuple consisting of the actual signature ($\mathcal{SD}$ proof), disclosed attributes and the message to the verifier.

The notation with regard to attributes within a credential during an issuance and an ABS verification is explained in the Table 4.1.

Table 4.1: Notation used for handling attributes.

Attribute	**Issuance**	**ABS**
a	the value of a is known to both the issuer and the user	the value of a is revealed to the verifier
$[a]$	the value of a is known only to the user and not to the issuer as a result of blind issuance	the value of a is known only to the user, hidden from the verifier

4.3 Attribute-based Webshopping

A typical shopping transaction within our attribute-based webshopping scheme consists of four participants: Purchaser, Webshop, Bank and Delivery entity. The nature of the delivery entity depends on the delivery mode. If the purchaser chooses home delivery mode, then the entity is a delivery company and if it is locker delivery mode, then the entity is a locker. Figure 4.2 shows the main stages in an attribute-based shopping transaction. The shopping cart and the payment initiation stages are common to both the delivery modes. But after the purchaser pays the webshop for her cart, the subsequent actions depend on the chosen delivery mode. Overall, our webshopping scheme follows the same work flow as the current webshopping: order, pay, delivery. This ensures that the user experience while shopping online remains unchanged. However, under the hood, our scheme is different from the current webshopping shown in the Figure 4.1.

Figure 4.3 outlines the steps carried out between the participants in a shopping transaction under our webshopping scheme. First, the Purchaser P anonymously

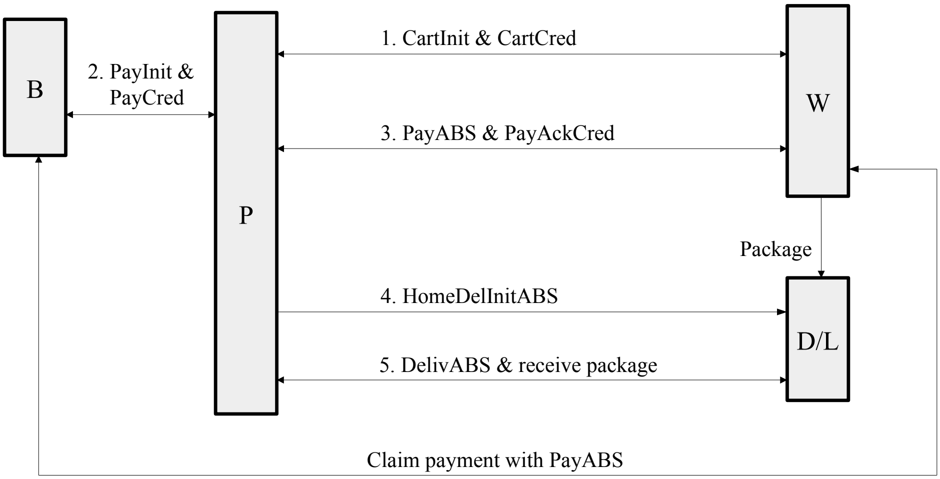


Figure 4.3: Steps carried out between a purchaser P, a webshop W, a bank B and a delivery company D (or a locker L) in our attribute-based webshopping scheme. Here we note that P initiates the main stages: order, payment and delivery and reveals the minimum required information in each stage using ABCs.

initialises a shopping cart and fills it up at the Webshop W (CartInit). When the cart is ready, the shop issues a credential (CartCred). Second, using some out-of-scope payment method (cash, bank transfer, credit card, PayPal, Bitcoin, etc.) and some (hidden) information from the CartCred, the Purchaser receives a credential (PayCred) from the Bank B that certifies that she has paid. Third, the Purchaser goes back to the Webshop, proves that the cart has been paid for at the Bank by providing a payment ABS (PayABS). If this ABS is correct, then the Webshop issues an acknowledgement credential (PayAckCred) to the Purchaser. The attributes in this credential and packaging depends on the mode of delivery (home or locker delivery) that the Purchaser has selected. In the case of home delivery, the Purchaser provides the shipping address to the delivery company D (via HomeDelInitABS). At the time of delivery, the Purchaser proves to the delivery person or to the locker L that she is the cart owner and the Webshop has accepted her payment, using PayAckCred credential (via DelivABS) and takes her delivered package. DelivABS acts as the Purchaser's acknowledgement to the received package. In a separate session, the Webshop presents the PayABS to the Bank and claims its payment for the Purchaser's shopping cart. (Note that the Webshop has not learnt the Purchaser's real identity. As we will see later in the detailed description of the payment phase, PayABS does not identify the Purchaser to the Bank, so the Bank does not learn the relation between the Purchaser and the Webshop.)

Reason for not using ABSs with timestamps. One might wonder why we are using ABSs without timestamps instead of using IRMA signatures which include trusted timestamps. The reason is that these timestamps could lead to trivial linking of a purchaser P and a webshop W by the bank B. In the above summary of our webshopping scheme, we can see that the issuance of the credential PayCred (step 2) is immediately followed by the use of that credential by the purchaser to create PayABS (step 3). If PayABS were an IRMA signature, then a trusted timestamp is embedded in it. This implies that the timestamp is necessary to verify this signature. The main issue here is that when the webshop presents PayABS to the bank while claiming payment for the cart, the bank will learn the timestamp included in the PayABS. The bank can now search for the purchaser who requested for a PayCred just before the time mentioned in that PayABS' timestamp. By doing so, the bank can recognise which of its customers shopped at webshop W around that time. We note that this purchaser-webshop linking is trivial as the bank could rely on the timestamp received as a part of PayABS and does not need to collude with the webshop to trace the purchaser. As we aim to prevent such linking, we avoid using IRMA signatures and use ABSs without timestamps instead.

In the proposed webshopping scheme, we follow the data-minimisation principle while deciding which attributes are necessary to complete each stage of a shopping transaction. The data minimisation principle states that "a data controller should limit the collection of personal information to what is directly relevant and necessary to accomplish a specified purpose."[3]. We make our own policy that defines which data is collected from which party during each step of the transaction. For instance, we make the webshop issue a cart identifier attribute during the shopping phase that allows the webshop to keep track of the purchaser's items and, statuses of the corresponding payment and delivery phases. Here we emphasise that attributes allow the scheme implementors to define and execute their own policies regarding the nature of the attributes, how they are issued (values known or blinded to the issuer at issuance) and disclosed (all, few or none) from the credentials. Thus, ABCs provides flexibility for defining attribute disclosure policies over them to any extent (typically, based on the context). But we strive to achieve the minimal set of attributes to be issued and disclosed in our webshopping scheme. Furthermore, we stress that irrespective of the revealed or unrevealed attributes, ABCs provide a high degree of reliability about the binding of a credential to its holder because of the secret key (that is never revealed) and the cryptographic robustness (see Section 1.1 for security and privacy properties of ABCs). That's why they are especially suitable to achieve security and privacy simultaneously. In the rest of this section, we give the technical details of the scheme.

4.3.1 Assumptions

We now describe the assumptions that we make in our attribute-based webshopping scheme with regard to the participants and the communications that take place

[3] European Data Protection Authority's Glossary: `https://edps.europa.eu/data-protection/data-protection/glossary/d_en` [last accesed: August 30, 2018].

between them over the course of a shopping transaction.

- All communications between a purchaser and other participants happen over an encrypted anonymous channel (e.g. Tor) to prevent identification of the purchaser on the network layer, for instance, based on the IP address. To prevent getting recognised as a returning user by the webshop based on preset cookies, the purchaser can, for instance, use 'New Identity' feature[4] in the Tor browser for every shopping transaction. This will close all the open tabs and windows, clear all private information such as cookies and browsing history, and use new Tor circuits for all connections.
- All ABC issuance and ABS generation instances are implemented on the server and the client sides respectively in a proper way such that the security and privacy properties of ABCs are ensured. For example, the credential issuer's signing key, and the user's secret key associated with her ABCs do not leak.
- A suitable public-key infrastructure is assumed to be in place for the companies involved in the scheme. In particular, the bank and the webshop have certified public-private key-pairs (required for the credential issuance) for the ABC system. For instance, the webshop verifies and accepts the credentials issued by the bank which uses a certified key-pair for the issuance.
- The bank is semi-trusted (aka honest but curious), i.e., it follows the protocol but it wishes to learn as much as possible about the purchaser's shopping (where, when, what).
- The webshop is also semi-trusted; in particular, it honestly follows the protocol but may be curious to find out the purchaser's identity.
- The webshop and the bank do not collude to deanonymise purchasers. This means that they will only share the information needed for the protocol and will not share any extra information (e.g. the time at which a purchaser presented the proof) with an intention to learn the user identities.
- The delivery company, which is responsible for home delivery, is an independent entity (not affiliated with the webshop). The purchaser trusts the delivery company not to collude with the webshop to leak the delivery address or other purchaser-specific information. We also assume that the database of the delivery company that contains the purchaser data (e.g. delivery address) is protected using standard encryption mechanisms.
- In the case of locker delivery, the locker facility may be controlled by the webshop.
- An external adversary does not have full access to the internal states and databases of more than one participant at any point in time.

[4] `https://tb-manual.torproject.org/en-US/managing-identities.html`

4.3.2 Table of notation

Table 4.2 briefly summarises the notation for the webshopping participants and the most important variables and objects.

Table 4.2: Notation used in our webshopping protocol.

Symbol	Interpretation
$\mathcal{P}, \mathcal{W}, \mathcal{L}, \mathcal{B}$	Purchaser, Webshop, Locker, Bank respectively
$\mathcal{W}_{id}, \mathcal{L}_{id}$	Identifiers for $\mathcal{W}$, $\mathcal{L}$ respectively
ID_c	Cart identifier
σ	Total price of all items in the cart
$sk_{\mathcal{P}}$	Secret key of $\mathcal{P}$ associated with its ABCs
$pk_{\mathcal{W}}, sk_{\mathcal{W}}$	Public, private signing keys of $\mathcal{W}$
$pk_{\mathcal{B}}, sk_{\mathcal{B}}$	Public, private signing keys of $\mathcal{B}$
$C_1^{\mathcal{W}}, C_2^{\mathcal{W}}$	Cart, Payment acknowledgement credentials issued by $\mathcal{W}$ to $\mathcal{P}$
$C^{\mathcal{B}}$	Payment credential issued by $\mathcal{B}$ to $\mathcal{P}$
$C_3^{\mathcal{W}}, C_4^{\mathcal{W}}, C_5^{\mathcal{W}}$	Return, Voucher, Refund credentials issued by $\mathcal{W}$ to $\mathcal{P}$
$\mathcal{ABS}$	Attribute-Based Signature

4.4 Scheme Description

In this section, we describe the four main steps of the proposed webshopping scheme. The scheme is depicted on a high-level in Figure 4.3. The first three stages of a shopping transaction in our scheme is described in more technical detail in message sequence diagram 4.4. Then, depending on the mode of delivery chosen by the purchaser, we describe the remaining steps in the transaction in in Section 4.4.4 and Section 4.4.5. The packaging and delivery stages in the case of home delivery and locker delivery are described in message sequence diagrams 4.5 and 4.6 respectively. In these sequence diagrams, each rectangular box represents a secure session between any two participants within a shopping transaction; interactions involving ABCs are denoted by $\longrightarrow$, other interactions by $\dashrightarrow$.

4.4.1 Shopping cart

The shopping cart phase consists of the cart initialisation, addition of items to the cart and finally closing of the cart. The communication between the purchaser $\mathcal{P}$ and the webshop $\mathcal{W}$ that takes place in this phase is described in the following steps and in **1. Cart** box in Figure 4.4.

- `CartInit`: $\mathcal{W}$ assigns a random cart-identifier ID_c to $\mathcal{P}$. ID_c can be viewed as a session-specific pseudonym for $\mathcal{P}$.
- $\mathcal{P}$ browses through the items on $\mathcal{W}$'s website, makes her choice and adds items

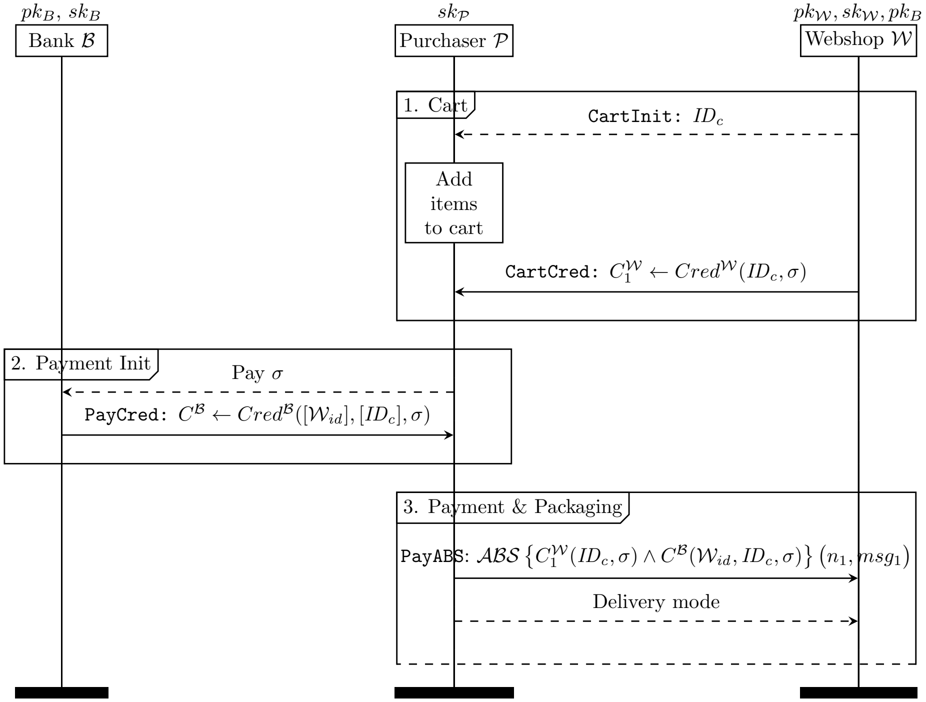

pk_B, sk_B
Bank $\mathcal{B}$
$sk_{\mathcal{P}}$
Purchaser $\mathcal{P}$
$pk_{\mathcal{W}}, sk_{\mathcal{W}}, pk_B$
Webshop $\mathcal{W}$
1. Cart
CartInit: ID_c
Add items to cart
CartCred: $C_1^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(ID_c, \sigma)$
2. Payment Init
Pay σ
PayCred: $C^{\mathcal{B}} \leftarrow Cred^{\mathcal{B}}([\mathcal{W}_{id}], [ID_c], \sigma)$
3. Payment & Packaging
PayABS: $\mathcal{ABS}\left\{C_1^{\mathcal{W}}(ID_c,\sigma) \wedge C^{\mathcal{B}}(\mathcal{W}_{id}, ID_c, \sigma)\right\}(n_1, msg_1)$
Delivery mode

Figure 4.4: Shopping at the webshop

to the cart ID_c. Let the total price of all the added items in the cart be σ.

- `CartCred`: After all the items have been added to the cart, $\mathcal{P}$ closes the cart and then $\mathcal{W}$ issues a cart credential to $\mathcal{P}$:

$$C_1^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(ID_c, \sigma)$$

4.4.2 Payment initiation

Payment initiation is the next phase in the webshopping scheme after the shopping cart has been closed. In this phase, $\mathcal{P}$ pays for the shopping cart at the bank $\mathcal{B}$ (within a separate secure session) and gets a credential from the bank in return. This phase is separately carried out by $\mathcal{P}$, independently of $\mathcal{W}$, to prevent leaking $\mathcal{P}$'s financial information (e.g. bank account number) to $\mathcal{W}$. It can be implemented as a redirect from the webshop to the bank's website. The steps carried out by $\mathcal{P}$ and $\mathcal{B}$ in this phase are given below. Also see **2. Payment Initiation** box in Figure 4.4.

- $\mathcal{P}$ requests $\mathcal{B}$ to debit an amount σ from her account in exchange for a payment-credential. The actual payment method is out of scope in this study; that is, the purchaser can pay in any way that $\mathcal{B}$ accepts, including cash, bank transfer, debit card/credit card payment, PayPal or Bitcoin. Then $\mathcal{P}$ blinds the values of $\mathcal{W}$'s identity $\mathcal{W}_{id}$ (e.g. uniquely identifying name string or an integer) and $\mathcal{W}$-assigned cart identifier ID_c and sends them to $\mathcal{B}$.

- `PayCred`: $\mathcal{B}$ processes $\mathcal{P}$'s request, debits σ from $\mathcal{P}$'s account and issues a payment credential to $\mathcal{P}$:

  $$C^{\mathcal{B}} \leftarrow Cred^{\mathcal{B}}([\mathcal{W}_{id}], [ID_c], \sigma)$$

  in which the only attribute value that is known to $\mathcal{B}$ is the amount σ. Blinding the attributes $\mathcal{W}_{id}$ and ID_c from $\mathcal{B}$ during the issuance of `PayCred` aims to prevent $\mathcal{B}$ from linking the issuance of a payment credential from its usage. However, we note that $\mathcal{B}$ can still use the amount σ to link the credential issuance and usage but this is possible only when the amount σ is very specific (e.g. 49.58 euros). This can be prevented by for instance, rounding up σ to the nearest multiple of 5 or 10 (e.g. 50 euros). By doing so, we increase the anonymity set of σ i.e., the number of purchasers who have purchased something for σ amount from any of the webshops[5] that are using this webshopping scheme.

The payment credential `PayCred` is bound to the purchaser $\mathcal{P}$'s secret key, her shopping cart and the webshop's identifier. That is why this credential is non-transferable and specific to a shopping transaction. In other words, this credential can only be used by to pay for a specific shopping cart at the webshop. Furthermore, it is similar to a 'dinner cheque' or 'present cheque' that one can spend but for which can never get back the money.

4.4.3 Payment and packaging

In this phase, the purchaser $\mathcal{P}$ creates and sends an attribute-based signature – `PayABS` to $\mathcal{W}$. This ABS proves that she is the owner of the cart ID_c and she has got the payment credential from the bank to pay $\mathcal{W}$ for the cart. The webshop can present the `PayABS` to the bank while claiming the cart amount. `PayABS` is the first message exchanged between $\mathcal{P}$ and $\mathcal{W}$ in the session denoted by **3. Payment and Packaging** box in Figure 4.4. This signature is computed as

$$\mathcal{ABS}\left\{C_1^{\mathcal{W}}(ID_c, \sigma) \wedge C^{\mathcal{B}}(\mathcal{W}_{id}, ID_c, \sigma)\right\}\left(n_1, msg_1\right)$$

where $\mathcal{P}$ discloses all the attributes from both $C_1^{\mathcal{W}}$ and $C^{\mathcal{B}}$ credentials. In addition to the attributes, the signature takes $\mathcal{W}$'s nonce n_1 and $msg_1 =$ *"PayABS"* as inputs. The nonce adds freshness to the ABS i.e. $\mathcal{W}$ can verify locally that this ABS is bound to the current payment session. The message string *"PayABS"* is fixed for this stage in the shopping transaction and it is meant to add context to the ABS.

[5] As we hide the webshop's identifier $\mathcal{W}_{id}$ from the bank during `PayCred` issuance, the anonymity set is not limited to purchasers shopping at that particular webshop but spans across the purchasers shopping at all the webshops that use the webshopping scheme.

This ABS provides the following guarantees to the webshop: (i) the signer possesses the disclosed attributes; (ii) the signature is bound to the current payment context (via the signed message) and to the cart identifier (via the disclosed attribute), so it cannot be used for payment anywhere else; (iii) the signer cannot deny paying the webshop the amount σ after signing.

If the ABS verifies correctly, then $\mathcal{W}$ asks $\mathcal{P}$ to choose the delivery mode and issues a payment-acknowledgement credential to $\mathcal{P}$. The contents of the `PayAckCred` credential and packaging of the cart items depend on the chosen delivery mode for the cart items to $\mathcal{P}$. If it is home delivery mode, then the delivery company $\mathcal{D}$ is contacted. The communication between $\mathcal{P}$, $\mathcal{D}$ and $\mathcal{W}$ in the delivery phase of the shopping transaction is described in Section 4.4.4. If the chosen mode is locker delivery, then the cart items are packaged and sent to the locker $\mathcal{L}$ from where $\mathcal{P}$ can pick it up. The communication between $\mathcal{P}$, $\mathcal{L}$ and $\mathcal{W}$ are described in Section 4.4.5.

4.4.4 Home delivery

Home delivery is a common mode of delivery for the items shopped on the web as it is convenient and does not require any travel from purchasers. This mode requires a purchaser to share her personal details such as name, address and phone number with a delivery company. If a purchaser trusts a delivery company to handle most of the deliveries from her shopping transactions, then it seems reasonable for her to subscribe for a delivery service at that company, share the above details with it and pay the company directly[6]. It is better from a data protection aspect than sharing the information with multiple webshops which provide their own delivery service or entrust delivery to other companies without the knowledge of the purchaser. The latter case leads to the purchaser's personal data ending up in several companies' databases and thus increases the data loss, misuse and privacy risks. That is why, we consider a scenario in which the purchaser $\mathcal{P}$ chooses her trusted delivery company $\mathcal{D}$ before the webshop $\mathcal{W}$ proceeds to packaging her cart items. Subsequently, $\mathcal{P}$ contacts $\mathcal{D}$, independently of $\mathcal{W}$, and informs $\mathcal{D}$ about her package at $\mathcal{W}$ and provides the delivery address. Below we describe the communication between $\mathcal{P}$ and $\mathcal{W}$ that follows a successful verification of `PayABS` in the same secure session. See Figure 4.4.4.

- $\mathcal{P}$ sends the identity $\mathcal{D}_{id}$ (e.g. uniquely identifying string or an integer) of her trusted delivery company to $\mathcal{W}$.
- `PayAckCred`: $\mathcal{W}$ assigns the cart identifier ID_c as the package identifer and issues a second credential – `PayAckCred` to $\mathcal{P}$ as follows.

$$C_2^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(\mathcal{D}_{id}, ID_c)$$

  This credential indicates the payment acknowledgement by $\mathcal{W}$. ID_c attribute binds the shopping cart and the package.

[6] The pricing model could one-time, fixed price (per billing cycle) or usage-based, depending on what the delivery company offers. The payment method used to pay the delivery company remains out of scope in our webshopping scheme.

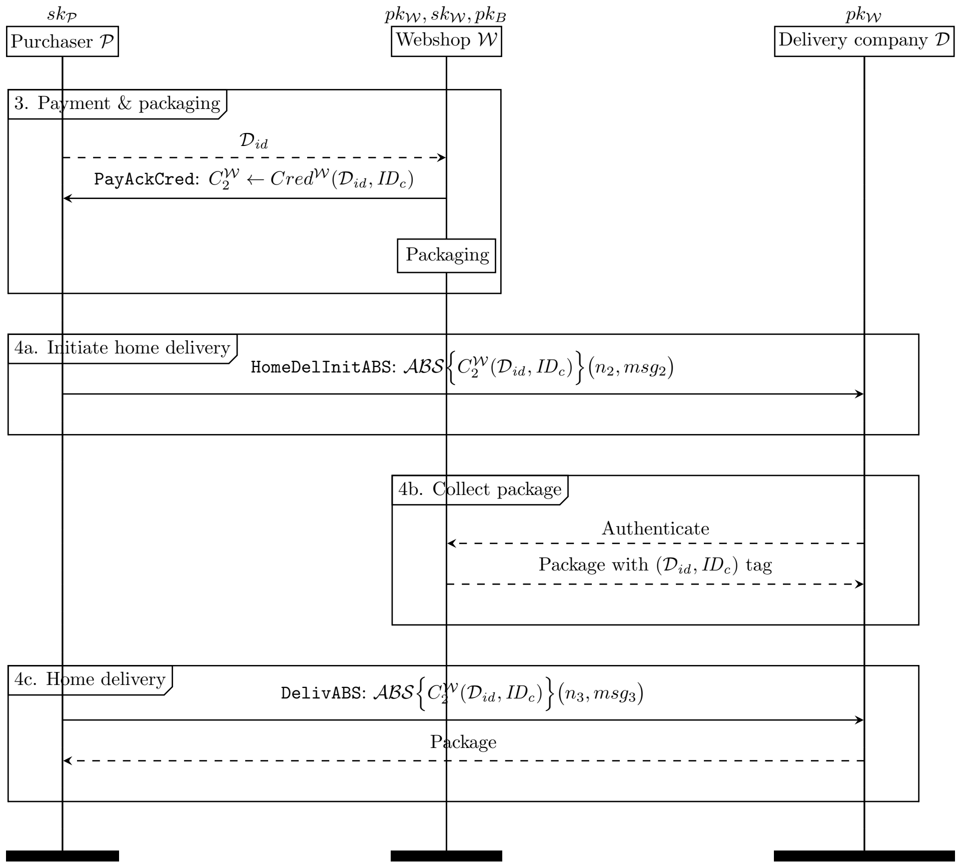

$sk_{\mathcal{P}}$
Purchaser $\mathcal{P}$
$pk_{\mathcal{W}}, sk_{\mathcal{W}}, pk_B$
Webshop $\mathcal{W}$
$pk_{\mathcal{W}}$
Delivery company $\mathcal{D}$
3. Payment & packaging
$\mathcal{D}_{id}$
PayAckCred: $C_2^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(\mathcal{D}_{id}, ID_c)$
Packaging
4a. Initiate home delivery
HomeDelInitABS: $\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{D}_{id}, ID_c)\Big\}(n_2, msg_2)$
4b. Collect package
Authenticate
Package with $(\mathcal{D}_{id}, ID_c)$ tag
4c. Home delivery
DelivABS: $\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{D}_{id}, ID_c)\Big\}(n_3, msg_3)$
Package

Figure 4.5: Home delivery mode

- $\mathcal{W}$ tags the package containing all the items in $\mathcal{P}$'s shopping cart with $\mathcal{D}_{id}$ and ID_c values for $\mathcal{D}$'s reference using, for instance, a QR code.

Finally, $\mathcal{P}$ is redirected to $\mathcal{D}$'s webpage from the webshop where she can provide her name, address and the package identifier. If $\mathcal{P}$ holds a subscription for the delivery service at $\mathcal{D}$ then, she authenticates to $\mathcal{D}$. This authentication method could be for instance, disclosing the subscription ID attribute from an ABC issued by $\mathcal{D}$ or via a username-password; however, this is independant of the webshopping scheme. The communication between $\mathcal{P}$ and $\mathcal{D}$ takes place as follows. Also see **4a. Initiate home delivery** box in the Figure 4.5.

- $\mathcal{P}$ informs $\mathcal{D}$ that $\mathcal{D}$ needs to collect her package from $\mathcal{W}$ and deliver it to her address. However, $\mathcal{D}$ should be first convinced that the delivery request and the address is coming from the same purchaser who has ordered this package at the webshop $\mathcal{W}$. To prove this to $\mathcal{D}$, $\mathcal{P}$ creates the following signature, which we call `HomeDelInitABS`.

$$\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{D}_{id}, ID_c)\Big\}(n_2, msg_2)$$

In the above ABS, $\mathcal{P}$ discloses $\mathcal{D}_{id}$ and ID_c from the `PayAckCred` credential. The signature is on $\mathcal{D}$'s nonce n_2 and msg_2 that contains $\mathcal{P}$'s delivery address[7]. This ABS proves that $\mathcal{P}$ is the legitimate purchaser who has paid for the package at $\mathcal{W}$, binds her delivery address to the proof, preserves the integrity of address and also prevents her from denying later that she initiated the delivery of the cart items at $\mathcal{D}$.

- $\mathcal{D}$ verifies the ABS, stores the attribute ID_c and $\mathcal{P}$'s delivery address. We trust $\mathcal{D}$ not to collude with $\mathcal{W}$ and share the purchaser's information with it. This assumption is necessary to preserve the purchaser's anonymity towards the webshop. Furthermore, all the purchaser-related information are encrypted and the decryption key is stored securely (e.g. in a hardware secure module) by $\mathcal{D}$. This ensures that $\mathcal{W}$ or any other adversary cannot learn the contents even in case of a database breach.

With this, the online shopping transaction between $\mathcal{P}$ and $\mathcal{W}$ (also including $\mathcal{B}$ for payment and $\mathcal{D}$ for home delivery) is completed.

Now the delivery company needs to track the purchaser's package at the webshop to deliver it to the purchaser. See the session **4b. Collect package** in Figure 4.4.4; this is a physical encounter. After $\mathcal{D}$ has received the purchaser's delivery request, it authenticates to $\mathcal{W}$ by using some means (e.g. prove its identifier $\mathcal{D}_{id}$ attribute with a selective disclosure proof, a challenge-response authentication with its public-private key-pair) and collects $\mathcal{P}$'s package that has $(\mathcal{D}_{id}, ID_c)$ tag. If there are more packages of other purchasers at $\mathcal{W}$ that are tagged with $\mathcal{D}_{id}$, then $\mathcal{D}$ collects all of them. Then, it maps the cart identifier ID_c on each package to the ID_c attribute and the delivery address provided by the purchaser $\mathcal{P}$ in her delivery request. Finally, $\mathcal{D}$ sends the package with one of its delivery personnel to $\mathcal{P}$'s address.

At the time of delivery, the interaction between the delivery person and the purchaser is shown in **4c. Home delivery** box in Figure 4.4.4. The delivery person hands over the package to $\mathcal{P}$ and asks her to sign with the cart identifier attribute from `PayAckCred`. Then $\mathcal{P}$ creates a fresh ABS, which we call a `DelivABS`:

$$\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{D}_{id}, ID_c)\Big\}(n_3, msg_3)$$

where the attributes $\mathcal{D}_{id}$ and ID_c are disclosed from `PayAckCred`. It is $\mathcal{P}$'s signature on $\mathcal{D}$'s nonce n_3 and $msg_3 =$ *"Delivery Acknowledgement"*. The delivery person can verify the `DelivABS` using a handheld terminal that is capable of verifying attribute-based signatures. Its successful verification guarantees that $\mathcal{P}$ holds the correct cart identifier attribute that is bound to $\mathcal{D}$'s identifier $\mathcal{D}_{id}$ and she cannot deny having received the package at a later point in time.

[7] If $\mathcal{P}$ already has address attribute as a part of an ABC issued by some authorised issuer (e.g. municipality), then $\mathcal{P}$ can sign using this attribute instead of including it in the message. By this, $\mathcal{D}$ can verify the authenticity of the address attribute as well.

Delivery to a different address. If the purchaser wishes to order something from the webshop and get it delivered at somebody's address (different from her own address), it can be done in our webshopping scheme as follows. While initiating home delivery, $\mathcal{P}$ creates the `HomeDelInitABS` by signing the name and address of the person (e.g. a friend) to whom the package must be delivered. Then she sends this ABS to her friend to authorise the friend to receive the package. At the time of delivery, $\mathcal{P}$ friend can show the `HomeDelInitABS` along with her own signature `DelivABS`:

$$\mathcal{ABS}\Big\{C^I(name, address)\Big\}(n_x, msg_x)$$

to prove to the delivery person that she is indeed the person for whom $\mathcal{P}$ ordered the package. In the `DelivABS`, $\mathcal{P}$'s friend signs a message for instance *"Delivery Acknowledgement"* with her name and address attributes issued by some issuer I. The delivery person from $\mathcal{D}$ verifies both the signatures: (i) if they are valid, and (ii) if the name and address given in the message of `HomeDelInitABS` are same as the attributes included in the `DelivABS`. A successful verification of both ABSs guarantees $\mathcal{D}$ that the package has been delivered to the right person and the receiver of the package cannot deny having received it at a later point in time.

Our webshopping scheme supports the home delivery mode as it is convenient and very common among the webshoppers. However, this mode of delivery has the following limitation. The purchaser places full trust in the delivery company $\mathcal{D}$ to protect her personal information and not to share the information with any third party. Without this trust assumption, it is not possible to ensure purchasers' privacy in a webshopping transaction involving home delivery. The webshop is also trusted not to share the details of cart items with the delivery company.

4.4.5 Locker delivery

In contrast to home delivery, locker delivery mode allows a purchaser to choose a location of a locker (or a cabinet) from which she can pickup her package. This type of self-service parcel delivery is offered as a service by many webshops nowadays, for instance, Amazon[8] and DHL Locker[9]. This delivery mode obviates the need for the disclosure of the purchaser's personal details such as name, address and phone number to anybody during a shopping transaction. The locker mode is technically different from the home delivery mode as a purchaser does not share her personal details with the locker and hence there is no need for her to trust the locker. Even when the locker is directly controlled by the webshop, the webshop does not find out the identity of the purchaser. Although this mode of delivery requires some work on the purchaser's part in traveling to the locker location and picking up the package by herself, it provides more privacy to the purchaser than the home delivery mode. That is why we consider locker as a second mode of delivery in our attribute-based webshopping scheme. The purchaser has the freedom to choose either of the two modes in our scheme.

Now we describe how the webshop and the purchaser continue communicating

[8] `https://en.wikipedia.org/wiki/Amazon_Locker` [last accessed: June 15, 2018]

[9] `https://www.dhlparcel.nl/nl/dhl-locker` [last accessed: June 15, 2018]

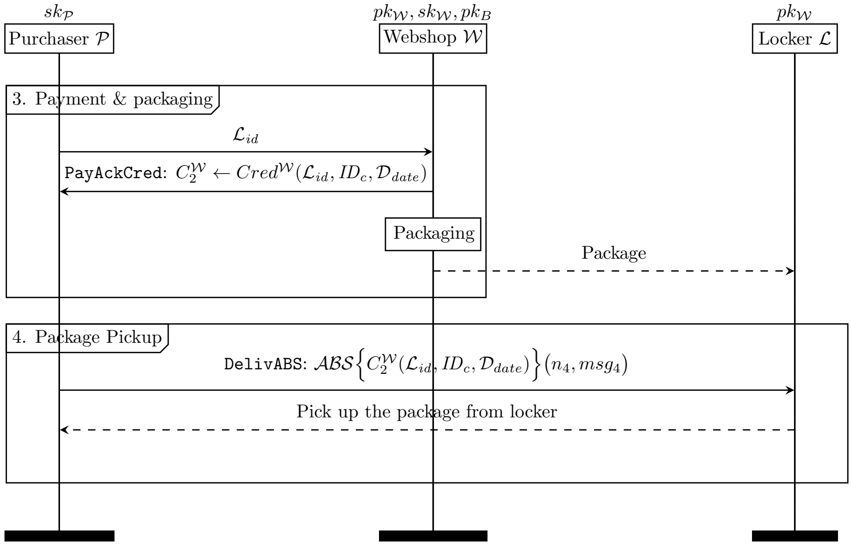

$sk_{\mathcal{P}}$
Purchaser $\mathcal{P}$
$pk_{\mathcal{W}}, sk_{\mathcal{W}}, pk_{B}$
Webshop $\mathcal{W}$
$pk_{\mathcal{W}}$
Locker $\mathcal{L}$
3. Payment & packaging
$\mathcal{L}_{id}$
PayAckCred: $C_2^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, \mathcal{D}_{date})$
Packaging
Package
4. Package Pickup
DelivABS: $\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, \mathcal{D}_{date})\Big\}(n_4, msg_4)$
Pick up the package from locker

Figure 4.6: Delivery pickup from a locker

in the same secure session after a successful verification of the purchaser's PayABS. See Figure 4.6.

- $\mathcal{P}$ sends her choice of locker location $\mathcal{L}_{id}$ to $\mathcal{W}$.
- PayAckCred: To indicate that it acknowledges the purchaser's PayABS, $\mathcal{W}$ issues a second credential to $\mathcal{P}$:

  $$C_2^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, \mathcal{D}_{date})$$

  where $\mathcal{L}_{id}$ attribute denotes the identifier of the locker to which $\mathcal{P}$'s package will be delivered, ID_c attribute is the cart identifier that binds the shopping cart and the package and $\mathcal{D}_{date}$ is the delivery date on or after which $\mathcal{P}$ can pickup her package from the locker.
- Then $\mathcal{W}$ transports $\mathcal{P}$'s package to the locker facility $\mathcal{L}_{id}$.

To pickup the package from the assigned locker, $\mathcal{P}$ has to first show a valid cart identifier and prove to the locker $\mathcal{L}$ that she is the cart owner and $\mathcal{W}$ has acknowledged her payment via PayABS. For this purpose, we make the purchaser sign a context-specific message with her attributes from PayAckCred as follows:

$$\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, \mathcal{D}_{date})\Big\}(n_4, msg_4)$$

This ABS is called a DelivABS and it discloses all the attributes from $C_2^{\mathcal{W}}$ to $\mathcal{L}$. It is $\mathcal{P}$'s signature on $\mathcal{L}$'s nonce n_4 and msg_4 = *"Delivery Acknowledgement"*. The nonce n_4 and message msg_4 adds freshness and context to the ABS respectively. If DelivABS is verified successfully, $\mathcal{L}$ opens the locker door and allows the purchaser to take her package. We use ABSs because they provide signer authentication by proving the validity of the attributes and also the non-repudiation guarantee, that is, a purchaser cannot deny later having authenticated at the locker and picking up her package.

If $\mathcal{P}$ is dissatisfied with the product at a later time, she can return the product in exchange for a replacement, a voucher or cash according to the procedure described in Section 4.4.7, within a return period stipulated by the webshop, say, two weeks from the delivery date.

4.4.6 Payment collection

Payment collection is performed by the webshop at any time after the completion of a shopping transaction with the purchaser. Although this step is not a part of the shopping transaction, it is important to know how the webshop gets the money from the bank that was reserved by the purchaser for the shopping cart.

To redeem the payment for the shopping cart, $\mathcal{W}$ approaches $\mathcal{B}$ with the `PayABS` that was provided by $\mathcal{P}$ in the payment phase. $\mathcal{B}$ maintains a double-spend database that logs all the previous transaction identifiers corresponding to the payment collection claims. The interaction between $\mathcal{W}$ and $\mathcal{B}$ is detailed in the following steps.

- $\mathcal{W}$ authenticates to $\mathcal{B}$ in some form (e.g. logging in with its bank-credentials) independent of the current scheme and then presents $\mathcal{P}$'s `PayABS` to $\mathcal{B}$. Note that $\mathcal{B}$ cannot link an ID_c to the payment initiation phase as $\mathcal{B}$ had blindly issued ID_c attribute to $\mathcal{P}$ and had not known its value then.
- $\mathcal{B}$ checks if
  - $\mathcal{W}$'s claimed identity during authentication and the payee's identity attribute $\mathcal{W}_{id}$ are the same;
  - transaction identifier ID_c (disclosed from `PayABS)`) is not present in its double-spend database, and
  - `PayABS` verifies correctly i.e., it is a valid signature involving $C^{\mathcal{B}}$ on the message "*PayABS*" (and nonce n_1).

  If all the three checks are successful, $\mathcal{B}$ gives or transfers the money worth σ amount to $\mathcal{W}$.
- $\mathcal{B}$ stores the ID_c attribute in its double-spend database so that it can verify if the same ABS is presented to it for the second time.

Now we elaborate on the payment initiation and collection phases from the bank's perspective. It is an invariant in our scheme that the number of outstanding payments at the bank is always equal to the number of payment collection claims by the potential payees (i.e. webshops). The bank $\mathcal{B}$ maintains a pool (multiset) of outstanding payments following the issuance of payment credentials to its customers. During a payment collection, $\mathcal{B}$ sees the disclosed attributes – the payee's identifier $\mathcal{W}_{id}$ and the transaction's identifier ID_c – belonging to a payment claim i.e., a `PayABS` for the first time. As the bank does not know these identifiers earlier to the submission of the payee's claim, it cannot link the payment initiation and collection stages which is equivalent to saying that $\mathcal{B}$ cannot link the issuance and showing of the payment-credential (ABCs' issuer unlinkability feature – See Section 1.1 for details). Let us consider an example in which the bank's pool has ten outstanding payments of 50 euros each. When a payee claims the recovery of 50 euros and provides a valid `PayABS`, the bank reimburses the payee with any one of the ten outstanding payments from its pool. The bank does not know which of its customer's 50-euro payment is going to that particular payee.

4.4.7 Return-replacement-refund scenario

One of the frequent scenarios in a webshopping setting is when a purchaser wishes to return a delivered product to the webshop in return for a replacement or a refund. Although we primarily focus on the order, payment and delivery functions in a shopping transaction, we show that ABCs can easily support additional functions such as product returns. In this section, we describe a simple case for the return of the products picked up from a locker. Such products can be returned to the webshop via the same or a different locker. They could either be replaced or refunded with cash or a gift voucher to the purchaser with the procedure below. We describe this procedure mainly to demonstrate that such scenarios can also be handled with attributes in our shopping scheme. The communication between $\mathcal{P}$ and $\mathcal{W}$ during the procedure is described in the following steps.

1. $\mathcal{P}$ visits the returns section of $\mathcal{W}$'s website, submits a return request with the following signature using the credential `PayAckCred`:

   $$\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, \mathcal{D}_{date})\Big\}(n_5, msg_5)$$

   This is $\mathcal{P}$'s ABS on $\mathcal{W}$'s nonce n_5 and $msg_5 =$ *"Return Request"*. It proves to $\mathcal{W}$ that $\mathcal{P}$ has paid for a shopping cart ID_c and has received the package at locker location $\mathcal{L}_{id}$. $\mathcal{W}$ can also track the `PayABS` corresponding to ID_c.

2. If the ABS verifies successfully and if $\mathcal{W}$ approves $\mathcal{P}$'s request for returning the delivered product, then it issues a return-credential to $\mathcal{P}$:

   $$C_3^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, R_{date}, opt)$$

   where $\mathcal{L}_{id}$ is the locker's identifier (same locker as in the first package-pickup step unless $\mathcal{P}$ explictly mentions a different locker in her return request), ID_c is the original shopping cart identifier, R_{date} is the date of return and *opt* is the option attribute whose value could be either 'replacement', 'exchange for a voucher' or 'cash refund'. Then $\mathcal{W}$ notifies $\mathcal{P}$ to go to the locker facility and deposit the package to be returned.

3. $\mathcal{P}$ shows the return-credential to the locker $\mathcal{L}$ with the ABS

   $$\mathcal{ABS}\Big\{C_3^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, R_{date}, opt)\Big\}(n_6, msg_6)$$

   and deposits the package in the locker. This is $\mathcal{P}$'s signature on the locker's nonce n_6 and $msg_6 =$ *"Return Product Deposit"* which proves that $\mathcal{P}$ has deposited the product to be returned at the locker. If the signature verifies correctly then the locker accepts the product and sends it to the webshop.

4. $\mathcal{W}$ retrieves the package, checks the returned product's condition or its defect (in case a defective item is returned) and then performs one of the following procedures based on the value of the *opt* attribute:

   (a) replaces the returned product and sends the replacement to the locker. $\mathcal{P}$ can pick it up at the locker after proving the attributes from $C_3^{\mathcal{W}}$ credential with an ABS:

   $$\mathcal{ABS}\Big\{C_3^{\mathcal{W}}(\mathcal{L}_{id}, ID_c, R_{date}, opt = \text{“}replacement\text{”})\Big\}\big(n_7, msg_7\big)$$

   where n_7 is a nonce provided by the locker and $msg_7 =$ *“Replaced Product”*.

   (b) issues a voucher-credential:

   $$C_4^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(\sigma, V_{id}, V_{val}, opt = \text{“}voucher\text{”})$$

   where σ is the voucher's worth, V_{id} is the voucher identifier and V_{val} is the voucher validity. V_{id} attribute is randomly chosen by $\mathcal{P}$ and blindly issued to $\mathcal{P}$ by $\mathcal{W}$. It is included to prevent the double spending of the voucher by $\mathcal{P}$ at $\mathcal{W}$. $\mathcal{P}$ can use this voucher-credential to purchase some other item worth σ amount at $\mathcal{W}$ within the voucher's validity period by creating `VoucherPayABS` similar to `PayABS`. The only difference is that, in the payment-by-voucher scenario, the bank $\mathcal{B}$ is not involved; $\mathcal{W}$ checks if the presented voucher identifier is present in its voucher-double-spend database and if not, $\mathcal{W}$ accepts the payment in the form of a voucher. $\mathcal{W}$ cannot link a voucher's issuance and its use by a specific purchaser.

   (c) issues a cash-refund credential:

   $$C_5^{\mathcal{W}} \leftarrow Cred^{\mathcal{W}}(\sigma, ID_c, opt = \text{“}refund\text{”}, \mathcal{W}_{id}, \mathcal{W}_{ac})$$

   using which $\mathcal{P}$ can prove to the bank $\mathcal{B}$ that she has rightfully received the cash refund from the webshop $\mathcal{W}$. The bank transfers σ amount from the $\mathcal{W}$'s account $\mathcal{W}_{ac}$ to the purchaser's account.

Options (a) and (b) maintain the anonymity of the purchaser towards the webshop and the bank but option (c) reveals to the bank that the purchaser had previously bought something worth σ amount at the webshop.

4.5 Protocol Analysis

The proposed attribute-based shopping scheme aims to provide (i) privacy to purchasers in terms of anonymity towards the webshop and unlinkability of their actions at the webshop and the bank and, (ii) security in terms of authenticity and integrity of transaction data that includes the attributes revealed by the purchaser, confidentiality of transaction on a network level (achieved by the network channel), non-replay of messages by the purchaser and non-repudiation by the purchaser.

Under the assumptions stated in Section 4.3.1, we analyse our attribute-based webshopping scheme (shopping cart to payment collection stages), list the possible attack scenarios and countermeasures in this section. We leave out the product return, replacement and refund protocols in the analysis as they are not the primary focus of the chapter.

4.5.1 Privacy attacks

Due to the use of a secure, encrypted and anonymous channel, an external adversary cannot gather much information about the purchaser $\mathcal{P}$'s shopping session nor can it see any data exchanged between $\mathcal{P}$ and the webshop $\mathcal{W}$. The potential attacks against $\mathcal{P}$'s privacy and the ways in which they are countered in our scheme are described below.

1. Bank $\mathcal{B}$ learns the relation between a purchaser $\mathcal{P}$ and the webshop $\mathcal{W}$ by linking `PayABS` and `PayCred`. In the proposed webshopping scheme, there are three ways in which $\mathcal{B}$ can perform this linking:

   - Linking based on ID_c
     Countermeasure: $\mathcal{B}$ blindly issues the attributes $\mathcal{W}_{id}$ and ID_c in the payment-credential. It means that $\mathcal{B}$ does not know the values of these two attributes at the time of `PayCred` issuance. When $\mathcal{B}$ gets `PayABS` from $\mathcal{W}$, it sees these values for the first time. So, $\mathcal{B}$ cannot link the `PayABS` to a specific $\mathcal{P}$'s `PayCred` based on the values of ID_c and $\mathcal{W}_{id}$. If these values were not blinded, then the linking is trivial i.e. $\mathcal{B}$ does not need to collude with $\mathcal{W}$ to find out this link.

   - Linking based on the cart amount σ
     Countermeasure: It is difficult to carry out this linking when the anonymity set for the total cart amount is considerable. One way to increase the anonymity set for the amount is by rounding up the cart prices to their next multiple of 5 or 10. In this case many purchasers would receive payment credentials of similar amount and it will become hard for $\mathcal{B}$ to correlate a `PayABS` with a specific $\mathcal{P}$ purely based on σ. In the client side implementation, purchasers could be given two choices: whether to accept the total cart price as it is or to anonymise the cart price by rounding it up.

   - Linking by time correlation
     Let us call the time at which $\mathcal{B}$ issues `PayCred` to $\mathcal{P}$ as t_i and the time at which $\mathcal{P}$ sends the `PayABS` to $\mathcal{W}$ as t_p. $\mathcal{B}$ knows t_i and if it comes to know t_p, then it can correlate these times and identify $\mathcal{P}$. As mentioned in Section 4.3, using ABSs with trusted timestamps (IRMA signatures) lead to trivial linking between a purchaser and her ABS due to the timestamp embedded in the signature. This is because the timestamp is almost same as t_p (assuming that the purchaser sends the ABS to the webshop immediately after creating it).

     However, even when ABSs without the embedded timestamps are used in the webshopping scheme, linking is possible if the webshop and the bank collude. Then $\mathcal{W}$ would willingly tell the bank the time t_p and $\mathcal{B}$ can find the payment credentials whose issuing times are close to the time t_p. $\mathcal{B}$ might use the cart amount σ to further narrow down its choices and succeed in tracing the shopping transaction to a specific purchaser $\mathcal{P}$. As this seems unavoidable in the current scheme, we make a non-collusion assumption about the webshop and the bank in the scheme. In practice, this assumption is quite reasonable considering that the number of webshops is very large when compared to the number of banks. It is uncommon that a bank would collude with every webshop to trace purchasers.

There is another way a bank can deduce the link between a `PayABS` and a `PayCred` even when $\mathcal{W}$ is not colluding with it. This is when $\mathcal{W}$ always approaches $\mathcal{B}$ for payment collection as soon as it receives `PayABS` from $\mathcal{P}$. Then even without $\mathcal{W}$ explicitly telling t_p to $\mathcal{B}$, $\mathcal{B}$ can still guess t_p and perform linking t_p and t_i to identify $\mathcal{P}$. To prevent this, `PayCred` issuance and payment collection actions must be decoupled by introducing a suitable delay between them. One way to do this is for the webshop to aggregate all the payment ABSs received by purchasers over a period, say, a week, and then to present all of them to the bank at the end of the period. This measure minimises the network cost for the webshop as it can claim payments for multiple purchasers' shopping carts at once at the bank. It also breaks the time correlation between the payment-credential issuance to the purchaser and payment collection by the webshop at the bank, thus preventing the bank from tracing purchasers based on timing. This measure should be enforced independently of the attribute-based webshopping scheme.

2. Based on the transcripts of two shopping transactions, $\mathcal{W}$ attempts to trace a purchaser or find out if the same purchaser was involved in both the transactions.
   Countermeasure: Linking of two transactions to a particular $\mathcal{P}$ by $\mathcal{W}$ is not possible due to the multi-show unlinkability feature of ABCs (See Section 1.1 for details).

3. In the case of home delivery of products, the delivery company $\mathcal{D}$ learns the cart identifiers and some personal data (e.g. name, delivery address, phone number) of the purchaser. It could collude with the webshop and reveal the purchaser's personal data to the webshop.
   Countermeasure: In our webshopping scheme, we trust $\mathcal{D}$ not to collude with $\mathcal{W}$ and divulge purchasers' information. If this assumption cannot be made, then privacy of the purchasers cannot be guaranteed. As an alternative, locker delivery mode could be used as it preserves the privacy of purchasers without the above trust assumption. However, we note that a security camera with face recognition placed at the locker location subverts the privacy provided by this delivery mode.

4. $\mathcal{D}$'s database gets breached by an external adversary who makes all the purchaser- and delivery-related information public. This attack has two consequences: (1) The personal data of purchasers consisting of name, contact details, address along with their shopping cart identifiers are exposed. This also exposes the

relationship between a purchaser and the delivery company to the public; (2) The webshop can use the cart identifiers for linking purchasers' personal information to their shopping transactions made in the past. Note that this is just a fall-back to the current webshopping model.
Countermeasure: $\mathcal{D}$ is trusted to encrypt its database using a strong encryption scheme and store the decryption key safely and externally (not along with the database). $\mathcal{D}$ also needs a secure and an efficient key-retrieval mechanism as it frequently needs the decryption key for processing deliveries to its customers. With these measures in place, if there is a breach, we can avoid both the consequences mentioned above.

4.5.2 Security attacks

1. $\mathcal{P}$ tries to create new shopping-cart or payment-acknowledgement credentials and claims for products from $\mathcal{W}$ with these fake credentials.
   Countermeasure: This attack is not possible because $\mathcal{P}$ does not have $\mathcal{W}$'s secret key. Here we rely on the unforgeability guarantee of the ABCs.

2. $\mathcal{P}$ steals someone's credentials and tries to prove as her own.
   Countermeasure: This attack is not possible because all attribute-based credentials are associated to a specific user's secret key and without that secret key, $\mathcal{P}$ cannot create a proof for that credential and thus cannot succeed in authenticating or signing with somebody else's credentials. The way the secret key is bound to the user is out-of-scope here as it is part of the ABC implementation.

3. $\mathcal{P}$ tries to create a new `PayCred` without $\mathcal{B}$'s involvement.
   Countermeasure: This attack is not possible because `PayCred` credential is signed by $\mathcal{B}$ using its own private key and $\mathcal{P}$ does not have $\mathcal{B}$'s private key to create such a credential. Here we rely on the unforgeability guarantee provided by the ABCs.

4. $\mathcal{P}$ uses an old `PayCred` to pay for the current shopping session – double-spending scenario.
   Countermeasure: This attack is not possible as a `PayCred` includes the shopping cart identifier ID_c. The webshop can easily detect double-spending if it sees the same ID_c in consecutive `PayABS`s (involving `PayCred` credentials) presented by $\mathcal{P}$. Thus, a `PayCred` can be used only to pay for a particular shopping transaction (i.e. a payment-credential is specific to a cart and the webshop).

5. $\mathcal{P}$ denies receiving the package.
   Countermeasure: $\mathcal{P}$ provides an attribute-based signature `DelivABS` on the message *"Delivery Acknowledgement"* to $\mathcal{D}$ in the case of home delivery and $\mathcal{L}$ in the case of locker delivery. This ABS proves to $\mathcal{D}$ (or $\mathcal{L}$) that she holds the correct cart identifier ID_c from the $\mathcal{W}$-issued credential `PayAckCred`. At the same time, it ensures that $\mathcal{P}$ cannot deny receiving the package later.

6. $\mathcal{P}$ tries to replay a `PayABS` from another shopping transaction at the same

webshop if a successful cart identifier collision is found, i.e., if two transactions get the same identifier assigned by the webshop.
Countermeasure: This replay attack is not possible because in our scheme, every ABS includes an unpredictable nonce sent by the verifier. Even when the cart identifier and the context-specific message are the same, the nonces will differ for two `PayABS`s. So $\mathcal{W}$ can detect if an ABS was replayed.

7. $\mathcal{W}$ tries to modify the amount in $\mathcal{P}$'s `PayABS`.
   Countermeasure: This attack is not possible because the ABS will not be valid if any change is made to the attributes or the message included in the signature. Here we rely on the integrity guarantees provided by ABSs.

8. $\mathcal{W}$ tries to collect payment twice by repeatedly presenting the same `PayABS` from $\mathcal{P}$ at $\mathcal{B}$ during payment collection.
   Countermeasure: This attack is not possible because, $\mathcal{B}$ stores the `PayABS` (Each `PayABS` includes the cart identifier which is unique for a shopping session) and if it is presented for the second time, then $\mathcal{B}$ checks it against its records and rejects it.

9. $\mathcal{B}$ fails to add an outstanding payment entry (maliciously or erroneously) to its pool after a customer's payment initiation (i.e. after issuing `PayCred` to a customer) which results in possible denial of payment to $\mathcal{W}$ during payment collection.
   Countermeasure: Under the above circumstance, if $\mathcal{W}$ provides a valid `PayABS` with a fresh ID_c that is not present in $\mathcal{B}$'s double-spend database, then $\mathcal{B}$ following the scheme pays the cart amount to $\mathcal{W}$. Here $\mathcal{B}$ relies on the unforgeability property of ABCs which makes it impossible for a payee (i.e., $\mathcal{W}$) to have come up with a fake payment-credential `PayCred` and a corresponding `PayABS` that includes attributes from this credential, even if $\mathcal{W}$ had colluded with the purchaser $\mathcal{P}$.

4.6 Related Work

Smith et al. [62] survey existing technologies that promote consumer privacy in e-commerce. They split the range of privacy-enhancing technologies that have been proposed in the literature into two main categories: 1. Those that attempt to preserve an individual's privacy by enabling anonymous communication channels for interaction between a customer and an e-business; 2. Those that attempt to minimise the amount of personal information given to an e-business during the interaction. In our webshopping scheme, we assume that all the protocol communication take place within an anonymous channel (point 1.) and mainly focus on achieving privacy through data minimisation (point 2.).

In their position paper [63], Diaz et al. review the e-shopping process and discuss privacy threats in each of its stages (i.e., purchase, payment, delivery and completion). They argue that it is not enough to protect a single stage (e.g. only payment) but rather that a complete solution that deals with threats and data leaks in every stage and interconnections between the stages is necessary. In this chapter, we devise a privacy-friendly shopping scheme using ABCs and corresponding protocols for the overall online shopping process that deals with the following privacy threats mentioned in [63]: leaking of shopped products to the bank, linking of a purchaser and a webshop by the bank or third parties, and the webshop or the third parties learning a purchaser's delivery address. The threat of purchasers' personal data getting leaked to the webshop if private feedback is involved at the completion of a shopping transaction, is out of scope in our webshopping scheme.

To make the purchase anonymous, many cryptographic e-cash schemes have been proposed in the literature (e.g [64, 65, 66, 67]) which make the cash withdrawal and deposit independent of each other. However, our online shopping scheme relies on traditional money and centralised banks for payment, but it uses blind issuance and selective disclosure properties of ABCs to make the cash withdrawal (payment credential issuance) and deposit (payment collection by the webshop) stages independent of each other.

Zhang et al. [68] propose a true fair exchange shopping protocol that incorporates physical delivery by using a delivery cabinet (similar to our locker). Although their protocol ensures anonymity of the customer and the merchant, it does not achieve unlinkability of a customer's transactions at the merchant as our protocol, because it heavily relies on public-key encryption and signatures.

Alqahtani proposes an e-commerce protocol[69] which ensures fair exchange of information and digital goods between a customer and a merchant with the help of a semi-trusted third party. Their protocol also hides the identity of the customer from the merchant by using digital cash for payment and anonymous channel for information exchange. In contrast, our protocol is not restricted to digital goods and the payment through digital cash (or cryptocurrencies). Thus, our protocol has a broader scope and can be more easily integrated with the existing infrastructure for online shopping.

4.7 Discussion

In this section, we discuss ways in which the proposed attributes-based webshopping scheme can be implemented and present the details of our prototype implementation. Then we present a brief comparison between our webshopping scheme and anonymous marketplaces.

4.7.1 Implementation aspects

First we discuss two models for putting our attribute-based webshopping protocols into practice. See Table 4.3.

1. A purchaser shops on her personal computer (PC) via a shopping website and uses a smartphone application that implements ABCs which we call the ABC app for receiving and using credentials during the shopping transaction.
2. A purchaser shops on her smartphone and uses the ABC app on the same de-

vice; the ABC app is invoked at every credential issuance and usage (signing) instance over the course of a shopping transaction. Here, both the shopping and ABC applications on the smartphone communicate via inter-app communication.

Table 4.3: Implementation models for attribute-based webshopping scheme

Model	**Shopping app**	**ABC app**
1	PC(Desktop/Laptop)	Smartphone
2	Smartphone	Smartphone

We used IRMA's smartphone implementation of ABCs [37] to develop a basic prototype of our shopping scheme. This prototype implementation was carried out as a part of a master student's internship project. Specifically, this prototype includes the ABC issuances and selective disclosure proofs that make up most of the interactions between the participants in our webshopping scheme. Under this prototype implementation, the ABC application (ABC app) stores the purchaser's ABCs and creates selective disclosure proofs on her behalf during a shopping transaction. However, it does not implement blind issuance of attributes and selective disclosure proofs with context-specific messages as one of the inputs.

The prototype follows the first model (shown in Table 4.3), that is, the webshop is on the purchaser's PC and the ABC app is on her smartphone. On the client (i.e. purchaser) side, the prototype makes use of the ABC app on the smartphone in addition to a web browser, the Tor network (or any other anonymous network) and TLS on the PC. On the server (i.e. webshop) side, it runs a web server that calls IRMA's credential issuer and verifier modules.

Although the prototype is not a full-fledged implementation of the webshopping scheme, it successfully demonstrates that:

- developing an online shopping framework by using existing ABC implementations is relatively easy, and
- privacy-preserving webshopping transactions with ABCs are not only feasible but also efficient. There is no observable delay in comparison with the prevalent webshopping transactions.

4.7.2 Comparison with anonymous marketplaces

In the last decade, we have witnessed the emergence, flourishing and sometimes demise of online anonymous marketplaces (e.g. Silk Road, Agora, Silk road 2.0). Such marketplaces are designed to provide an online rendezvous place for sellers and purchasers. Some of their main features as mentioned in [61] are as follows. These marketplaces themselves do not sell goods but they act as mediators between purchasers and sellers. They enforce both purchasers and sellers to manually register and log in to view the listings and initiate a shopping transaction. They act as risk management platforms for transacting parties by providing superior anonymity guarantees to both purchasers and sellers compared to other alternatives, payment escrow and dispute resolution mechanisms. The anonymity guarantees shield the transaction participants to some degree from law enforcement intervention[10]. Payment escrow systems aim to prevent financial risk and they are similar to those developed by e-commerce platforms such as eBay or the Amazon Marketplace. Such an escrow system allows the marketplace to adjudicate any dispute that could arise if a seller claims the item has been shipped, but the purchaser claims not to have received it. Furthermore, online anonymous marketplaces provide a feedback system to enforce quality control of the goods being sold.

Emboldened by the anonymity properties of marketplaces such as Silk Road, sellers and purchasers often traded narcotics and contraband. Some of these marketplaces were eventually seized by law enforcement agencies, voluntarily shut down or fraudulently closed due to absconding operators [61]. Although, nowadays, new anonymous marketplaces[11] have replaced the old ones in response to market and user demand, their future seems very uncertain.

We believe that events such as marketplace shutdowns can be avoided if some control can be exercised on who is selling what. If an online marketplace recruits only the registered (or tax paying) sellers who sell legal goods, then fraud is automatically curbed. This is precisely why we do not focus on seller or product anonymity in our proposed scheme. We protect the privacy of the purchasers alone and enable direct communication between identified sellers and unidentified purchasers without a trusted third party such as a marketplace operator. As we also use traditional money and banks to handle payment, we identify the seller and the value of the transaction to the bank. This discourages the coming up of illegal webshops and also allows the banks to exercise some control at payment collection, for instance, to detect fraudulent transactions, to ask the webshop to reveal the nature of the goods involved in such transactions, and to take suitable follow-up actions, such as abort the payment to the webshop. Our aim is to provide privacy (data minimisation, anonymity and unlinkability of transactions) for purchasers. That is, in our scheme, purchasers can freely buy any product (regular or sensitive) at a legitimate webshop without being watched over by either the webshop or the bank. The possibility of blacklisting purchasers by webshops is not considered in our scheme, however, technically, we can use the epoch-based revocation scheme [58] (or other ABC revocation mechanisms) for that purpose. Nevertheless, our focus is the construction of a privacy-friendly webshopping scheme with basic functionality for ordering products at a webshop, paying for them and getting the products delivered; revocation/blacklisting is an orthogonal problem, and out of scope in this research.

[10] Physical items still need to be delivered, which is a potential intervention point for law enforcement.

[11] Dark web-market list: `https://darkwebnews.com/dark-web-market-list/` [last accessed: August 30, 2018]

4.8 Future work

In this chapter, we propose a new way of shopping online based on attributes. This is just a first step to ensure that a minimum amount of data is shared among the parties involved in a transaction. We see that all parties benefit in some way by this approach. However, in the future, the proposed webshopping scheme can be improved in several ways depending on the requirements and trust assumptions in a particular shopping ecosystem.

Some of technical improvements that provide a higher level of privacy to purchasers under stricter trust assumptions are mentioned below.

- *Hiding the cart amount σ from the bank.* In the current scheme, a very specific amount σ could lead the bank to link the payment credential issuance and spending instances. Then the bank can link which of its customers purchased at the webshop. To prevent this, we suggest that purchasers could get payment credentials for a rounded up amount so as to decrease the probablility of linking by the bank. However, there could be more elegant solutions to hide the cart amount from the bank which are subject to future work.
- *Hiding the time correlation between `PayCred` issuance and its use in the `PayABS` from the bank.* In Section 4.5.1, we mention how a bank can use the time correlation to trace a specific purchaser who was involved in the shopping transaction in the case of bank-webshop collusion. In future, we need to find ways to eliminate the timing side channel under the assumption that the bank and the webshop can collude.
- *Using cryptocurrencies (or other anonymous payment methods) for payment.* In our webshopping scheme, the payment module using a payment credential could be replaced by an anonymous payment module (e.g. some cryptocurrency). As this solution does not use a central entity such as the bank, the purchaser traceability issue mentioned in the above two points might become a non-issue. Then the scheme should be evaluated if this new mode of payment can provide all the security and privacy guarantees envisioned by our webshopping scheme.
- *Avoid using persistent identifiers.* In future, we could find a way to prevent linking of the webshop and delivery company's database based on a persistent identifier ID_c. While doing so, we need to keep two points in mind:
  - $\mathcal{W}$ needs to know some identifier to link the shopping transaction with the package of the purchaser. It uses this identifier to label the package.
  - $\mathcal{D}$ needs to be able to pick up the right packages from $\mathcal{W}$ and then map the labels on the packages back to its customers and their delivery addresses.

  We present an idea in this direction whose evaluation is subject to future research. This idea makes use of a separate identifier for a purchaser's package that cannot be linked to a particular shopping cart. In this case, the webshop and the delivery company would know only their respective identifiers: cart identifier ID_c and package identifier ID_p without the possibility to link them and identify the purchaser.

Under this idea, $\mathcal{W}$ and $\mathcal{P}$ together generate a new package identifier ID_p in a way that the final value of ID_p is known only to $\mathcal{P}$. Then, $\mathcal{W}$ issues `PayAckCred` to $\mathcal{P}$ with $\mathcal{D}_{id}, [ID_p], ID_c$ attributes. $\mathcal{P}$ encrypts ID_p with a symmetric key that it shares only with $\mathcal{D}$ and sends the ciphertext c to $\mathcal{W}$ to label her package. Then she connects to $\mathcal{D}$, sends a delivery request with `HomeDelInitABS`: $\mathcal{ABS}\Big\{C_2^{\mathcal{W}}(\mathcal{D}_{id}, ID_p, [ID_c])\Big\}\big(n_2, msg_2\big)$ where the msg_2 includes $\mathcal{D}$'s nonce, $\mathcal{P}$'s delivery address and the ciphertext c. Note that this ABS hides the cart identifier ID_c from $\mathcal{D}$. Next $\mathcal{D}$ collects all the packages that are marked to be delivered by it from $\mathcal{W}$ and then maps the package labels with the ciphertexts it has received from purchasers as a part of delivery requests. After spotting $\mathcal{P}$'s package, $\mathcal{D}$ decrypts c with the symmetric key (unique for a $\mathcal{P}$-$\mathcal{D}$ pair), checks if the decrypted value equals the ID_p disclosed in $\mathcal{P}$'s `HomeDelInitABS`. If the check is successful, then $\mathcal{D}$ securely erases the ciphertext c from its database. By this, we prevent c from becoming a linking identifier in the case $\mathcal{D}$'s database is breached. In the case of a breach, an external adversary who has access to both $\mathcal{W}$ and $\mathcal{D}$'s databases cannot link shopping transactions with the delivery addresses. $\mathcal{D}$ must still ensure that it stores the symmteric key used for encrypting ID_p safely and externally from this database. Otherwise, the adversary can use this key to encrypt ID_p and use the ciphertext to link shopping data and delivery data to compromise the purchaser's privacy. If the symmteric key is freshly generated by $\mathcal{P}$ for every shopping transaction and securely sent to $\mathcal{D}$ in the delivery request, then $\mathcal{D}$ uses this key to decrypt c on the package and then erases both the ciphertext and the key. In this case, $\mathcal{D}$ only stores ID_p in its database which cannot be used by anyone to link the shopping and delivery data even when the database is breached.

There is scope for future work in other areas as suggested below.

- *Formal security and privacy analysis.* A shopping transaction under our scheme is composed of many secure sessions where each session consists of multiple message exchanges among the participants. We perform an informal analysis on our scheme by considering various ways in which an adversary (external or internal) can compromise the security and privacy of the scheme. However, a more formal analysis is to be made on the proposed webshopping scheme. (This may use either a game-based approach or a universal composability framework to prove the security and privacy guarantees provided by the scheme.)

- *Product return scenario.* Currently, we only provide a sample protocol for product returns via a locker and do not include it in the security and privacy analysis as it is not the primary focus of the chapter. In the future, this protocol could be extended to the home delivery scenario. Also, the security and privacy of the webshopping scheme together with the returns, replacement and refund protocols could be analysed.

- *Implementation.* The current prototype implementation only aims to demonstrate the feasibility of using ABCs in shopping transactions. Future work includes the implementation of blind issuance of attributes and, attribute-based signatures which are in fact selective disclosure proofs with context-specific messages. Furthermore, the webshopping scheme could be implemented purely on a smartphone such that a purchaser can carry out a shopping transaction solely from her phone. In this case, the webshop (that is open as a website on the phone's browser or as a shopping app) and the bank communicate directly with the ABC app in an automated manner without much intervention from the purchaser. This automation will lead to an enhanced user experience and a shorter duration for each shopping transaction than in the current prototype implementation.

4.9 Conclusion

Attribute-based credentials (ABCs) make it possible to design applications with security and privacy simultaneously and they also allow great flexibility for defining and enforcing contextual policies in relation to the attributes. Using the two specific ABC protocols – issuing and selective disclosure of attributes (either for authentication or signing purpose) – a wide variety of web transactions can be described. Unlike earlier work which mostly focused on authentication, authorisation and encryption with attributes, we have demonstrated a more general approach. In this chapter, we have described how ABCs can be used in the design of privacy-preserving electronic commerce which offers privacy for purchasers. Our data-minimising webshopping scheme is also incentivising for webshops which are data controllers under data protection regulations. For instance, the European GDPR makes privacy by design and by default mandatory for all data controllers and imposes high penalties for the rule violators. The scheme also creates new business opportunities for banks (e.g. facilitating anonymous payments) and for delivery companies (e.g. offering subscription-based delivery services). Furthermore, our scheme can be efficiently implemented with existing components of ABCs on smartphones.

It is expected that ABCs will be applied in various other contexts, such as anonymous donations, ridesharing, discount vouchers, and dissemination of electronic goods (e.g. media streaming, e-books). With this research, we aspire to encourage a privacy-preserving way of thinking about the applications on the web, and we hope that it will inspire other researchers and developers as well.

Part II

Techniques to strengthen security and trust in ABC applications

Chapter 5

Securing user keys with TANDEM

5.1 Introduction

The security of cryptographic schemes such as, attribute-based credential schemes [15, 16], electronic cash schemes [70, 71] hinges on the security of the underlying keys that are locally stored on users' devices. A straightforward approach to achieving key security is to use secure hardware, e.g., smart cards, hardware tokens, or trusted execution environments [72, 73, 74], both to store the keys and to perform cryptographic operations with them. Hardware-based solutions offer strong security guarantees but they might not be available on, or too expensive to add to user-devices, not accessible to developers [75, 76], or harmful to usability [77]. Software-based solutions to store and manage keys may be easy to setup and maintain, but, they are extremely difficult to secure [78, 79, 80, 81].

An alternative is to use a secure central server to store the keys and to perform cryptographic operations on behalf of users. A centralised solution is easier to secure and additionally enables to easily *block a user's key* whenever the user's device is lost, stolen, or compromised; and to rate limit the use of a key to ensure that it cannot be abused, for instance, to launch denial-of-service attacks. However, centralisation introduces new security and privacy issues. First, users must trust the central server to not impersonate them. Second, by mediating users' interactions with other services, the central server may learn private information. Both threats are analysed by Brandão et al. in the context of nation-scale brokered-identification systems [12]. They show how a central hub that acts as the broker between users, identity providers and service providers can impersonate users, link users' transactions across different service providers, and also learn private identifiable information of users. In addition to the above issues, the central server becomes a single point of failure in terms of availability and also a valuable target for attackers as it stores user identities and corresponding transaction logs.

A natural solution to the impersonation problem is to involve a user (actually, the user's personal device such as, smart phone) in the storage and the usage of the keys by using threshold cryptography. That is, we can share the key between the user and the central server. Additionally, this approach strengthens authentication security: To use the key, the user not only needs to authenticate herself to the central server but also needs to have a key share, i.e., authentication is augmented with 'something that the user has'. However, threshold cryptography does not address the privacy or the availability concerns associated with centralisation. Let us focus on the privacy concerns. The central server recognises a user by her authentication credential and the key-share that she holds at the server. Basically, these pieces of information act as the user's pseudonym that the server can use to link all the threshold operations to this user and learn the key-usage patterns. If the pseudonym is linked to the real identity of the user at the server, for instance, if the user has given her name, email address etc., to the server at the time of registration, then the server can identify the user whenever she authenticates and link all key-usage to the identity of the user. The privacy concerns for the user escalates when the server colludes with verifiers (service providers). Because the time of access of the key-share at the server and the time of its use at a service provider are almost the same, the server can use this timing information to deanonymise users (assuming the server knows user identities) for the service providers in anonymous transactions, e.g., linking a user's identity to the use of an anonymous credential, correlating interactions to public activities such as updates to a public blockchain ledger [82]. When the server only knows the user's pseudonym (and not the real identity), then the server can still link subsequent threshold operations performed by the user and tell the colluding service provider if the same user was involved in two or more anonymous transactions. Together they can build a profile for the user.

In this chapter, we present Tandem: a set of protocols that augment threshold-cryptographic schemes to enable secure and privacy-preserving usage of key shares stored on a central server. To use a key, a user sends a *one-time-use key-share token* to the central server using an anonymous communication channel. This token contains a randomised version of the central server's key share for this user. The server uses this key share to run the threshold-cryptographic protocol *without* learning the user's identity. The construction of key-share tokens permits to decouple the stages of obtaining and using the tokens, eliminating the possibility of time-correlation attack. The one-time property enables two additional functionalities. Without the need to identify users, it enables the blocking of keys in case the user's key-share is lost or compromised, and the rate-limiting of key-usage to restrict how often an attacker can use an unblocked key.

To demonstrate the potential of our approach, we use Tandem to enhance the security of attribute-based credentials (ABCs) [83, 16, 15, 18] when they are used on a user's personal device (e.g. smart phone, tablet) [20]. All the ABCs on the user's device are bound to her secret key and if this key is compromised, then the security provided by the ABCs is entirely lost. Although traditional two-party threshold cryptography involving a user and a central server protects the security of the user's secret key in ABCs, it is vulnerable to time-correlation attack. For example, consider a user who uses an ABC to anonymously access a health service such as MIT Medical[1], or NHS online health services[2]. A malicious central server can learn which user accesses which health service based on the time of key access and use. By running threshold ABC protocols using Tandem, the user can show her credential to the health service without ever having the complete key in her device and without letting the central server find out who is using the key, guaranteeing the user's privacy even if the server and the service provider collude.

For the purpose of demonstration we derive a threshold version of Idemix [15, 18] credentials, which we augment with the Tandem protocols. We note that Tandem can also be applied to threshold versions of other credential systems such as, U-Prove [16] and BBS+ [83][3]. The anonymity provided by ABCs opens the door to malicious users abusing service providers. We show a simple modification to the threshold ABC schemes that enables service providers to confirm that Tandem is used, as long as all users use Tandem. So the service providers can trust Tandem for not allowing blocked or rate-limited users to use ABCs. This way Tandem obviates the need for complex ad-hoc cryptographic techniques to block users [84, 85] or limit key-usage [86].

In a broader context, Tandem can be used to secure the keys of any cryptographic scheme (e.g., encryption, signature, or payments) for which a *linearly randomisable* threshold-cryptographic version of the scheme exists. As long as the threshold version is *private*, i.e., the scheme does not require information that identifies the user besides the key, Tandem ensures that not even a malicious central server can learn with which user it is interacting. For example, Tandem can be applied to threshold variants of Schnorr [87] and RSA signatures [88], ElGamal-based [89, 90] and RSA decryption [88], as well as threshold-cryptographic versions of electronic cash schemes [70, 71]. Tandem cannot be applied to the threshold DSA schemes because they are multiplicative [91] or require identifying auxiliary information [92].

In summary, we make the following contributions:

✓ First, we describe a basic key-sharing scheme that is prevalent in traditional threshold cryptography. We show that it can protect the security of user keys in terms of blocking and rate-limiting the use of keys but not the privacy of users if the server is malicious with respect to privacy. A simple time-correlation attack can break a user's anonymity with respect to such a server, assuming that the server knows the user identities.

✓ We introduce Tandem; it enables the use of threshold-cryptographic protocols with a central server to secure cryptographic keys without this server learning what keys are used by whom. Additionally, Tandem enables blocking and rate limiting of key usage. Tandem involves many cryptographic details in its construction that are needed to make it provably secure and private. However, to be consistent with the level of abstraction followed in the other chapters of this thesis, we only provide the essence of Tandem in this chapter without going deep into the cryptographic details. For the detailed description of Tandem protocols, refer to the full paper [24]. .

[1] `https://medical.mit.edu/services/mental-health-counseling`

[2] `https://www.nhs.uk/Conditions/online-mental-health-services/Pages/introduction.aspx`

[3] The application of Tandem to BBS+ credentials is described in the original paper [24].

✓ We provide a threshold version of an attribute-based credential system, and show how TANDEM can be used to augment its security. We show how the underlying constructions in TANDEM permit rate limiting and revocation of credentials *without* relying on complex purpose-built cryptographic techniques.

✓ We prove the security and privacy of TANDEM, and we use a prototype implementation to validate its practicality. All operations in TANDEM take less than one second, imposing a reasonable overhead on both server and users. However, in this thesis, we leave out the formal security analysis and the implementation details as they are not the author's contributions. The reader is referred to the full paper [24] for these details.

5.2 Basic Key-sharing Scheme

We consider a scenario in which *users* are required to perform cryptographic operations to interact with a *service provider* (SP). Users use insecure devices, such as smartphones, tablets, or laptops, without secure hardware, to run the cryptographic protocols. To keep their keys safe, they use a central server to run cryptographic protocols in a distributed way.

In a basic key-sharing scheme, a user's key x is additively shared between two parties: the user's personal device and the central server. Both key shares are necessary to successfully carry out a threshold cryptographic protocol. Under this scheme, when a new user joins the system she generates long-term key shares jointly with the server. Let the user's key share on her device be x_U and key share on the server be x_S. Now the user's key is $x = x_U + x_S$. When she needs to run a threshold cryptographic protocol (TCP), the user first authenticates to this server so that the server can retrieve the key-share x_S corresponding to the user. The user and the server then execute the TCP using the key-shares x_U and x_S as their respective inputs.

As an example of a basic key-sharing scheme, we show how the user's side of Schnorr's protocol (a zero-knowledge proof of knowing a discrete logarithm) [27] can be distributed between the user and the central server. Schnorr's protocol allows a user to demonstrate to a relying party (e.g. a service provider) that she knows a private key x corresponding to a public key $h = g^x$, where g is the generator of a cyclic group $\mathbb{G}$ of prime order p. To jointly create the Schnorr proof, the user and the server essentially each create a proof of knowledge of their own secret, the user for x_U, the server for x_S. The user combines these proofs to obtain a proof of knowledge of the original secret x where $x = x_U + x_S$, see Figure 5.1. Clearly the service provider's view of the Schnorr's protocol is unchanged. Since the server's proof is a proof of knowledge as before, the user does not learn anything about x_S. However, the server remains essential for every use of the user's private key x.

This basic key-sharing scheme offers key security. On the one hand the server alone cannot use the user's key. On the other hand, if an attacker compromises the user's device that holds x_U, the user can authenticate to the server (e.g. using her username and password) from another device and request it to block her key. This

Common information: $\mathbb{G}, g, h = g^x$				
Server $x_S \in \mathbb{Z}_p$		User $x_U \in \mathbb{Z}_p$		Service provider
$t_S \in_R \mathbb{Z}_p$ $u_S = g^{t_S}$	$\xrightarrow{u_S}$	$t_U \in_R \mathbb{Z}_p$ $u_U = g^{t_U}$	$\xrightarrow{u = u_S u_U}$	
	$\xleftarrow{c}$		$\xleftarrow{c}$	$c \in_R \mathbb{Z}_p$
$r_S \equiv t_S + cx_S$	$\xrightarrow{r_S}$	$r_U \equiv t_U + cx_U$	$\xrightarrow{r \equiv r_S + r_U}$	check $g^r = uh^c$

Figure 5.1: An example for basic key-sharing: Schnorr's proof of knowledge of a discrete logarithm in which the server and the user jointly prove knowledge of $x = x_S + x_U$ such that $h = g^x$, while the server only knows x_S and the user only knows x_U.

scheme also provides key rate limitation: since the server can observe when a user accesses her key, the server can easily enforce a limit on the number of times the key is used. However, the user is recognised by the server based on her authentication credentials while using the key for a TCP. So, the privacy of the user is ensured as long as the server is trusted to keep all the key-use patterns of users to itself and not to divulge this information to any other party. But if the server colludes with a service provider, then it can use the time of key access and use to link a transaction to the corresponding user. This time-correlation attack is described in detail below.

Time correlation attack In the basic key-sharing scheme, the server learns when a user uses her key. The lack of key-use privacy may have further implications for users' privacy when the protocol between the user and an SP requires anonymity (e.g., showing an anonymous credential). By colluding with the central server, the SP can exploit the fact that, given the interactiveness of the key-sharing protocol, there is a strong correlation between the time when the authenticated user interacts with the server, and when the anonymous user interacts with the SP. Thus, for every anonymous transaction the SP engages, the anonymity set of the user is reduced to the authenticated users interacting with the TS at similar times. Such type of correlation attack has been used in the early days of Tor to identify users and hidden services [93, 94]. This attack relies solely on time correlation between accesses. Therfore, the attack cannot be prevented by making the messages seen by the TS and the SP cryptographically unlinkable [16].

There are two straightforward approaches to prevent time-correlation attacks: introduce delays or dummy requests. These solutions are, however, difficult to use in practice. In order to significantly increase the anonymity set for users, operations may need to be delayed for long time. This rules out applications that require short delays, such as showing an anonymous credential or performing a payment. Dummy traffic not only imposes an overhead on users and the TS, but it is widely known that generating dummy actions that are indistinguishable from real activity is very difficult [95, 96]—especially because it is unrealistic that users would be always online so that their devices could produce such requests.

5.2.1 Basic key-sharing to secure Attribute-based Credentials

Attribute-based credentials (ABCs) can be conceptualised as digital equivalents to classic documents like passports, driver's license, student cards, etc. Credentials contain a collection of attributes describing the owner, e.g., name, date of birth, gender, social security number and country of birth, and are signed by a relevant party, called the issuer. The owner of a credential can selectively disclose any subset of attributes to a service provider in such a way that the the validity of the disclosed attributes can be validated. In many ABC systems credentials are unlinkable, that is, users are anonymous within the set of users having the same disclosed attributes. To bind credentials to a user, and to ensure that only the owner can operate with them, credentials contain the user's secret key. Typically, all credentials of a user contain the same secret key. The security of this key is crucial to ensure secure use of ABCs. The concept of ABCs are described in detail in Section 1.1.

The IRMA identity platform [37] efficiently implements Idemix ABCs and provides a practical way to use these credentials for user authentication and signing. In the early days of IRMA, the Idemix ABCs were implemented on smart cards and an ABC owner's secret key was stored on the secure and tamper-resistant chip of her smartcard. However, when IRMA migrated to a smartphone implementation of ABCs, it gained in terms of user interfaces and better efficiency but key security became an issue. So, the author of the thesis with few other researchers involved the IRMA project started to explore practical ways to secure users' secret keys in IRMA.

As a first step towards achieving key security in IRMA, basic key-sharing scheme as described in Section 5.2 is implemented in the IRMA platform by some of the IRMA team members. This implementation involves storing a share of the user's key on a central server while all the ABCs and the other key share are stored on the IRMA application (or the IRMA app) that is installed on the user's smart phone. The Privacy by Design foundation is currently the IRMA scheme manager (entity that manages the IRMA infrastructure) and is responsible for running the central server. The server is called the *key-share server* in the IRMA jargon.

When a user registers to the IRMA system, the IRMA app generates the device's key-share x_U and informs the key-share server to generate a key-share x_S for the user. Furthermore, at registration, the app asks the user to choose her authentication credentials: (i) a PIN code to authenticate to the key-share server before using an ABC from the IRMA app; (ii) an email address to authenticate to the key-share server to enable blocking of her key-share at the server[4]. However, IRMA makes it optional for the user to provide the email address. Essentially, the key-share server knows the user's pseudonym (hash of the PIN code) and the email address (if provided by the user). So, whenever the user uses her ABCs to authenticate or sign, the user enters her PIN code and if it is correct then the key-share server participates in the threshold protocol with the IRMA app on the user's smartphone. Then the IRMA app sends the output of the threshold protocol to the service provider.

[4] When a user loses her phone and can no more authenticate with her PIN, the key-share server authenticates the user via her email address (by sending a one-time passcode to the email address).

The current implementation of basic key-sharing in IRMA is simple and secures the user's secret key that is bound to her ABCs. As the key-share server recognises the user by her pseudonym everytime her key-share is accessed, it is easy for the server to limit the number of times a user can use her key. If this pseudonym is linked to the user's email address, the server can block the user's key-share x_S when the phone is lost or stolen.

However, in this setup, if the server colludes with service providers (SPs), then it can use the timing information to link a transaction to a specific user and reveal the pseudonym and the email address of that user to the SP. This compromises the user's privacy to some extent and deanonymises users in anonymous transactions. Thus, the current implementation of the key-sharing scheme in IRMA relies on a strong assumption that the key-share server is trusted to not collude with service providers. Considering that IRMA is still in the pilot phase and PbD foundation is the only scheme manager in the IRMA system, the key-share server run by the PbD foundation is trusted not to collude with service providers. However, when IRMA scales up and will be deployed in various real-time use cases, then there may be many more IRMA scheme managers who may wish to run their own key-share servers or they may choose to use an external cloud server to perform the tasks of the key-share server. In such situations, we cannot unconditionally trust the key-share server with respect to the privacy of users. That is, the key-share server can then make use of time correlation to compromise users' privacy. So, with an aim to provide privacy to users under weaker trust assumptions about the key-share server, the author with her coauthors continued the research on this topic to come up with a more privacy-friendly key-sharing solution. This solution is described in the rest of the chapter.

5.3 Problem Statement

As we saw in the previous section, when the users' keys are shared between their devices and an untrusted central server, the privacy of users is at risk due to the presence of time correlation attack. So, we propose that users use Tandem to run threshold-cryptographic protocols with a central server while retaining their privacy towards the server. We call this server the Tandem *Server* (TS) and every execution of a threshold-cryptographic protocol by the user and the TS a *transaction*. We have chosen the name 'Tandem' for the proposed privacy-friendly key-sharing solution because the user's device and the server need to work in *tandem* to successfully perform a transaction.

For simplicity, we assume that there is only one Tandem server throughout this chapter. However, we note that the security of Tandem protocols can be increased by secret-sharing the key with multiple Tandem servers. Tandem then ensures that keys can be blocked and rate-limited as long as at least one of the Tandem servers is honest. Privacy is not affected by the number of servers.

5.3.1 Tandem properties and threat model

We now describe the security and privacy properties of Tandem protocols.

Property 1 (Key security). Tandem *protects the use of the user's key.* No entity other than the user is able to use the user's key. Even if the user's device is compromised, the user can maintain this property by *blocking the key-share* at the Tandem server. Thereafter, the attacker cannot further use her key.

Any solution that recomputes the user's key on the user's device, e.g., by deriving it from a user-entered password, does not satisfy this key-security property. In such a solution, an attacker who compromises the user's device can observe the key when it is used. Thereafter the attacker can use the key indefinitely, making blocking impossible.

Property 2 (Key rate-limiting). Tandem *limits the rate of usage of keys.* Users can limit the number of times her key is used in a given interval of time. We call this interval an *epoch.*

The security and rate-limiting properties of Tandem are related to the revocation and n-times-use concepts of attribute-based credentials [86], respectively. Yet, they are not the same. On the one hand, revocation and n-times-use credentials trust the service providers to block credentials completely and to block a credential after n uses, respectively. Using Tandem on the other hand, users need to trust only the Tandem server, which *they* choose, to block and rate-limit keys. Tandem can ensure this property for a large class of protocols, even if a system does not rely on credentials.

Property 3 (Key-use privacy). Tandem *protects the privacy of key use in transactions.* The Tandem server (TS) cannot distinguish between two users performing transactions. Even if the TS colludes with the service provider (SP) it cannot distinguish users (unless the SP could distinguish the users, in which case collusion leads to a trivial and unavoidable privacy breach).

We assume that the Tandem server is honest with respect to security. That is, it follows the protocols so as to protect the security of users' keys (Property 1) and to ensure that keys are only used the allowed number of times (Property 2). Moreover, we trust the Tandem server to be available, i.e., Tandem does not protect from denial of service. However, the Tandem server is malicious with respect to privacy: It is interested in breaching the privacy of the users by trying to learn which keys and services they use; for this, it can arbitrarily deviate from the protocol (Property 3).

5.3.2 Tandem at a glance

We now provide a high-level overview of how users can use the Tandem server to perform threshold-cryptographic protocols in a privacy-preserving way, see Fig. 5.2. The core idea of Tandem is to enable users to obtain one-time use key-share tokens for an epoch from the Tandem server (TS) and use it later (but in the same epoch) to run threshold protocols. By decoupling the access of the key-share stored at the

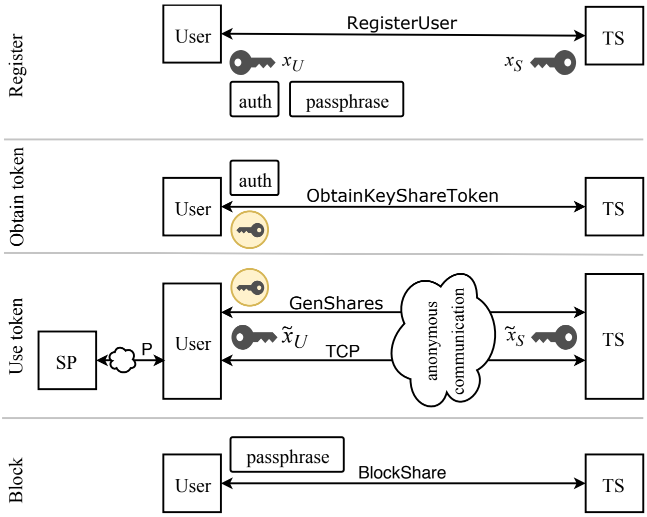


Figure 5.2: Tandem process: after registration, a user can authenticate herself and obtain a key-share token (indicated by the yellow-circled key), which can later be used anonymously to execute a threshold cryptographic protocol TCP (the user and SP run protocol P). The user can block her keys at any time. Inputs are shown above the arrows, outputs below.

server and its use at a service provider (SP), Tandem prevents the time-correlation attack by the server. Tandem is designed with an aim to prevent deanonymisation and traceability of a user by a malicious SP with the help of the TS when the SP by itself cannot distinguish users. We assume that users use an anonymous communication channel [97, 98] to communicate with the TS and SPs to protect their privacy at the network layer.

Registration. Prior to using the Tandem server to run threshold-cryptographic protocols, users need to register with the TS using RegisterUser protocol.

During registration, the user and TS jointly compute long-term shares x_U and x_S of a long-term secret x appropriate for the threshold-cryptographic protocols they seek to run later. The user obtains credentials (e.g., a password) to authenticate when obtaining a key-share token and also a means to block her keys (e.g., a passphrase); she stores the latter outside her device.

Obtain Token. First, the user authenticates herself to the TS. Then the user and the TS construct a one-time-use key-share token. This token enables the user to anonymously use her key when running a threshold-cryptographic protocol with the TS (see below). The user can obtain multiple tokens in advance during an epoch that can be used to run threshold protocols with the TS later but within the same epoch. At this stage, the TS can limit the number of tokens it provides the user, thus limiting the number of times the user can use her key. The Tandem protocol ObtainKeyShareToken handles these steps.

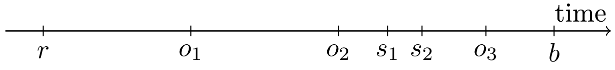
Using Keys. After obtaining tokens, a user can run threshold-cryptographic protocols with the TS. First, the user and the TS use one of the obtained tokens to derive fresh shares $\tilde{x}_U$ and $\tilde{x}_S$ of the secret x by running a protocol called GenShares. These new shares cannot be linked to x, thus can be used as subsequent inputs to a threshold-cryptographic protocol without revealing the user's identity (or even her pseudonym) to the TS. The user and the TS use the fresh shares $\tilde{x}_U$ and $\tilde{x}_S$ as one-time input to the threshold-cryptographic protocol, allowing the user to use her key in the cryptographic protocol with the service provider.

We note that the TS never communicates directly with service providers. Instead, all communication is facilitated by the user. This approach enables the user to anonymise protocol data before sending them to the TS. We note that the use of the TS may remain invisible to the service provider or the presence of the server may be made known to the SPs based on the use case. The presence of a central server may be seen as an advantage in terms of security by SPs. This is because, a malicious user cannot use the key without the cooperation of the TS and the TS only cooperates if the user presents a valid key-share token during the TCP (the user should have authenticated correctly to the TS for obtaining the key-share token in the first place). So, SPs may wish to know if the TS is involved in the computations over the user's key. In Section 5.6.1, we show how a simple modification can confirm the use of the TS in the ABC-based transactions by the user.

Blocking keys. Whenever a user wants to block her key, she uses her blocking means (e.g., the passphrase) to request the TS to block her key by using the BlockShare protocol so that it cannot be used anymore. Thereafter, unused tokens become invalid, and no new tokens can be obtained (not even by an adversary having learned the authentication credential used to obtain tokens).

Timing of events. When obtaining tokens, the user is authenticated and recognised by the server as the user corresponding to the key-share x_S. Hence, for privacy reasons, the actions of obtaining and using tokens *must* be uncorrelated. In particular, tokens should not be obtained right before being used. To avoid correlation, the user can configure her device to obtain tokens at random times or at regular times (e.g., every night), thus ensuring that tokens are always available. The device can ask the user to authenticate herself by entering her password at the designated times.

Here we show an example time line of registration (r), obtaining tokens (o_i), using tokens (s_i), and blocking the key (b) events:



This example illustrates that obtain and use events do not necessarily follow each other, but can be interleaved. As a result, the timing of these events need *not* correlate. We note that every token can be used only once. The token o_3, unused before the key is blocked by b, cannot be used after this instant.

In Section 5.5.2 we explain why private information retrieval is not a suitable alternative to decouple obtaining and using of key shares. In that section we also explain how Tandem outperforms generic constructions based on secure multi-party computation.

5.4 Cryptographic Preliminaries

Let ℓ be the security parameter. Throughout this chapter, $\mathbb{G}$ is a cyclic group of prime order p (of 2ℓ bits) generated by g. We write $\mathbb{Z}_p$ for the integers modulo p, $a \in_R A$ to denote that a is chosen uniformly at random from the set A. Furthermore, we write $[n]$ to denote the set $\{0, \dots, n-1\}$. For reference, Table 5.1 in Section 5.5 explains frequently-used symbols in Tandem.

5.4.1 Cryptographic building blocks

Tandem relies on a couple of cryptographic building blocks. First, we use an additive homomorphic encryption scheme (e.g. Paillier's scheme [99], Joye and Libert's encryption scheme [100]) given by the algorithms $\mathbf{E}^{+}_{pk}, \mathbf{D}^{+}_{sk}$ with plaintext space $\mathbb{Z}_N$ (i.e., integers modulo N) and space of randomisers $\mathcal{R}$. We write $c = \mathbf{E}^{+}_{pk}(m)$ to denote the homomorphic encryption of the message $m \in \mathbb{Z}_N$. The scheme is additively homomorphic, so

$$\mathbf{E}^{+}_{pk}(m_1) \cdot \mathbf{E}^{+}_{pk}(m_2) = \mathbf{E}^{+}_{pk}(m_1 + m_2 \pmod{N}).$$

Second, Tandem uses a CPA secure encryption scheme $\mathbf{E}_{pk_{id}}, \mathbf{D}_{sk_{id}}$ with plaintext space $\mathbb{G}$ such as ElGamal [101], that allows simple verifiable encryption[5].

Third, Tandem uses a commitment scheme that is information-theoretically hiding and computationally binding (e.g. Pedersen's commitments [102]) and that supports blind signatures.

Fourth, Tandem uses a blind signature scheme (e.g. Schnorr's blind signature [103]) for the construction of one-time-use key-share tokens.

5.4.2 Threshold-cryptographic protocols

In this chapter, we focus on cryptographic protocols run between a user and a service provider, e.g., showing a credential to a service provider or spending an electronic coin. The threshold-cryptographic version of such a protocol splits the user's key x and the user's side of the original protocol in two (or more) parts, run by different parties. Each party operates on a secret-share of the user's key. Security of the threshold-cryptographic protocol (TCP) ensures that a large enough subset of shares (two in the case of two parties) are required to complete the protocol.

We consider TCPs where the user's side of the protocol is distributed between the user and the TS. After registration, the user and the TS hold the key shares

[5] Simply put, the encryptor in a verifiable encryption scheme proves in zero knowledge that the encryption of a message satisfies some property.

x_U and x_S of x, respectively such that the user's secret key is $x = x_U + x_S$. After running $\mathsf{GenShares}$, the user and the TS hold the fresh key shares $\tilde{x}_U$ and $\tilde{x}_S$ with $x = \tilde{x}_U + \tilde{x}_S$. They then run the TCP, which we denote as:

$$\mathsf{P}(\mathsf{in}_{SP}) \leftrightarrow \mathsf{TCP.U}(\tilde{x}_U, \mathsf{in}_U) \leftrightarrow \mathsf{TCP.TS}(\tilde{x}_S), \tag{5.1}$$

where the SP, the user and the TS respectively run the interactive programs P, $\mathsf{TCP.U}$ and $\mathsf{TCP.TS}$. The user mediates all interactions between the SP and the TS. The user and the SP take extra inputs needed for the execution of the target cryptographic protocol denoted as in_U and in_{SP}. For simplicity, we denote the complete protocol from (5.1) by $\mathsf{TCP}(\tilde{x}_U, \tilde{x}_S, \mathsf{in}_U, \mathsf{in}_{SP})$.

Tandem can only enhance the privacy (Property 3) of certain TCPs. We formalise the condition that these TCPs should satisfy. To avoid that the TS can recognise the user based on the shares input to the TCP, we randomise the long-term secret shares. Thus, we require that TCPs enhanced with Tandem still function with randomised key shares. In addition, our privacy-friendly $\mathsf{GenShares}$ protocol requires this randomisation to be linear.

For simplicity, we assume that the user's secret $x \in \mathbb{Z}_p$ for some field $\mathbb{Z}_p$ of prime order p (e.g., corresponding to the group $\mathbb{G}$ we defined above) that is known to public. We note, however, that our constructions can be modified to settings with unknown order arising from RSA assumptions (See Section 5.6.2). Formally, we require the TCP to be linearly randomisable:

Definition. Let $x_U, x_S \in \mathbb{Z}_p$ be secret shares of the user's secret x. Then, we say that the TCP is *linearly randomisable* if for all $\delta \in \{0, \ldots, 2^{\ell_\delta}\}$ we have that (1) if $\mathsf{TCP}(x_U, x_S, \mathsf{in}_U, \mathsf{in}_{SP})$ completes successfully, then so does $\mathsf{TCP}(x_U - \delta, x_S + \delta, \mathsf{in}_U, \mathsf{in}_{SP})$, and (2) $x_S + \delta$ is independent from x_S.

The first condition implies that the original secret sharing (x_U, x_S) and the randomised secret sharing $(x_U - \delta, x_S + \delta)$ must share the same secret, whereas the second implies that the TS cannot recognise the user from the randomised secret share alone.

Security and privacy properties of TCPs. To ensure that a TCP with Tandem satisfies the security properties (Property 1 and 2) we require that the TCP itself is secure. That is, if the TS no longer uses its share x_S to run its part of the TCP, then no malicious user can successfully complete the TCP with the SP.

To ensure that a TCP with Tandem satisfies the privacy property (Property 3) we require that the TCP itself offers privacy with respect to the TS and the SP. That is, if the TS runs its part of the TCP using a randomised key-share as input, it cannot recognise the user, not even when it colludes with the SP.

5.5 Technical Description of Tandem

Before we describe Tandem's one-time use key-share tokens, we present a simpler construction that enables anonymous users to use their keys with the TS *without* the TS learning which key is being accessed. It uses homomorphic encryption to decouple the action of accessing the user's long-term key-share x_S at the TANDEM server from its subsequent use in the threshold-cryptographic protocols. Thus, it prevents time-correlation attacks. This construction can be thought of as the stepping stone for TANDEM.

Initially, the TS generates a private-public key pair (sk, pk) for an additively homomorphic encryption scheme (see Section 5.4). The TS publishes the public key pk. Upon registration with the TS, a user receives $\overline{x_S} = \mathbf{E}^{+}_{pk}(x_S)$ – a homomorphic encryption of the TS's key-share x_S. Because the ciphertext $\overline{x_S}$ is encrypted against the TS' key, the user does not learn anything about the TS' share.

When the user wants to *use* her key, she produces a randomised version of the TS' key-share x_S without learning the actual value of x_S (this is possible due to the homomorphic properties of the encryption scheme). To produce this randomisation, she picks a large δ and computes $c = \overline{x_S} \cdot \mathbf{E}^{+}_{pk}(\delta) = \mathbf{E}^{+}_{pk}(x_S + \delta)$. On her side, she randomises her key as $\tilde{x}_U = x_U - \delta \pmod{p}$. Then, she sends the randomised ciphertext c to the TS via an anonymous channel. The TS decrypts c to recover its key for the threshold cryptographic protocol, $\tilde{x}_S = x_S + \delta \pmod{p}$. It is easy to see that a linear TCP with randomised shares completes successfully, because $\tilde{x}_S + \tilde{x}_U \pmod{p} = x_U + x_S \pmod{p}$. Because $\overline{x_S}$ is randomised with δ in c, the TS can no longer recognise its share x_S upon decrypting c, effectively decoupling this action from the key-share generation.

In this approach, however, the TS cannot block or rate-limit keys. This is because a user can randomise the TS' key-share as many times as she wants and send it to the TS during TCP without getting recognised by the TS. That is why we extend this approach in the next section and present a construction for one-time-use key-share tokens containing signed and randomised ciphertexts like c that enables blocking and rate-limiting while preserving users' privacy.

5.5.1 TANDEM's one-time-use key-share tokens

In this section, we describe at a high level how one-time-use key-share tokens are constructed, and how they enable blocking and rate limiting by giving relevant technical details. For an elaborate description with all the cryptographic details, refer to the publication.

Simply put, a one-time-use key-share token contains a randomised version of encrypted TS' key-share and the TS's signature on this randomised ciphertext. A TANDEM user obtains such a token from the TS and uses it later to run a threshold cryptographic protocol with the TS such that there is no time correlation between the two actions. The TS cooperates with the user in a TCP only when its signature in the token is verified successfully. When the user asks the TS to block her key, the TS no longer creates key-share tokens for this user and as we will show further in this section, TS also blocks all the tokens in the current epoch that have already been received by the user. This prevents attackers from further running threshold-cryptographic protocols, even if they corrupt the user's device. Finally, since tokens are one-time use only, to restrict the number of times a user can use her key (rate-limit), the TS signs a limited number of randomised ciphertexts per-epoch per-user.

Table 5.1: Notation in Tandem protocols

Symbol	Interpretation
δ	randomiser of TS' key share
ℓ_δ	Bitlength of δ
id	Token identifier
ℓ, k	Generic and token security parameter
ℓ_k	$\log k + 2$ bits
x	Long-term secret key for a user
pk, sk	Public-private key-pair of TS
pk_{id}, sk_{id}	Public-private key-pair of the user U
p	Order of the group $\mathbb{G}$
ℓ_p	Bitlength of p
x_U	Long-term key share held by the user
x_S	Long-term key share held by the TS
$\overline{x_S}$	Homomorphic encryption of x_S
$\tilde{x}_U$	User's key share output by GenShares
$\tilde{x}_S$	TS' key share output by GenShares
ϵ	The current epoch
σ	Blind signature of the TS

In other words, TS can enforce a rate-limit by issuing limited number of tokens per epoch to a user.

Below we describe the Tandem protocols for (i) registering users with the TS, (ii) obtaining key-share tokens from the TS, (iii) using a token to execute threshold cryptographic protocol with the TS and, (iv) blocking the key-share at the TS and the unused tokens in the ongoing epoch. Table 5.1 provides a list of frequently used symbols in the Tandem protocols.

Registering Users.

Protocol 1. The RegisterUser protocol is run between a user and the TS, and proceeds as follows.

1. The user U and the TS generate secret shares $x_U \in_R \mathbb{Z}_p$ and $x_S \in_R \mathbb{Z}_p$[6], respectively. Now the user's long-term secret $x = x_U + x_S$. Then the user sets up some authentication credentials (e.g. PIN/password) that is needed to authenticate to the TS before obtaining key-share tokens. The user also generates a public-private key-pair (pk_{id}, sk_{id}) for encrypting token identifiers and sends pk_{id} to the TS. The user needs the secret key sk_{id} to block unspent tokens if needed. We assume that the user stores sk_{id} externally so that it is available even after she loses her device. We propose that the user's device generates sk_{id} based on a high-entropy passphrase (such as a Diceware passphrase[7]), so that users can write down this string as a stand-in for sk_{id}.

[6] The TS generates a unique key-share x_S for every user.

[7] `http://world.std.com/~reinhold/diceware.html`

2. The TS encypts its key-share as $\overline{x_S} = \mathbf{E}^{+}_{pk}(x_S)$ and sends the encrypted version $\overline{x_S}$ to the user. Furthermore, to ensure that the TS cannot hide an identifier in higher-order bits of x_S that are not randomised by the user in the remainder of the protocol the TS proves that the plaintext x_S is in the correct range.
3. The TS records $(x_S, \overline{x_S}, pk_{id})$ for this user, and marks this user as active. The user stores $(x_U, \overline{x_S}, pk_{id})$ on her device, and stores sk_{id} externally.

Obtain a Key-share Token. In this protocol, the user randomises the ciphertext $\overline{x_S}$, proves to the TS that the randomised ciphertext c is of the correct form and then obtains a blind signature on c from the TS.

PROTOCOL 2. The ObtainKeyShareToken protocol is run between a user and the TS, and proceeds as follows.

1. The user recovers $(x_U, \overline{x_S}, pk_{id})$ from storage, and authenticates to the TS (using her password, for instance). The TS aborts if this user exceeded the rate-limit for the current epoch, was banned, or was blocked. Otherwise, the TS looks up the user's record $(x_S, \overline{x_S}, pk_{id})$. The user is recognised by the TS in this protocol; that is, the TS knows which user is obtaining the tokens for which key-share.
2. To construct a token the user picks a large randomiser $\delta \in_R \{0, \ldots, 2^{\ell_\delta}\}$[8] and homomorphically computes $c = \overline{x_S} \cdot \mathbf{E}^{+}_{pk}(\delta)$, a randomised encryption of the TS' key share. The user sends a commitment C to the ciphertext c to the TS, together with a proof that the committed c was constructed by randomising $\overline{x_S}$. This proof is needed to enable secure blocking (See the explanation below.). The user also generates a token identifier $id \in_R \mathbb{Z}_p$ at random and encrypts the token identifier as $\overline{id} = \mathbf{E}_{pk_{id}}(id)$ and sends $\overline{id}$ to the TS in this step.
3. Then the user engages with the TS to obtain a blind signature σ on the tuple $\{c, id, \epsilon\}$ where c is the randomised ciphertext (containing randomised x_S), id is the token identifier and ϵ is the ongoing epoch. The values of c, id and the final signature σ are only known to the user at this stage. The user stores the token $\tau = (\sigma, c, id, \epsilon)$ and the randomiser δ.

For the blocking of keys to be effective, attackers must not be able to construct key-share tokens for a blocked user. Here is where the proof becomes handy that c is constructed as $\overline{x_S} \cdot \mathbf{E}^{+}_{pk}(\delta)$, where $\overline{x_S}$ belongs to the current user. Suppose that we omit the proof. Then, an attacker controlling an unblocked user can create tokens for a corrupted blocked user. The attacker uses the unblocked user's account to make the TS blindly sign encrypted key shares for the blocked user. The attacker can use the resulting token to use the blocked user's key, defeating the purpose of TANDEM. This attack is prevented if the TS verifies which user's key share is embedded into the ciphertext before blindly signing it.

However, it seems difficult to prove directly, for example in zero-knowledge, that the randomised ciphertext produced by the user is of the correct form. Therefore, we use a standard cut-and-choose approach [104, 64] to allow the TS to check that the encrypted key share it is blindly signing is correct with overwhelming probabil-

[8] The ideal size ℓ_δ for δ for the TANDEM protocols is mentioned further in the section.

ity. On a high level, the cut-and-choose approach used in TANDEM is described as follows. The user constructs $2k$ randomised ciphertexts $c_i = \overline{x_S} \cdot \mathbf{E}^+_{pk}(\delta_i)$, and sends commitments C_i to them to the TS. The TS then asks the user to open a subset $\mathcal{D}$ of cardinality k, so that the TS can verify that these k ciphertexts were correctly formed. Having checked all opened ciphertexts, the TS blindly signs the remaining k ciphertexts. By nature of the cut-and-choose protocol at least one of the remaining ciphertexts is a correct randomisation of $\overline{x_S}$ with high probability. Refer to the original paper [24] for the detailed description of how the cut-and-choose method is applied in ObtainKeyShareToken protocol.

An important aspect in ObtainKeyShareToken protocol is that a randomiser δ chosen by the user should fully hide the TS's key-share x_S. This prevents the TS from recovering the original key-share from the randomised key shares. Keeping this in mind, we choose the bitlength of δ, $\ell_\delta = \ell_p + \ell + \ell_k$. So, when the cut-and-choose method is applied while obtaining the key-share token, the size ℓ_δ ensures that k unopened $x_S + \delta_i$ values statistically hide x_S. So when these values are revealed during GenShares protocol, the TS cannot recognise x_S and identify the user based on that. Furthermore, we require that the modulus of the homomorphic encryption scheme, $N > 3 \cdot 2^{\ell_\delta}$ to ensure no overflows occur. We can take the bitlength of N to be $\ell_\delta + 2$. For instance, in a system with $\ell = 128$, $\ell_p = 256$, $\ell_k = 10$, the ideal length of randomisers ℓ_δ is 256+128+10 = 394 and N is at least 396 bits long.

Using a Key-share Token. The user can anonymously use a previously-obtained token to execute a threshold cryptographic protocol with the TS. The protocol for this is described below.

PROTOCOL 3. The GenShares protocol is run between an anonymous user and the TS.

1. To run a threshold-cryptographic protocol the user anonymously contacts the TS and sends her key-share token $\tau = (\sigma, c, id, \epsilon)$.
2. The TS checks that σ is a correct signature on (c, id, ϵ), token id was not blocked or used before and ϵ corresponds to the current epoch. The TS aborts if any check fails. Then, the TS recovers the randomised key-share $\tilde{x}_S = \mathbf{D}^+_{sk}(c) \pmod{p} = x_S + \delta \pmod{p}$ and uses it as the key for the threshold cryptographic protocol.[9]
3. The user, on the other hand, uses $\tilde{x}_U = x_U - \delta \pmod{p}$ as the key. Because c is fully randomised, the TS cannot leverage it to identify users. Moreover, as σ is a blind signature on c the TS cannot use σ or c to link the token creation to the token use.

Blocking the Key. To block her key, the user runs the BlockShare protocol with TS to ensure no new key-share tokens are created for her, and that all her unspent tokens are blocked.

PROTOCOL 4. The BlockShare protocol is run by a user and the TS. The user takes as input her long-term key sk_{id} (which she recorded outside her device). The user

[9] Here we have provided a simplified representation of the actual interactions between the user and the TS during this step. Refer to the GenShares protocol description in the publication for all the cryptographic computations done by the user and the TS in this step.

authenticates to the TS (possibly using sk_{id}). The TS marks the user as blocked, so that it will no longer issue new tokens. Then they continue as follows to invalidate unspent tokens. The TS sends a list of all encrypted token identifiers $\overline{id}_1, \ldots, \overline{id}_t$ that the user obtained in this epoch. The user looks up a list of all spent token identifiers (see below). The user then uses sk_{id} to decrypt $\overline{id}_1, \ldots, \overline{id}_t$ and sends the decrypted token identifiers that have not yet been spent to the TS. The TS will then block all tokens with these identifiers.

Since we assume the TS is honest with respect to blocking, the TS accurately provides the list of encrypted token identifiers. In the $\mathsf{ObtainKeyShareToken}$ protocol, the user verifiably encrypts the token identifier id. As a result, even if the user's device is corrupted, the TS stores a correct encryption $\overline{id}$ of id, so the above procedure blocks all unspent tokens.

In the unlikely case that the user cannot recover the identifiers, she still knows that the attacker can only use the TS a limited number of times, as the attacker is also subject to the rate-limit.

List of spent tokens. The TS may be malicious with respect to privacy. So, it might try to trick the user into revealing the identifiers of tokens she has already spent (thus revealing that these tokens were hers). In particular, the TS is not necessarily trusted to provide an accurate list of spent tokens. Therefore, we propose that users externally store spent token identifiers, so that they have a reliable record. Alternatively, the TS can keep a verifiable log of spent tokens by appending spent token identifiers to a public append-only log (users must then verify that each spent token identifier is in fact added to the log). Users can then use this log as a record of spent tokens. Finally, if epochs are short, and users are willing to risk revealing their actions in the current epoch, they can also use a list provided by the TS. If the TS is malicious, users reveal at most their actions within the most recent epoch when they block their keys.

5.5.2 Alternative constructions

An alternative method to construct tokens could be to use an authenticated encryption scheme that the user and the TS evaluate using secure multi-party computation [105]. The server inputs its key share x_S while the user inputs the randomiser δ. The user's output is the authenticated encryption of $x_S + \delta$ for the TS's symmetric key which serves as a token. To ensure that the TS cannot recognise this token, the protocol should resist malicious servers and the circuit should validate the TS' input (i.e., that they are always the same). Even though recent secure multi-party computation schemes that are secure against a malicious server boast impressive performance [106], they still require at least one order of magnitude more computational power as well as more bandwidth than the TANDEM scheme.

Another alternative construction is to let users retrieve $\overline{x_S} = \mathsf{Enc}(x_S)$ using private information retrieval (PIR) [107, 108]. Using PIR alone still reveals key-usage patterns. To avoid this, retrieval can be made via an anonymous channel. Then, users randomise $\overline{x_S}$ similarly to our construction, and the TS decrypts the ciphertext to recover $\overline{x_S} + \delta$, which it then uses in the TCP. To enable blocking of keys, the TS needs to frequently refresh its encryption keys, effectively invalidating previously retrieved ciphertexts $\overline{x_S}$. This simple protocol, however, has serious drawbacks. First, blocking is only enforced upon key refreshing, thus the timespan when compromised keys can be used depends on the refreshing schedule of the TS. Second, because the encryption of x_S for the current period can be randomised as often as the user wants (and the use of PIR precludes record-keeping), this scheme cannot provide rate-limiting.

5.6 Securing ABCs with Tandem

In this section we show how Tandem can be used to strengthen the security of the secret key when attribute-based credentials are stored on insecure platforms such as smart phones as in the case of IRMA (see Section 5.2.1). This ensures that valuable credentials cannot be abused, and can be blocked, while preserving users' privacy towards the Tandem server.

To use ABCs with Tandem, we need to convert the protocols for issuing and verifying credentials into threshold-cryptographic alternatives that are secure, private, and linearly randomisable. During issuance, the issuer (taking the place of the service provider in Section 5.3) provides the user with a credential bound to her secret key. The issuer does not learn the user's secret key. During verification, a user authenticates to a service provider by selectively disclosing attributes from her ABCs. In typical ABC schemes, these two protocols rely heavily on zero-knowledge proofs over the user's secret key. We show how these non-threshold protocols for Idemix credentials [15, 18] can be converted to threshold-cryptographic versions suitable for Tandem in 5.6.2 respectively.

Attribute-based credentials contain the user's secret key x as an attribute. For simplicity, we describe the Tandem-Idemix issuance and showing protocols with one other attribute: an issuer-determined attribute a_1. It is straightforward to generalise it to any number of attributes. To obtain a credential, the user (and the the Tandem server) runs a TCP version of the issuance protocol with the issuer. The issuance protocol is run jointly by the user, TS, and an issuer. Let $\tilde{x}_U$ and $\tilde{x}_S$ be the two shares of the user's secret key $x = \tilde{x}_U + \tilde{x}_S$ that are held by the user and the TS respectively after running $\mathsf{GenShares}$. The user first commits to her secret key x; as we share x between the user and the TS, they both have to participate in creating the commitment and in proving the knowledge of the secret key. The user combines her proof over $\tilde{x}_U$ and the TS' proof over $\tilde{x}_S$ before sending it to the issuer. If the proof created by the user-TS combination verifies correctly, then the issuer blindly signs the user's commitment to the secret x and the attribute a_1 and issues it as a credential to the user. In the showing protocol, the user proves the possession of a credential to a service provider to get access to a service or a resource. Again we convert the showing protocol into a TCP that uses the Tandem server. The distributed proof created by the user-TS combination during the issuance and showing protocols is similar to the proof of knowledge shown in Figure 5.1.

Security and privacy of the TCPs. These TCPs satisfy the TCP security and privacy notions defined in Section 5.4.2. For security, note that the TS computes zero-knowledge proofs of knowing $\tilde{x}_S$. A malicious user learns nothing about $\tilde{x}_S$ (and therefore x_S) as a result of the zero-knowledge property. Hence, the TCP security property holds for the TCP version of the showing and issuance protocol.

For privacy, the TS operates on a fully randomised key $\tilde{x}_S$, so the TS cannot distinguish users based on the key. The unlinkability property of the credential scheme ensures that the SP cannot distinguish users based on the resulting showing proof either. Hence, the TCP version of the showing protocol satisfies the TCP privacy property.

5.6.1 Rate-limiting in ABCs

Anonymous users can use the cover of privacy to misbehave, negatively impacting the system. ABC systems are not exempt from such misbehavior. Suppose, for example, that a user shares her "I am older than 18" credential with many under-aged users who do not hold such a credential. Then, those under-aged users can incorrectly convince service providers that they are over 18 years of age. If this happens often, service providers can no longer rely on these credentials to verify that a user is older than 18.

To limit such misbehavior, ABCs could benefit from rate-limiting. One method to limit abuse is to rate-limit credentials by ensuring that they can only be used a limited number of times. For instance, solutions such as n-times anonymous credentials [86] use custom cryptographic techniques to construct a special type of ABC that can be used only a limited number of times.

Tandem can achieve a similar type of rate-limiting *without* modifying the underlying cryptographic construction of ABCs. To rate-limit use of a system, the TS enforces a per-user and per-epoch limit q on the number of tokens it issues per user and per epoch. As a result, a credential cannot be shown more than q times per epoch. In fact, this approach limits *all* credentials associated to a user's key. If desired, Tandem can equally be applied on a credential basis.

This rate-limiting strategy *requires* that all users use Tandem. However, recall that the SPs (issuers and verifiers) cannot detect the use of Tandem, allowing users to forego sharing their keys with the TS, thus avoiding the rate limit. To allow the TS to enforce a rate-limit on all credentials, issuers must only issue credentials on keys that are shared with the TS.

A small change to the threshold-cryptographic version of the issuance protocol enables the issuer to confirm that users do use Tandem. To signal its involvement, the TS signs its proof (commitment, challenge, response) and sends the signature σ_{TS} to the user. The user forwards the TS' commitment, response and σ_{TS} from the TS to the issuer together with its own proof. The issuer, rather than the user, combines the proofs and verifies them. Moreover, the issuer checks the signature σ_{TS}. If the signature and proofs are correct, then the issuer is convinced that user's key is shared with the TS and thus, it signs the credential. Service providers may

TS		User		Issuer
$\tilde{x}_S, R_0, n$		$\tilde{x}_U, n, S, R_0, R_1$		p, q, n, S, R_0, R_1
$\hat{x}_S \in_R \mathcal{R}$		$\hat{x}_U, \hat{v}' \in_R \mathcal{R}$		
$u_S \equiv_n R_0^{\hat{x}_S}$	$\xrightarrow{u_S}$	$u_U \equiv_n R_0^{\hat{x}_U}$		
		$\tilde{U} \equiv_n S^{\hat{v}'} \cdot u_U \cdot u_S$	$\xrightarrow{\tilde{U}}$	
	$\xleftarrow{c}$		$\xleftarrow{c}$	$c \in_R \mathcal{R}$
$r_S = \hat{x}_S + c \cdot \tilde{x}_S$	$\xrightarrow{r_S}$	$r_{v'} = \hat{v}' + c \cdot v'$		
		$r_U = \hat{x}_U + c \cdot \tilde{x}_U$		
		$r = r_U + r_S$	$\xrightarrow{r_{v'}, r}$	$\tilde{U} \overset{?}{\equiv}_n U^{-c} S^{r_{v'}} R_0^{r}$

Figure 5.3: Full details of the proof of knowledge of the user's commitment $U = S^{v'} R_0^{x_U} R_0^{x_S}$ in the Tandem Idemix issuance protocol. The randomness for the commitments is chosen from a large set of randomisers denoted by $\mathcal{R}$. We use a short notation $\equiv_n$ to denote $\bmod n$. The Tandem server only knows $\tilde{x}_S$ and the user knows $\tilde{x}_U$ and the randomness v' (recall $\tilde{x}_S$ and $\tilde{x}_U$ are the respective outputs of the GenShares protocol). The TS effectively creates a zero-knowledge proof of knowing $\tilde{x}_S$. We note that this proof is just the relevant excerpt from the issuance protocol and for simplicity we have depicted it as an interactive proof of knowledge whereas the original proof in Idemix (Algorithm 2) is non-interactive.

follow the same procedure during verification to ensure that the user's key is shared with the TS or just trust the issuer to have checked this before issuing the credentials. In the latter case, the user combines her and the TS' proofs and sends a single proof to the service provider as shown in Figure 5.1.

5.6.2 Applying Tandem to Idemix credentials

Identity Mixer or in short, Idemix is an anonymous credential system with a rich feature set [15]. As we mentioned in Section 5.2.1, the IRMA identity platform implements Idemix ABCs and their most important functions: credential issuance and selective disclosure of attributes. As our goal is to secure users' secret keys that are bound to these credentials with the help of Tandem, we show how to use Tandem with Idemix ABCs.

Recall that the Idemix ABCs are built from Camenisch–Lysyanskaya (CL) signatures [18]. The CL signature scheme is described in Section 2.2 of the Preliminaries chapter. CL signatures work in the quadratic residue subgroup QR_n whose order[10] is unknown to everyone except the signer herself who has generated the group. In an Idemix credential system with two attributes x and a_1, the public key of the issuer pk_I is denoted by (n, Z, S, R_0, R_1). The private key sk_I consists of the (safe) prime factors p, q of n.

Obtaining a credential. The Idemix credential issuance protocol in which the

[10] The order of the QR_n group is given by $|QR_n| = \frac{(p-1)(q-1)}{4}$ where p, q are the (safe) prime factors of the RSA modulus n.

complete secret key is possessed by the user is described in Section 2.3. To use Tandem, we convert this protocol into a threshold protocol where the key is shared between the user and the Tandem server. Now the secret key x is comprised of key shares $\tilde{x}_U$ and $\tilde{x}_S$ (output of $\mathsf{GenShares}$) such that $x = \tilde{x}_U + \tilde{x}_S$. Tandem directly affects a part of the issuing protocol: computation of the commitment U and the proof of correctness for U. As the first step in this protocol, the user and the TS create a commitment

$$\begin{aligned} U &= S^{v'} R_0^{\tilde{x}_U} R_0^{\tilde{x}_S} \mod n \\ &= S^{v'} R_0^{\tilde{x}_U + \tilde{x}_S} \mod n \\ &= S^{v'} R_0^{x} \mod n \end{aligned}$$

To prove to the issuer that U is well-formed, the user and the TS construct the proof

$$PK\{(x, v') : U = S^{v'} R_0^x \mod n\}. \tag{5.2}$$

When the secret key is distributed between the user and the TS, the above proof is constructed as shown in Figure 5.3. In this proof, the TS proves its knowledge of $\tilde{x}_S$ and the user proves her knowledge of the key share $\tilde{x}_U$ and the randomiser v'. We note that unlike zero-knowledge proofs in the known-order groups, the responses calculated in the Idemix proofs do not undergo modular reduction (i.e. no $\mod p$ is applied) because the group order is unknown to the user and the TS. If the proof for the correctness of U verifies successfully, the issuer goes ahead with signing and issuing the credential to the user as described in Algorithm 4. The signature is computed as follows:

$$A = \left(\frac{Z}{S^v R_0^x R_1^{a_1}}\right)^{1/e} \pmod n$$

The final credential is denoted by $\sigma = (A, e, v)$.

Showing a credential. The showing protocol for an Idemix credential is described in Section 2.4. Let us consider an example where the user wishes to disclose an attribute a_1 from her Idemix credential to a verifier while hiding her secret key x. First, the user randomises the issuer's signature and then creates a zero-knowledge proof as shown in Algorithm 6 to prove the possession of a credential $\sigma = (A, e, v)$ over her key x and the attribute a_1. The proof is denoted as follows.

$$PK\{(e, v, x) : Z = A'^e \cdot S^v \cdot \mathbf{R_0^x} \cdot R_1^{a_1} \mod n\}$$

In the threshold version of the showing protocol, the user and the TS need to compute the above proof together. We can see that the user can easily generate all the factors in the proof except for the third factor R_0^x. This is because, only the third factor contains the user's secret key x of which the user only has a secret share. Thus, the user has to contact the TS to construct this part of the proof. This proof is just like in equation (5.2), albeit a bit more complex. As a result, a very similar construction as in Figure 5.3 allows the user and the TS to jointly compute this proof. During both issuance and showing of a credential, the TS will have to cooperate with the user by creating a Schnorr-like zero-knowledge proof over its key-share $\tilde{x}_S$. As suggested in Section 5.6.1, if TS signs its proof during the issuance to signal its involvement in the credential creation to the issuer, then the TS can distinguish between issuance and showing protocols. Otherwise, the TS cannot tell if it is participating with a user in an issuance or a showing of a credential.

Now we have seen that creating a threshold version of Idemix issuance and showing protocols is possible. But there is an important difference in the group setting for which Tandem was designed and that of Idemix. For simplicity, Tandem was designed for groups with known order, that is, all users in the system know the order of the group in which Tandem operates. In the context of Idemix credentials however, the order of the group is unknown to users. Only an issuer (signer of credentials) knows the prime factors of the RSA modulus n and hence the order of quadratic residues group in which Idemix operates. In this case, the secret x can be calculated without modular reduction as follows:

$$x = x_S + x_U = \tilde{x}_S + \tilde{x}_U \tag{5.3}$$

where $\tilde{x}_S = x_S + \delta$ and $\tilde{x}_U = x_U - \delta$.

To use Tandem with Idemix, we need to find out how the inability of users to perform modular reduction affects the Tandem protocols (e.g. GenShares). Consider an Idemix setting in which the long-term key-shares x_S and x_U are of the same length and δ is bigger than both shares. When the user derives her fresh key-share as $\tilde{x}_U = x_U - \delta$ during GenShares protocol, it results in a negative $\tilde{x}_U$. Then equality in eqn. (5.3) does not hold because

$$\tilde{x}_S + (-\tilde{x}_U) = (x_S + \delta) - (x_U - \delta) = x_S - x_U + 2\delta \neq x_S + x_U \neq x$$

For Tandem to work as intended, eqn. 5.3 must be satisfied. To satisfy this equation, the key-share $\tilde{x}_U$ must always be positive. For this, we must ensure that the user's long-term key-share x_U is bigger than the randomiser δ. But we also have to keep in mind that the TS' key-share x_S must always be (at least ℓ bits) smaller than δ. This is to ensure that x_S is statistically hidden in $\tilde{x}_S$ so that the TS cannot identify the user during the threshold protocol based on the value of $\tilde{x}_S$.

Furthermore, there is another place where the unknown group order makes a difference: the distributed zero-knowledge proof (e.g. proof for the user's commitment as shown in the Fig. 5.3). The response computation step in Idemix distributed proofs do not involve modular reduction. In spite of this, the responses created by the TS and the user must statistically hide the secret shares $\tilde{x}_S$ and $\tilde{x}_U$ respectively. So the size of the random numbers used to commit to $\tilde{x}_S$ and $\tilde{x}_U$, denoted by $\hat{x}_S$ and $\hat{x}_U$ respectively in Figure 5.3 should be chosen such that they statistically hide the key shares in the response computation step.

In sum, by choosing the sizes of the long-term key shares and randomness for commitments appropriately in the distributed proofs of knowledge, we have shown that Tandem can easily be adapted to Idemix.

5.7 Related Work

Exisiting solutions to protect cryptographic keys fall in two coarse categories, either *single-party* or *decentralized*. The former typically relies on secure hardware [109, 74, 73, 72]. Cryptographic keys are stored and processed securely within the secure environment and can never leave. However, secure hardware is expensive, not widely available, and is often not flexible enough to run advanced protocols.

Decentralised solutions, on the other hand, distribute the user's secret key among several parties using threshold cryptography. This approach was first proposed by Desmedt [110] and Boyd [111]. Several threshold encryption and signature schemes have been proposed since then [112, 113, 114, 87, 88, 115, 116, 117]. More recently, Atwater et al. [118] built a library to execute such protocols in users' personal devices. Other works have tackled more complicated protocols. For instance, Brands shows how to distribute the user's secret key in attribute-based credentials [16], and Keller et al. [119] show how to make threshold cryptographic versions of zero-knowledge proofs.

Many works propose systems in which the user's secret key is shared between a user's device and a central server to protect the key and also to enable instant blocking of the key by the user [120, 121, 122, 123, 124, 125]. However, none of these schemes provide privacy for the user towards the central server. In all of these schemes users authenticate themselves to the server, making them identifiable to the server based on their key shares stored on the server. This in turn makes users susceptible to time-correlation attacks [82]. Camenisch et al. [121] attempt to ensure privacy to some extent in signature schemes by blinding the message being signed during the threshold protocol with the server. Yet, the server learns when and how often the user uses her signing key. Hence, users are still vulnerable to timing attacks. The scheme by Brands [16] protects against these attacks as long as the key-share holder is a smartcard, which cannot store a timed log of operations. However, if the smartcard is replaced by an online server that holds the key share, this server learns the key-usage patterns of users. Then, the cryptographic measures proposed by Brands to hide the messages seen by the TS and the service provider cannot prevent time-correlation attacks.

TANDEM is designed to complement these solutions to make them privacy friendly. It allows users to anonymously access their key-shares at the TS via pre-obtained one-time-use tokens when they need to run threshold protocols. So the TS cannot identify users based on their key-shares during TCPs, log their key-usage patterns and deanonymise users by colluding with service providers.

5.8 Conclusion

Protecting cryptographic keys is imperative to maintain the security of cryptographic protocols. As users' devices are most of the time insecure, the community has turned to threshold-cryptographic protocols to strengthen the security of keys. When run with a central server, however, these protocols raise privacy concerns. This is because a malicious server can collude with a service provider and use the time-correlation between the access and the use of the key to identify users involved in anonymous transactions (e.g. showing an anonymous credential to a service provider). In this chapter, we have proposed Tandem, a provably secure and practical scheme that when composed with threshold-cryptographic protocols, provides privacy-preserving access to the key shares stored at the server. Furthermore, it adds capabilities to block and rate-limit key usage. The proof-of-concept implementation of Tandem developed by one of the co-authors shows that for reasonable security parameters Tandem's protocols run in less than 60 ms.

Tandem is particularly suited for privacy-friendly applications such as attribute-based credentials because it retains their inherent privacy properties. Yet, Tandem can be used to strengthen a wide variety of primitives, including signature and encryption schemes, as long as they can be transformed into linearly-randomisable threshold protocols. For the case of ABCs, we have shown that deriving such a threshold protocol can be done with standard techniques, and that adding Tandem is straightforward.

Chapter 6

Secure Self-Enrolment

6.1 Introduction

In the current digital world, there are increasinlgly many scenarios where users authenticate to access web services. Users need to use various authentication mechanisms and manage a wide array of credentials for this purpose. The personally identifying information (PII) of users contained in or associated with some credentials are more trusted than others. For instance, national identity documents, bank credentials are more trusted than Google or Facebook credentials. The reason for the difference in the level of trust is the way the credentials are bound to the real identity of users. This binding is a result of an enrolment process that includes identity proofing the user before the issuance of an identity credential.

Identity proofing is the procedure to verify the identity of individuals applying for the issuance of electronic authentication means [126, 127]. In this chapter, we consider the means to be the credentials that are used to authenticate users to online service providers. The identity-proofing procedures are designed based on the required assurance level of the authentication means, to ensure that the issuer or the enroller knows the true identity of the applicant. Specifically, the requirements include measures to ensure that:

- A person with the applicant's claimed attributes exists, and those attributes are sufficient to uniquely identify a single person;
- It is difficult for a user to later repudiate the enrolment and dispute an authentication using an issued credential.

An applicant may appear in person for identity proofing as in the case of face-to-face enrolments or may provide an identity proof remotely.

For example, face-to-face enrolment is common for the issuance of passports or national electronic identification (eID) documents. That is, an individual applying for an eID document undergoes an in-person identity proofing at a trusted enroller's physical location (e.g. municipality office). The enroller collects the applicant's identity information and verifies the real identity of the user along with some existing ID proof (e.g. birth certificate). Upon successful identity proofing, the enroller sends the identity-proofed PII of the user to the issuer (e.g. passport issuing authority) who then issues the eID document that contains the PII sent by the enroller. The eID can be used by the user to authenticate herself in several online or offline authentication scenarios; for instance, to file tax online or before boarding a flight at an airport. Note that there is a trust relationship between the enroller and the issuer.

Attribute-based credentials (ABCs) are a type of privacy-friendly credentials which contain personal attributes (e.g. name, age, nationality) of a user that constitute her identity. A user can obtain signed ABCs from certified issuers and use them later to authenticate at service providers by selectively disclosing attributes from the credentials based on the context. However, issuance without identity-proofing is vulnerable to impersonation attacks. For example, if a user's real identity is not verified at or right before issuance, a malicious user Eve can impersonate an honest user Alice and get Alice's credentials issued to herself from an issuer. Then the issuer cannot tell if the user claiming to be Alice is actually Alice or Eve. Thus, the ABCs resulting from such an issuance cannot be trusted by authenticating entities. A possible way to instil trust in ABCs is to issue them after identity-proofing users in a face-to-face setting as in the case of eID documents. Although this can be reasonable in some cases, including student credentials at a university, in other cases, alternative solutions may be beneficial in terms of expenses or usability. In this chapter, these alternative ways to issue credentials are explored and compared. We emphasise that our aim is to establish high levels of trust for specific ABCs that contain uniquely identifying attributes (PII) of users.

Our basic assumption is that users already possess some trusted electronic authentication means i.e. credentials, which make it possible for them to access online services. Although these credentials may be relevant only in a restricted domain, such as government services or banking, they can often be considered trusted in a broader context. A national identification document (such as the Dutch DigID[1], the German eID[2] or the Estonian eID[3]), for instance, provides identification of citizens for many online services. A bank card, a SIM card with a subscription at a mobile network operator (MNO) or a university identity card can sometimes be used for online identification.

We propose to use these pre-existing authentication credentials of users as *trust anchors* in the self-enrolment of users in ABC systems. The eIDAS regulation uses the term 'authoritative source' for a trust anchor. It refers to any source irrespective of its form that can be relied upon to provide accurate data, information and/or evidence that can be used to prove identity [128]. We define secure self-enrolment (SSE) in the context of ABCs as an issuance that involves out-of-band verification of some trust anchor which can be carried out by users remotely (e.g. sitting at home) through some web platform. The out-of-band verification provides the identity proof for the user based on which she can receive a new attribute-based credential. With

[1] https://www.digid.nl/en/

[2] https://www.bsi.bund.de/EN/Topics/ElectrIDDocuments/German-eID/german-eID_node.html

[3] https://e-estonia.com/estonian-eid-card-entering-the-contactless-world/

such an enrolment, the PII contained in or associated to different trust anchors can be collected by users remotely and placed into their ABC wallets in a trustworthy and a convenient manner.

Specifically, during an SSE process, an online enroller collects PII and verifies the real identity of an enrolling user (or identity-proofs[4] a user) on the basis of a trust anchor. Upon successful identity-proofing, the enroller sends a confirmation message to an issuer via the user. Subsequently, the issuer issues an ABC consisting of the attributes that are derived from the trust anchor to the user. The enroller and issuer may be the same entity or separate entities with a trust relationship between them. The result of SSE are ABCs with personal attributes of the user such as name, date of birth, address, which are bound to her real identity.

Let us see an example of an SSE where a bank is the enroller and Alice is the user. We assume that the bank already has Alice's PII such as name, date of birth and address in its records. Alice first visits a bank's website, proves her identity by logging in with her bank credential. If the verification succeeds, she receives an ABC with her PII as attributes from the bank itself or from some other issuer to whom the bank sends a confirmation message along with the identity-proofed user's attributes. In this example, we notice that a bank can either (i) assume the roles of both enroller and issuer, or (ii) identity-proof Alice as the enroller with existing authentication mechanisms and delegate the issuance of ABC to a proxy issuer (e.g. an authority solely responsible for the issuance of ABCs). The second approach minimises the setup required at the bank to support SSE and thus maybe preferred over the first approach.

The trust for the issued ABCs is derived from the identity binding of the user with the trust anchor that is used during enrolment. An SSE protocol with an eID document or a bank account as the trust anchor is trustworthy, as it is built on top of an earlier face-to-face enrolment that was carried out for the issuance of the respective trust anchor. We note that a self-enrolment protocol with a Facebook profile as the trust anchor is less trustworthy as the user's identity information associated with it are self-asserted by the user and not verified by the social network against the real identity of the user. As we will see in this chapter, with SSE, users can get ABCs of various trust levels depending on the trust anchor(s) and the quality of identity proofing done at enrollers.

There are numerous advantages of SSE. First of all, the SSE methods enable users to leverage their personal information present in existing trust anchors to obtain trustworthy ABCs. Then later authentication instances, carried out by using the ABC, does not involve the original enroller or the issuer. For instance, a bank or Facebook does not know when and where a user uses information from their databases. Second, the attributes from the trusted identity providers (or issuers) can be used with all the benefits of ABCs, including selective disclosure and possible untraceability. Third, verifiers get reliable identifying attributes of users without having to communicate directly with the identity providers. Finally, in comparison with face-to-face enrolments, SSE methods are more convenient for users as they can

[4] The verb 'identity-proofs' is used in reference to an authority who verifies the true identity of an applicant. The verb 'proofs' is different from 'proves'. For example, an applicant 'proves' her identity to an enroller and the enroller identity-proofs the applicant.

be carried out remotely from any location and are inexpensive for the authorities (i.e. enrollers and issuers) in terms of time, costs and resources.

6.1.1 Secure Self-Enrolment in IRMA

The IRMA project has created an identity platform for using attribute-based credentials for authentication and signing via smart phones [37]. In the IRMA system, there are users who own ABCs, issuers who issue ABCs to users and service providers (or verifiers) who verify the ABCs presented by the users during an authentication protocol. The smartphone application called the *IRMA app*, is an ABC wallet where a user can collect her identity credentials from different domains (e.g. bank, university, government) and use them later to authenticate online to any service provider. However, the attributes in the credentials are only as trustworthy as their issuance process. That is why a user must undergo secure enrolment before getting credentials issued to her IRMA app.

Smart phones are enablers for secure self-enrolments as they can directly connect with enrollers online and can contain capabilities to communicate with eID documents (e.g. via Near Field Communication). As IRMA uses smart phones as the carrier of users' credentials, in principle, SSE methods are feasible for enrolling IRMA users.

A high-level picture of our approach to IRMA secure self-enrolment is given in Figure 6.1. The main entities in SSE are

- *User* - Entity who possesses a trust anchor (e.g. eID document, bank credential, personal mobile subscription) and initiates self-enrolment on her smart phone (via her IRMA app) to get authentic attribute-based credential(s) on her phone;
- *Enroller* - Entity who identity-proofs the user by verifying the identity data provided by the User against the PII present in the User's trust anchor before an ABC can be issued to her;
- *Issuer* - Entity who issues the PII derived from the trust anchor as an ABC(s) to the User's IRMA app upon getting an enrolment-confirmation message from the Enroller.

Note that the Issuer trusts the Enroller to have reliably verified the user's identity based on the trust anchor and derived the user's personal attributes from the trust anchor. In specific scenarios the Issuer and the Enroller are the same entity. In Figure 6.1, the steps 2 and 3 constitute the enrolment phase, which the User undergoes for obtaining attribute-based credentials on her IRMA app. With the issuance of ABCs, enrolment is completed. The last 'show' step suggests that the ABCs can be used to authenticate to multiple service providers. This chapter focuses on the enrolment steps, and will especially elaborate the user's interaction with an enroller and an issuer in the coming sections.

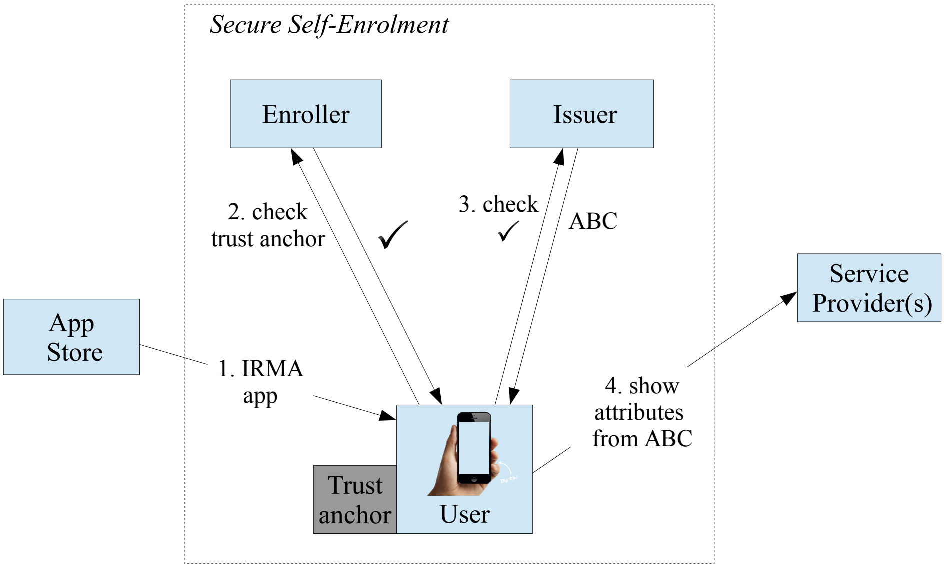


Figure 6.1: Secure self-enrolment of IRMA users with a trust anchor (e.g. a standard eID, a valid bank account, a valid SIM subscription) to get an attribute-based credential on the phone.

6.1.2 Our contribution

In this chapter, we explore several forms of secure self-enrolment which allow an ABC user to collect her personal attributes from several trusted sources (trust anchors) and place them securely in her ABC wallet i.e. the IRMA app. With SSE, users can get ABCs of various trust levels depending on the type of trust anchor and the identity proofing done at the enroller. We focus on enrolling users to get ABCs with identifying attributes such as name, date of birth, address.

First, we describe three methods for achieving secure self-enrolment: (1) eID-based SSE (Section 6.2), (2) Bank-account-based SSE (Section 6.3), and (3) SIM-subscription-based SSE (Section 6.4). Then we describe how we can combine SSE methods to elevate the assurance provided by the issued credentials in Section 6.5. Furthermore, we discuss the implementation of specific enrolment methods in IRMA in Section 6.6.

We envision that bootstrapping IRMA with SSE will increase the degree of confidence of the authenticating entities (service providers) to accept ABCs for user authentication in their use cases. This could lead to widespread use of this attribute-based technology in real-world applications. The conclusion that emerges from our research on this topic is:

1. Secure self-enrolment adds the much needed trust-dimension to the attribute-based authentication technology that allows users to collect identity attributes from various sources and use them at different authentication scenarios in a privacy-friendly manner;
2. Several methods exist or are appearing that make SSE a viable new approach in electronic identity management;
3. SSE requires more than a single protocol, and can be realised by combining several self-enrolment protocols. If they yield consistent outcomes, then the credentials resulting from a combination of enrolments may result in higher levels of trust than the ones from a single enrolment. In this way, the separate protocols reinforce each other.

6.2 eID-based SSE

Before describing the enrolment method that uses eID documents as the trust anchor, we give some background information on the type of eID documents that are considered for SSE.

6.2.1 Brief background on eID documents

Many countries issue electronic passports (or eIDs) to their citizens. These eIDs are physical identity documents with embedded chips that contain the eID holders' identity information such as name, date of birth, date of issuance etc., and biometric data such as photo, fingerprints. The chips can be accessed wirelessly (typically via NFC). Access to the fingerprints stored on the chips is restricted, but the other data can be accessed without prior authorisation [129]. The data in the eID are digitally signed, so that their integrity and authenticity can be checked. In general, there are standards that cover confidentiality, integrity and authenticity of the eID data. The security of an eID document must conform to international (public) standards issued by the International Civil Aviation Organization (ICAO). ICAO runs a 'machine readable travel documents' programme whose main purpose is to develop and maintain open specifications for automated access to data in eIDs. The physical characteristics of an e-passport and its machine readable zone (MRZ) are described in the ICAO specification [130]. In the current context, two ICAO protocols are of special importance [131].

- *Basic Access Control (BAC).* The user data in the embedded chip in a passport are cryptographically protected. The required cryptographic keys can be derived from the combination of the document number, date of birth and date of passport expiry. These can be obtained by scanning the machine readable zone at the bottom of the main passport page. They can also be provided manually, like in the screenshot on the left in Figure 6.7. The protocol that derives the relevant keys and uses them for data transfer is called Basic Access

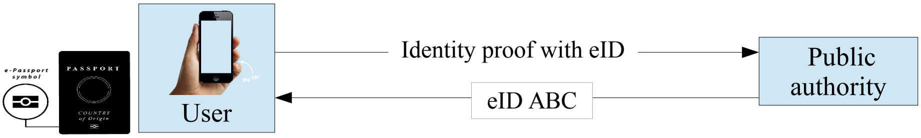


Figure 6.2: High level picture of eID-based self-enrolment where a public authority remotely identity-proofs a user based on her eID and issues an ABC consisting of eID attributes to the user's IRMA app.

Control. It is implemented in any device that reads e-passports.

- *Active Authentication (AA).* The data that are read via BAC includes a document-specific public key. The associated private key is securely stored inside the chip in the passport. The so-called Active Authentication protocol uses this key pair to verify the authenticity of the passport via a standard challenge-response check to ensure that the passport is not a clone.

The standard eID documents that implement the above protocols (e.g. electronic passports, Dutch identity card) are used for the eID-based SSE within the IRMA system.

6.2.2 eID-based SSE protocol description

The SSE protocol based on ICAO-standard eID documents allows a user to enrol remotely (from any location) through her smart phone using her eID document to get an authentic *eID ABC* on her smart phone. Public authorities affiliated to a national government are the traditional trusted root for identity information about citizens. They remain so in our SSE protocol, via the eID documents that they issue. We consider an authority (e.g. a municipality) that is trusted by the government to take up the role of an enroller and a public authority (similar to an authority that issues passports) to be the ABC issuer. Figure 6.2 gives a simplified illustration of eID-based self-enrolment where a public authority performs as both the enroller and the issuer of ABC.

First, we will consider a basic instance of eID-based SSE protocol. For this, we assume that the user's eID document has a Near Field Communication (NFC) chip and that the user's smart phone supports NFC, so that the eID document can be read by the phone. Below we describe the protocol that summarizes the communication between the User, Enroller and Issuer during eID-based SSE protocol. See also Figure 6.3 for technical details of this communication.

1. The User connects to the Enroller securely (e.g. via TLS) through her phone, requests for enrolment using her eID document. She enters the BAC (see Section 6.2.1) data present in the eID: document number, date of birth and document expiry date on her phone, and holds her eID document against (the

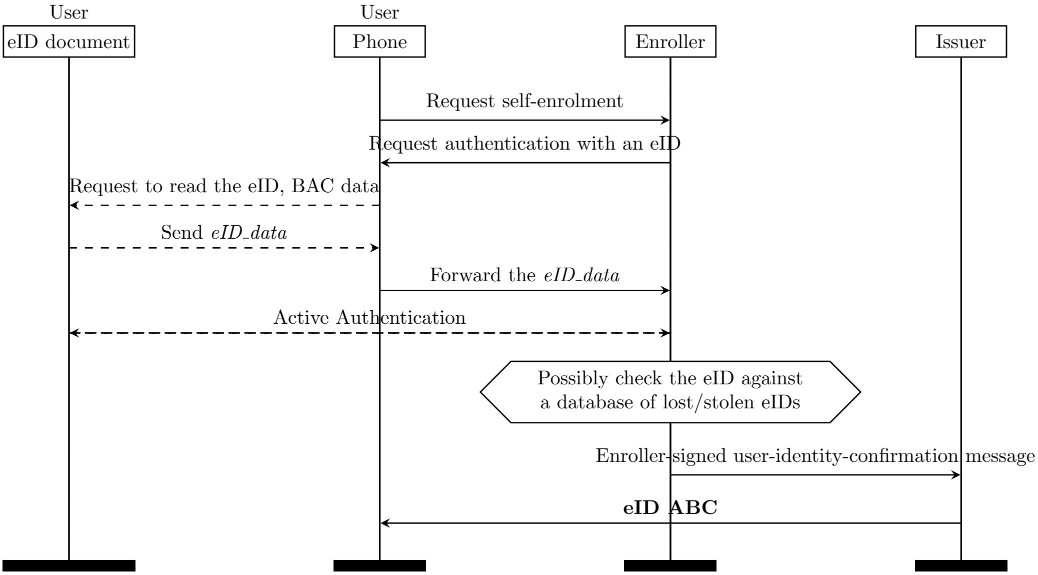

User
eID document
User
Phone
Enroller
Issuer
Request self-enrolment
Request authentication with an eID
Request to read the eID, BAC data
Send *eID_data*
Forward the *eID_data*
Active Authentication
Possibly check the eID against a database of lost/stolen eIDs
Enroller-signed user-identity-confirmation message
eID ABC

Figure 6.3: Basic secure self-enrolment protocol using a standard eID document. The dashed arrows in the figure indicate the phone as a NFC reader and the solid arrows indicate the phone as the currently enrolling user-device (potential credential carrier).

NFC reader of) her phone.

2. The Enroller reads the User's eID and performs the following checks on the eID:
   - Integrity: by verifying the digital signature on the (hashes of the) data groups, the Enroller verifies if the eID data are not altered;
   - Authenticity: by performing the ICAO-defined active authentication (see Section 6.2.1), the Enroller verifies if the presented eID is authentic or cloned;
   - Validity: by checking that (i) the eID has not expired and, (ii) the eID has not been revoked by matching it against a database of revoked (e.g. lost/stolen) eID documents. The latter check is possible only if the Enroller has access to such a database, which is typically maintained by public authorities.
3. If the above eID checks are successful, then the Enroller sends a digitally signed user-identity confirmation message to the Issuer. This message contains the User's eID data (e.g. name, date of birth, document number) that the Issuer can sign and issue to the User as an *eID ABC*.
4. The Issuer verifies the Enroller's signature on the confirmation message, connects to the User's phone and issues an ABC with eID data.

The protocol above considers the User's eID document as a *trust anchor* (or root of trust) from which the Enroller derives the user's identity credentials and the Issuer issues them securely to the User's phone. After this issuance, the User can use her phone as her authenticating device and authenticate to any entity with the eID ABC. The SSE protocol, although remotely done, ensures that the user is in possession of some valid eID and that the identity credentials are derived onto her phone from that authentic source.

Weakness of basic self-enrolment Although the basic instance of eID-based SSE protocol is user-friendly, inexpensive for both User and the Enroller, and results in an authentic eID ABC on the user's smart phone, it has an important weakness: a malicious user might use someone else's eID document (stolen, lost or borrowed) and carry out the protocol. This would lead to the malicious user wrongfully getting the eID owner's attributes issued to her phone's IRMA app as her identity attributes. From then onward, the user can impersonate the eID-owner during online authentications with her phone. This attack is possible because there is little binding between the user and the identity document (i.e. the trust anchor) that is used for the enrolment.

We will address this weakness by including some additional checks to ensure user-eID binding in the following subsections. The enhanced eID-based SSE protocol provides strong assurance about the user's identity that is almost equivalent to the assurance provided by a face-to-face enrolment. With such an SSE, a user can get a trusted eID ABC. This ABC is considered as the electronic counterpart of the user's identity credential stored on her eID document. Thus, this ABC can be used as the root of trust for enrolments that precede the issuance of other credentials in different domains. The eID ABC can also be used as a trusted credential for authenticating the user to various service providers.

6.2.3 Biometric check during eID-based SSE

A solution to achieve user-eID binding is to include a biometric check in the basic eID-based SSE. ICAO-standard eID documents digitally store a photo of the user and optionally fingerprints as well. Since fingerprints can usually be read only by a few authorized entities, we focus on biometric face verification.

Here we describe when and how the Enroller can verify the biometric aspect of the user to ensure she is indeed the legitimate owner of the presented eID document. During the SSE protocol, the Enroller performs some checks on the User's eID document to ensure its validity, authenticity and data integrity. If these eID checks are successful, the Enroller reads and stores the eID data that includes the User's identity data and her photo. Next, the Enroller requests the User to present a biometric evidence in the form of a live video or some other form of face recognition. Then the Enroller matches the User's photo from the eID to the biometric evidence. If there is a match, then the Enroller proceeds to issue a signed confirmation, which guarantees the Issuer that the eID document is bound to the user and that her identity has been checked by the Enroller. Finally, the Issuer signs the user's eID data (attributes) and issues them as an attribute-based credential to the user's phone.

We consider a method that can be used by the Enroller to carry out verification of the user's face during SSE: video legitimation with a human verifier. The video legitimation method is implemented by digital identification solutions such as, IDnow[5] and WebID[6]. This method involves a human verifier (e.g., an employee of the Enroller's organisation) who performs the task of identity-proofing users over video calls. Regardless of the physical separation, sensory perception of the users is possible, since the user to be identity-proofed and the employee sit opposite one another "face-to-face" through this video transmission and communicate with one another. The user is asked to hold both the front and rear sides of a valid official identity card or passport (eID) in front of the webcam. The eID document must be tilted several times and moved so that the hologram and further security features can be checked by the verifier. The document number is also recorded and photos are taken to secure the evidence. This confirms the user's ownership of the eID document. Additionally, a unique transaction number is sent to the user by e-mail or text message to check if the user has stated correct electronic address or telephone number in the enrolment application. All these checks constitute the video-legitimation procedure. This procedure has been examined and approved by BMF (German Federal Ministry of Finance) and BaFin (German Federal Financial Supervisory Authority) [132]. The basis for this type of legitimation is the new interpretation of Section §6 (Specifically, requirement number 2 of the GwG ("not personally present")) of the Anti-Money-Laundering Act [133]. A major bank ING-DiBa in Germany supports WebID's video legitimation to simplify its account opening process for its customers[7]. This allows new bank customers to identify themselves via video when opening an account.

Due to its real-time biometric checking capability involving the User and the Enroller's employee, video legitimation is a viable solution to verify if the enrolling user and the owner of the presented eID document are the same during an eID-based SSE. Similarly to the face-to-face identity proofing, the Enroller (employee) compares the photo that is digitally read from the eID document to the person's face that appears in a real time video session during enrolment. The result of the biometric check with a human verifier is reliable and convincing. Moreover, this biometric check adds a 'something you are' factor (face) in addition to the 'something you have' factor (eID document) to the user-authentication by the Enroller during an SSE protocol. However, the video addition will be costly, since it requires additional personnel. It might also increase the duration of each enrolment, especially when it leads to queues for the video procedure. Despite increased time and cost for an enrolment, biometric check over a video ensures that the self-enrolment provides similar security and trust guarantees to a face-to-face enrolment. We will call the eID-based self-enrolment that includes the biometric check of users as *enhanced eID-based enrolment.*

[5] `https://www.idnow.eu/`

[6] `https://www.webid-solutions.de/en/`

[7] `https://www.ing.com/Newsroom/All-news/NW/ING-Germany-simplifies-account-opening-process-with-video-legitimation.htm`

6.2.4 PIN check during eID-based SSE

Another solution to check the binding between an enrolling user and the presented eID during an SSE protocol is to rely on the eID PIN. Some eID documents such as, Common Access Cards (CACs) [134], German eID card [135], Italian ID card[8] support the use of PIN codes that are known only to their owners. Such PINs are typically delivered to eID owners via a separate channel, such as a (secure) PIN mailer. Users (i.e. eID card owners) will have to enter the PIN to authenticate online with the card.

If such eID documents are used in an eID-based SSE protocol, then the Enroller can easily verify if the enrolling user is the legitimate owner of the eID. The user proving the knowledge of the PIN provides 'something you know' factor (PIN) in addition to 'something you have' factor (eID) to the user authentication carried out by the Enroller during a self-enrolment. Including a PIN check during an eID-based SSE is probably be the simplest solution for checking if the enrolling user and the eID are bound to each other.

However, not all eIDs support the use of PINs. Thereby, the types of eIDs one can use for a secure self-enrolment becomes limited. Moreover, the trust assurance provided by the user PINs depends on how reliably and confidentially they have been generated in the first place and transported to the users. Further, it also depends on how securely users store and maintain their PINs.

6.2.5 Related work

On a conceptual level, the eID-based self-enrolment approach has some overlap with methods developed for deriving credentials from special U.S. identity cards such as, the Common Access Card (CAC) [134], but it also differs in several essential aspects.

A CAC is a Personal Identity Verification (PIV) card issued by the U.S. Department of Defense that is meant for closed user groups (e.g. employees of government agencies). An application that uses CAC as the mother card to derive PIV credentials on mobile devices is Entrust Mobile Derived Credential solution [136]. It requires the user to undergo a derived credential enrolment process which involves her PC (desktop or laptop) that is connected to a CAC via card reader, her mobile device and Entrust's Self Service Module (SSM). The enrolment takes place as follows.

1. The user navigates to the SSM's web page through a web browser on her PC and authenticates to the SSM using her PIV/CAC smartcard. The CAC gets activated by the user-specific PIN.
2. The SSM validates PIV credential on the card by checking PIV/CAC policy object identifier, authentication certificate, revocation status of the certicate. If checks are successful, the user can select the link to request a derived PIV credential.

[8] `https://www.gemalto.com/govt/customer-cases/new-national-identity-card-for-italy`

3. To ensure a secure user-device binding, the SSM uses 'QR code with password via an encrypted email' or 'Email with password via encrypted email' activation methods. The user either scans the QR code with her Entrust app on the mobile phone or clicks the link in the email and then she enters the password to activate the derived PIV credential issuance. The user can decrypt the password sent over the pre-registered email address only with PIV credentials found on the user's CAC smart card. The activation methods that display the password on the SSM screen instead of sending encrypted version to user's email do not achieve device-derived credential binding.

4. On entering the correct password, the user obtains the derived PIV credential from the SSM on her Entrust mobile app. The derived PIV credential is associated with a derived PIV authentication certificate. The derived credential may then be used for authentication to remote systems in the same way as the PIV authentication certificate on the CAC card is used.

In comparison with the enrolment process described above, our eID-based self-enrolment differs in the following ways. First, it can be used by any person who has some ICAO-standard electronic identification document (e-passport or a driver's licence) whereas Entrust's enrolment is available only to a closed group of government agency's employees who already have a CAC smartcard. Second, it supports eID documents without the PIN code activation. Biometric checks can be used to authenticate a user during SSE instead of PINs (see Section 6.2.3). Finally, it works in the context of attribute-based credentials (ABCs) instead of public-key certificates. However, the Entrust enrolment with CAC cards could possibly be used to derive PIV credentials in the form of ABCs on users' smart phones. This would provide the benefits of using ABCs for authentication to the CAC cardholders, namely, selective disclosure and possible unlinkability.

6.3 Bank-based SSE

In this section, we will see how bank accounts and associated bank credentials of users can serve as trust anchors and facilitate a secure self-enrolment. Just as eIDs, in many countries, individuals open bank accounts after undergoing a face-to-face enrolment at banks. So banks have reliable identity information of their customers in their databases. Our idea is to make use of this pre-existing relationship between individuals and their banks for secure self-enrolment.

For bank-based SSE, we assume (1) the Bank has performed a face-to-face enrolment of the user during the opening of the bank account; (2) the Bank has access to a database which contains authentic data of enrolling users and; (3) it uses an authentication method to authenticate the users when they request SSE.

We describe the bank-based SSE protocol below in which the Bank acts as Enroller and Issuer; see the high level description in Figure 6.4.

1. User requests SSE at the Bank's website through her phone;

2. Now the Bank requires the User to authenticate. A successful authentication

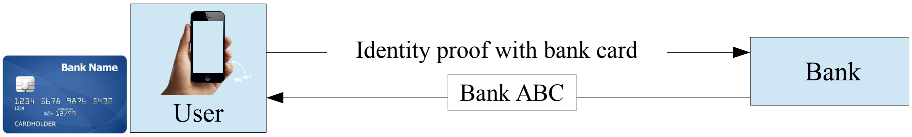


Figure 6.4: High level picture of bank-based self-enrolment where a bank remotely identity-proofs a user based on her existing bank account credential (e.g. via a bank card) and issues an ABC consisting of user attributes stored at the bank to the user's IRMA app.

results in retrieving the User's data from the Bank's records. If the authentication fails, then the Bank aborts the session.

3. As the Bank is both the Enroller and the Issuer in this SSE, it issues an attribute-based credential to the User with identity and the bank-account related attributes. We call this credential *Bank ABC*.

Alternatively, the Bank could act only as the Enroller by identity-proofing users and sending over the user attributes from its records to a proxy Issuer. The Issuer is then responsible for creating an ABC with the attributes received from the Bank and issuing it to the user. With this model, the Bank has to just authenticate its customers during SSE, similar to what it does at the beginning of internet banking transactions and delegate the issuance of ABCs to an ABC-issuer. This approach minimizes the setup required at the Bank to support SSE. Due to its practicality and ease of setting up SSE, IRMA follows the latter approach with separate identity-proofing and issuing steps in its implementation of bank-based SSE protocol. The details are given in Section 6.6.2.

Depending on the authentication mechanism of a particular bank used for identity proofing a user, this SSE method can provide different levels of binding between the user's real and the claimed identity. SSE by banks can potentially offer strong binding if an out-of-band authentication using a bank card, a secure card reader and a PIN is used to authenticate users during enrolment. The binding is not very strong if the bank authenticates its users based on username-password. This variety in security level of the authentication mechanisms makes evaluation of the assurance provided by this form of SSE hard. Thus, the downside of this SSE is the plurality of authentication mechanisms offered by banks. This means that a system that implements SSE such as IRMA should support the mechanisms of all participating banks. Additionally, this means that all these mechanisms will need to be evaluated separately and the resulting process may not provide a uniform user experience.

For SSE to be viable, a uniform system of an authenticating service combining several banks is necessary. Within the Netherlands, the main banks have started a joint authentication service called iDIN[9], where the different banks authenticate their customers with their existing e-banking tokens. The result is a uniform identity veri-

[9] http://www.connective.eu/financial/idin/

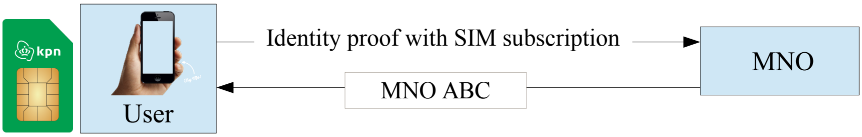


Figure 6.5: High level picture of MNO-based self-enrolment where an MNO remotely identity-proofs a user based on her SIM subscription and issues an ABC consisting of the user's identity attributes stored at the MNO to her IRMA app.

fication message. A system like iDIN makes the deployment of secure self-enrolment by banks much simpler. Section 6.6.2 describes the existing SSE implementation with iDIN in IRMA.

6.4 MNO-based SSE

Mobile providers, also called Mobile Network Operators, could function as Enroller and Issuer in SSE. The trust anchor in this case are the Subscriber Identity Module (SIM) cards with mobile subscriptions that individuals get from these MNOs. Major MNOs in countries like the Netherlands, carry out face-to-face identity proofing for personal mobile subscriptions. This is done at an MNO office or at the user's home, when the SIM card is delivered. Thus we assume that MNOs have authentic identity data and authentic SIM identities of the subscribers in their databases, obtained via a separate channel involving face-to-face authentication.

We describe the MNO-based SSE protocol as follows, see the high level description of this SSE in Figure 6.5.

1. The User connects to an MNO server through her phone and requests SSE.
2. The MNO server recognises[10] the SIM identity of the connecting user's phone and retrieves the subscriber's data corresponding to the SIM.
3. Then it issues an *MNO ABC* to the User that consists of the subscriber attributes such as, name, date of birth, phone number from its subscriber database. Alternatively, as described in the bank-based SSE, the MNO may decide to just act as an Enroller and delegate the issuance of the ABC to a proxy issuer.

The ABC is stored on the phone that contains the SIM card of the User. In the MNO-based enrolment, the MNO directly connects to the SIM card and checks its authenticity before issuing the ABC. In some sense, it provides binding between the device and the credential. Furthermore, as the User has undergone a prior face-to-face enrolment at the MNO when she bought the SIM card and signed the subscription contract, the SSE provides additional assurance about the identity of

[10] SIM identification by an MNO is a topic in itself, which is out of scope here.

the SIM card owner.

Suppose the phone that contains the SIM gets in the hands of an adversary before an enrolment. Then the adversary can attempt to enrol at the MNO as the original SIM card owner on the same phone. The adversary can only manage to get the owner's personal attributes on the phone not any other attributes. Let us consider the case when the adversary removes the SIM from the owner's phone, puts it in her own phone and requests the enrolment as the legitimate SIM owner. This instance is not possible if the SIM was protected by a PIN at booting time. In principle, an MNO has the capability to enable and preset PINs for all the SIMs it provides and inform all the subscribers to change the default SIM PIN at the initiation of their subscriptions. Basically, in an MNO-based SSE, it is impossible for an adversary to get her identity credential from the MNO on somebody's phone or to get somebody's credential on her phone without the knowledge of the SIM PIN. Even if the PIN was known to the adversary, we can prevent the above adversarial actions in the MNO-based SSE by including an additional verification factor, for example, an eID, along with the SIM identity. We will see how such an additional factor strengthens the security of this SSE method in Section 6.5.

The downsides of SSE by MNOs are: (1) it is limited only to the MNOs which carry out face-to-face enrolments of their subscribers for obtaining SIM subscriptions; (2) it is not applicable to users who own prepaid/anonymous SIM cards: they can often be obtained without the user having to go through a face-to-face enrolment at the MNO.

6.5 Combining Self-Enrolment Approaches

Since none of the above presented self-enrolment methods is clearly "the best", it is worthwhile to look into ways to combine these approaches. Trusted self-enrolment requires more than a single protocol, and can be realized by combining several self-enrolment protocols. If they yield consistent outcomes, then the credentials resulting from a combination of enrolments yield higher levels of trust than the ones from a single enrolment of a lower trust level. In the following sections, we describe how two enrolments can be merged into one to create ABCs of high level of trust. However, we note that the enrolments can be done separately, after a delay, and then stacked one over the other to create credentials with higher level of trust.

For instance, a user can get a Bank ABC with elevated trust, called *Bank*$^+$ *ABC* (see Figure 6.6). This is done by identity proofing the user based on two trust anchors: bank account and eID. The bank identity-proofs the user by verifying the user's eID document or an eID ABC in addition to her bank account credentials during enrolment. If the user's attributes in its customer database and the eID are the same, Bank (or a proxy issuer who receives user attributes from Bank) issues *Bank*$^+$ *ABC* to the user. Depending on the issuance policy, the bank may issue only the user attributes existing in the bank records or a union of attributes present in the bank records and the ones in the eID document. The user's identity information in both eID and the bank's database are matched and verified to be consistent and

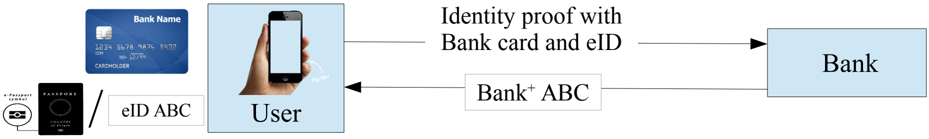


Figure 6.6: High level picture of combined SSE where a bank remotely identity-proofs a user based on her eID (physical document or ABC) and issues a Bank$^+$ ABC to her IRMA app.

unchanged since the eID enrolment before the issuance of Bank$^+$ ABC. Due to this, the attributes disclosed from it provides a higher level of trust to the service providers than the Bank ABC. The trust in the binding between the real identity of the user and the attributes in the ABC is even higher if a biometric of the user is verified during the self-enrolment. That is, if Bank conducts the biometric face verification as a part of the eID verification or if the eID ABC was issued as a result of enhanced eID-based self-enrolment.

The + superscript on the Bank ABC indicates that it has been issued after a strict identity-proofing based on two trust anchors and as a result, it relies on two previous face-to-face enrolments (one for the eID document and one for the bank account). In this way, the separate enrolment protocols reinforce each other. Moreover, now the user identity verification by the Bank during this combined SSE involves three factors: "something you know" – bank authentication password or card PIN, "something you have" – a bank account and an eID document and "something you are" – biometric check during eID verification. Thus, the +-superscripted ABC is more tightly bound to the real identity of the user than a regular bank ABC and thus, provides more assurance to the credential verifiers.

Similar to the Bank-eID SSE, eID-based and MNO-based SSE protocols can be combined to create a stronger SSE. The *MNO$^+$ ABC* is a result of an enrolment that involves checking of two trust anchors by the MNO: the personal mobile subscription and the eID of the User. Moreover, the identity verification by the MNO includes two "something you have" factors – valid SIM card and eID and a "something you are" factor if the biometric face verification was performed as a part of eID verification. By including the eID (document or ABC) verification step in the regular MNO-based SSE, we prevent an adversary with a lost or a stolen phone to be able to complete a self-enrolment and get an MNO ABC. Now the adversary cannot impersonate a User during self-enrolment unless she possesses both the phone and the eID of the User. Thus, the MNO$^+$ ABC has higher level of trust than the regular MNO ABC.

Benefits of combining different SSE methods. Combining self-enrolment methods with different trust anchors strengthens the security provided by the enrolment methods just as multi-factor authentication does. Within such a multi-step enrolment system one can support credentials of varied trust assurance levels, corresponding to the number (and nature) of the self-enrolment steps that the user performed.

So piling several SSE methods on top of one another can result in users obtaining attribute-based credentials with higher assurances (e.g. $^+$ or even $^{++}$). It would then be up to service providers to decide which assurance level they accept for their services. For instance, an online ticket service would likely accept lower assurance level credentials whereas a higher assurance level credential would be required to review your own medical data at a hospital portal.

6.6 SSE Implementation in IRMA

In this section, we describe the secure self-enrolment methods that are implemented for the issuance of trustworthy ABCs within the IRMA system. We also describe other self-enrolments that are supported by IRMA for issuing some user attributes (e.g. name attribute stored by social networks, email address attribute). These credentials need not provide a high level of trust as the ones resulting from SSE, but they can be useful in some circumstances. For instance, (i) the user can log into Facebook using her name attribute from the Facebook ABC without having to remember and enter the username-password at every login; (ii) the user can get an email address attribute that can be used as an additional check in an SSE.

6.6.1 Basic eID-based SSE

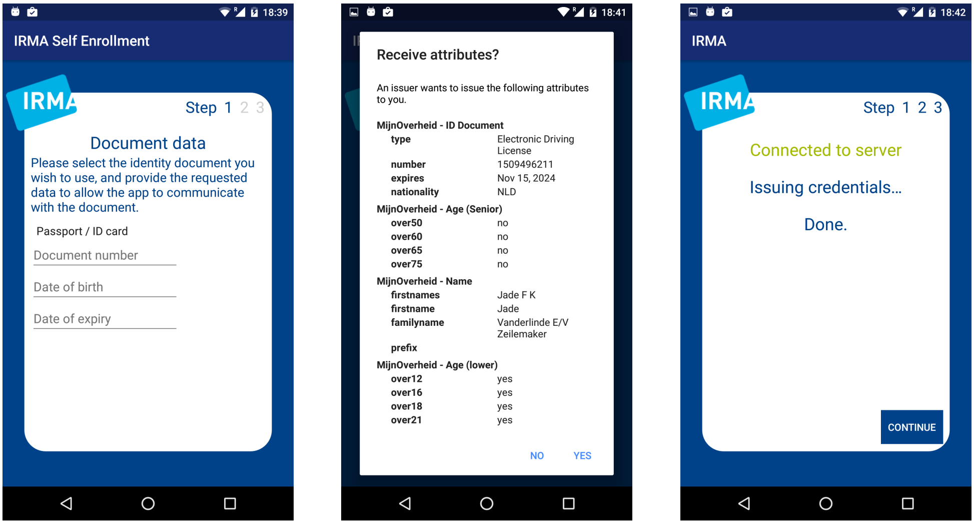
The IRMA team implemented the basic form of eID-based self-enrolment protocol[11]. This prototype implementation uses the following components.

- ICAO-standard eID documents (e.g. e-passports, identity cards, driver's licenses);
- Android smart phones that are enabled with NFC and have IRMA app installed on them;
- Enroller server;
- Issuer server.

Under this implementation, the user chooses to enrol with her eID document via her smartphone's IRMA app and enters the BAC data manually or by scanning a QR code that is printed on the latest version of Dutch driver's licence. The IRMA app essentially functions as a remote card reader. It reads some data from the eID document via the phone's NFC interface and sends them to the Enroller server. This server then verifies the validity of the document, extracts some user's personal data from it, and requests the Issuer server to issue some credentials containing the extracted data to the smart phone. Some of the screenshots of the IRMA app handling self-enrolment are provided in Figure 6.7 and the working of IRMA enrolment can be seen in action in the YouTube video[12].

[11] IRMA self-enrolment implementation details can be found at `https://github.com/credentials/irma_mno_server/blob/master/README.md`.

[12] `https://www.youtube.com/watch?v=q6IihEQFPys` (see especially from 1:24 to 1:52)



Document data

Please select the identity document you wish to use, and provide the requested data to allow the app to communicate with the document.

Passport / ID card

Document number

Date of birth

Date of expiry

Receive attributes?

An issuer wants to issue the following attributes to you.

MijnOverheid - ID Document	
type	Electronic Driving License
number	1509496211
expires	Nov 15, 2024
nationality	NLD
MijnOverheid - Age (Senior)	
over50	no
over60	no
over65	no
over75	no
MijnOverheid - Name	
firstnames	Jade F K
firstname	Jade
familyname	Vanderlinde E/V Zeilemaker
prefix	
MijnOverheid - Age (lower)	
over12	yes
over16	yes
over18	yes
over21	yes

Connected to server

Issuing credentials...

Done.

Figure 6.7: Screenshots from an IRMA eID-based self-enrolment session

Although this eID-based SSE implementation is a neat solution to deal with self-enrolment in the IRMA system, it has the following drawbacks.

- it is restricted to smart phones with NFC interface and to users having eID documents with an NFC chip. Thus, this SSE method limits the number of users who can enrol and get trusted ABCs on their IRMA apps, and
- it does not include the biometric face verification at the Enroller server as this check requires an expensive setup. The SSE implementation cannot rely on the PIN check as not all standard eID documents support PIN.
- The eID-MNO SSE approach was considered to achieve better user-eID binding. This combined SSE approach was discussed and evaluated in detail with a major MNO in the Netherlands. In the end, the MNO decided not to implement this SSE protocol because its database of subscriber data is highly protected and could not be used for research.

Due to the above reasons, this implementation has been made obsolete and the current version of the IRMA app no more supports self-enrolment involving eID documents. However, PbD foundation has implemented a separate tablet app (not a part of the IRMA app) that verifies a passport, extracts attributes from it, and issues them to a user. This is an instance of a face-to-face enrolment where an employee at a counter could also check a user's face against the picture in the passport, thus performing a biometrics check, and then start the issuance of ABCs. This enhanced eID-based enrolment can easily be used to issue trusted attributes in the user's passport to her IRMA app but the user will have to be physically present at the enroller's office during this enrolment. Currently, IRMA app actively supports only self-enrolments such as, bank-based SSE using the Dutch iDIN system and few others. We describe these implementations in the following subsections.

6.6.2 Bank-based SSE via iDIN

In this section, we discuss how IRMA implements the bank-based SSE using iDIN[13]. As mentioned in Section 6.3, many banks in the Netherlands support iDIN, an online identification service that allows bank customers to trustworthily identify themselves to other organisations or online services through their banks. When a website asks a user to provide her identity data such as, name and birth date, iDIN allows the user to choose her bank from a set of banks and authenticate to it using any means that the bank offers. Following a successful authentication, the bank provides the user's identity information to the website. So, iDIN is basically a system for users to retrieve and use their identity information stored at their banks for online identification purpose. Such a system makes the deployment of secure self-enrolment simple.

SSE with iDIN is supported by the IRMA platform[14]. In IRMA SSE with iDIN, the bank acts as the Enroller and the Privacy by Design (PbD) foundation [37] acts as the proxy Issuer. The PbD foundation has signed a contract with some of the banks in the Netherlands that support iDIN, to be the iDIN attributes receiver as service provider. A user, who requests self-enrolment at the PbD foundation, authenticates to her bank via iDIN. Upon successful authentication, the bank sends an enrolment confirmation and the validated attributes of the user to the PbD foundation. Subsequently, the PbD foundation issues the *Bank ABC (or iDIN ABC)* to the user[15]. This credential consists of the user's personal attributes stored at her bank: initials, family name, date of birth, gender, address, postal code, city. The iDIN ABC assures other credential issuers and service providers of the authenticity of the link between the user's real identity and the attributes within this ABC. Screenshots of an iDIN-SSE in IRMA where a user gets an iDIN ABC are shown in Figure 6.8.

6.6.3 University SSE

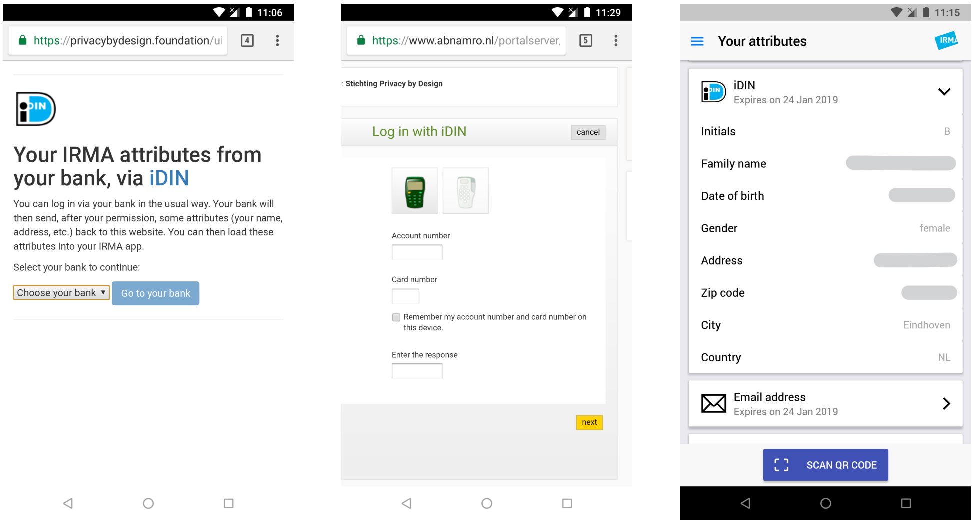
Ideally in attribute-based identity management, there are trusted parties other than governments, banks and mobile network operators that possess users' identity information. The IRMA platform supports another SSE method via SURFconext[16]. SURFconext provides federated identity management for the (higher) education sector in the Netherlands. Similar to the iDIN SSE, IRMA users can login to their educational institutes via SURFconext and get their *SURF ABC* from PbD foundation. This ABC consists of the users' identity attributes: given name, family name, email address, institution, staff/student, local registration number stored by their Dutch educational institutes (see Figure 6.9). Now, via SURFconext, connections

[13] Details on iDIN can be found at `https://www.idin.nl/`.

[14] See `https://privacybydesign.foundation/issuance-idin/` for details on the issuance of iDIN attributes in IRMA.

[15] The PbD foundation does not keep a log of such issuances since it only passes on attributes from third parties and does not wish to keep unnecessary information and run into unnecessary privacy risks.

[16] `https://www.surf.nl/en/services-and-products/surfconext/index.html`



11:06

https://privacybydesign.foundation/ui

Your IRMA attributes from your bank, via iDIN

You can log in via your bank in the usual way. Your bank will then send, after your permission, some attributes (your name, address, etc.) back to this website. You can then load these attributes into your IRMA app.

Select your bank to continue:

Choose your bank ▾ | Go to your bank

11:29

https://www.abnamro.nl/portalserver,

Stichting Privacy by Design

Log in with iDIN | cancel

Account number

Card number

□ Remember my account number and card number on this device.

Enter the response

next

11:15

Your attributes

iDIN
Expires on 24 Jan 2019

Initials	B
Family name	[redacted]
Date of birth	[redacted]
Gender	female
Address	[redacted]
Zip code	[redacted]
City	Eindhoven
Country	NL

Email address
Expires on 24 Jan 2019

SCAN QR CODE

Figure 6.8: Screenshots from an IRMA SSE session with iDIN

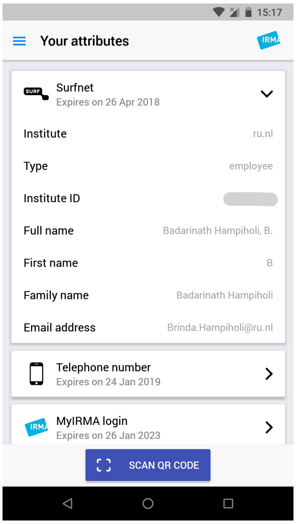
are also being established with eduGAIN[17], for issuing attributes from international educational institutions.

6.6.4 Other self-enrolments in IRMA

Getting social media ABCs. Social networks, including Google and Facebook, offer federated identification for its users. Effectively, people can introduce themselves with their Google or Facebook profile at other websites. IRMA allows its users to get their attributes stored at these social networks. The users can log into social media networks such as Facebook, LinkedIn and Twitter, via the PbD foundation and obtain respective ABCs. An example Facebook ABC is shown in Figure 6.10. Of course, the level of trust in a Facebook or a Twitter ABC is limited, because these social media attributes are provided by users themselves and they are not verified by the networks (i.e. users are not identity-proofed by the networks). Thus, it is up to verifiers whether or not to rely on attributes contained in social-media ABCs.

Getting email and mobile number ABCs. The PbD foundation carries out other credential issuances where it validates itself certain attributes, such as mobile phone numbers and email addresses, via one-time codes, and then issues them as ABCs to users. A user can get one or more email address ABCs in the IRMA app as shown in Figure 6.10. Here, the issuer can only verify if the user currently has access to this email address or phone number but not if these attributes reflect the real identity of the user. So, these ABCs, by themselves, may provide low or no trust assurance about the user's identity. However, they maybe used to prove a user's possession of a valid electronic address or a phone number in the issuance of other

[17] Details about eduGAIN can be found on `https://edugain.org/`.


15:17

Your attributes

Surfnet
Expires on 26 Apr 2018

Institute	ru.nl
Type	employee
Institute ID	[redacted]
Full name	Badarinath Hampiholi, B.
First name	B
Family name	Badarinath Hampiholi
Email address	Brinda.Hampiholi@ru.nl

Telephone number
Expires on 24 Jan 2019

MyIRMA login
Expires on 26 Jan 2023

SCAN QR CODE

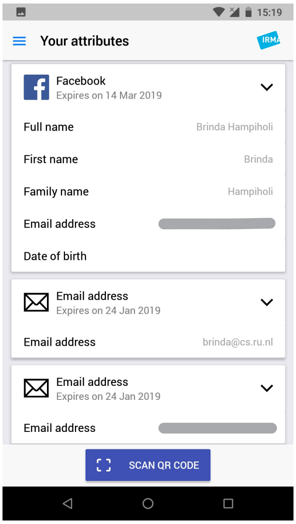
Figure 6.9: Surf ABC issued by PbD foundation as a result of University SSE

credentials to the user.

6.7 Assurance Levels of IRMA ABCs

Assurance levels characterise the degree of confidence in electronic identification means in establishing the identity of a person. They provide assurance that the person claiming a particular identity is in fact the person to whom that identity was assigned [127]. The eIDAS regulation 2015/1502 sets out minimum technical requirements for achieving low, substantial and high assurance levels for electronic identification means [128]. However, the regulation does not explicitly define what are the acceptable forms of electronic identification means (eIDM). The conventional forms of eIDM are physical documents such as identity cards. Although eIDAS attempts to abstract away from the chip-based cards in the levels of assurance implementing act [128], in some processes, the requirements for assurance levels are drafted by considering physical identity cards as the eID means. However, in this chapter, we consider non-physical electronic identification means: attribute-based credentials that are stored inside a smartphone application.

A user's identity is made up of different kinds of attributes based on the context. For example, the user holds a set of citizen attributes (from an eID document), bank-account holder attributes (from her bank), student attributes (from her university) etc. Each set of attributes are contained in an ABC and is used to authenticate the user in different contexts. Consider an example of an ABC with name, address and social security number attributes. This ABC can serve as electronic identification means for the ABC owner in various online scenarios, for instance, to declare taxes, to apply for social benefits or to buy things from webshops. IRMA provides a smartphone application called the *IRMA app* and the support infrastructure that realises



15:19

Your attributes

Facebook
Expires on 14 Mar 2019

Full name Brinda Hampiholi

First name Brinda

Family name Hampiholi

Email address [redacted]

Date of birth

Email address
Expires on 24 Jan 2019

Email address brinda@cs.ru.nl

Email address
Expires on 24 Jan 2019

Email address [redacted]

Figure 6.10: Facebook and Email ABCs issued by PbD foundation as a result of self-enrolments

attribute-based identification, authentication and digital signing in practice [37]. It is crucial that the ABCs used in identity platforms such as IRMA have reasonable assurance levels in order to establish confidence in the claimed identity of users. Many organisations and service providers who verify the identity attributes of users require certain assurance levels from the electronic identification means (ABCs in our case). Therefore, we perform a preliminary assessment on the ABCs within the IRMA system, based on the requirements per assurance level set by the European regulation eIDAS [127] to find out which assurance levels are provided by the ABCs. Although it is still early to assess the IRMA ABCs and the infrastructure from a management and organisational perspective, we can already look at how the IRMA ABCs meet the conceptual requirements set by eIDAS.

For our assessment, we consider the requirements of the eIDAS Levels of Assurance defined in Commission Implementing Regulation (EU) 2015/1502 pursuant to Article 8(3) of the eIDAS Regulation [(EU) 910/2014] [128]. We present our interpretation of each requirement in the context of IRMA ABCs and analyse if the ABCs can meet the requirements for the assurance level 'substantial ' or 'high'. The following processes are taken into account to determine the overall assurance level provided by the IRMA ABCs. The overall authentication assurance level provided by the ABCs is determined by the lowest assurance level achieved in any of the areas listed below.

1. Enrolment: This process involves subprocesses such as, application and registration, identity proofing and verification.
2. Electronic identification means management: This includes the characteristics and design of ABCs, delivery/activation, revocation, renewal and replacement.
3. Authentication: This includes the measures to be taken to mitigate the threats

associated with the use of the authentication mechanism that uses ABCs.

4. Management and organisation: This includes general provisions, published notices and user information, record keeping, facilities and staff, technical controls implemented to manage the risks posed to the security of the services, compliance and audit.

Our analysis will focuss only on the conceptual requirements that are stated for the functional processes 1-3 and leave out the operational requirements corresponding to the management and organisation. This is because, IRMA ABCs are technically ready to be put into practical use but the IRMA platform itself in a state of flux with respect to management and organisational aspects. The analysis made in this section is not final or complete but it is meant to provide a preview of IRMA ABCs' potential to become an electronic identification means with a high assurance level. In future, the IRMA ABCs need to be assessed by a standard conformity assessment body[18] referred to in Article 2(13) of Regulation (EC) No 765/2008 or by an equivalent body. Such an assessment will confirm the overall authentication assurance level provided by the IRMA ABCs.

6.7.1 Enrolment

The eIDAS document 1502 [128] uses the term 'Enrolment' to denote the complete process, consisting of the following steps:

- Application and Registration;
- Identity proofing and verification – in the current analysis, we restrict ourselves to the eIDAS requirements for identity-proofing natural persons.

Application and registration. The eIDAS requirements with respect to application and registration steps and our interpretation for each of these requirements in the IRMA system are given below. The electronic identification means needs to meet these requirements to attain any of the three assurance levels.

1. *Ensure the applicant is aware of the terms and conditions related to the use of the electronic identification means.*

   The applicant in this case is the user who is applying for the electronic identification means. We refer to the eIDAS guidance document [137] for analysing the requirements to achieve the three levels of assurance. This document provides some examples for how the above requirement can be interpreted: the information, terms and conditions for the functioning and security of the eID scheme could be a part of national legislation and therefore presumed to be known to the applicant, or the applicant gets the information in a written form, or the applicant is made to explicitly accept the terms and conditions.

   In the IRMA system, the IRMA app does not present users with any terms and conditions for using the ABCs. Before installing and using the IRMA app,

[18] https://ec.europa.eu/futurium/en/system/files/ged/list_of_eidas_accredited_cabs-2018-07-27.pdf

users can read the key concepts of ABCs, their security and privacy properties, the main functionality of the IRMA app, setup instructions etc., on the Privacy by Design foundation website [37]. The IRMA app also contains some rudimentary information and pointers to the PbD website that are accessible to users before registration. Based on this, users can make a conscious choice whether to install the IRMA app and complete the registration or not.

2. *Ensure the applicant is aware of recommended security precautions related to the electronic identification means.*

   The guidance document [137] provides some examples for security precautions such as, securely choosing the password/PIN, not handing over the eID means to another person and keeping it safe.

   In the IRMA context, the PbD foundation website provides instructions to users regarding how to securely use the IRMA app such as, the user must not share the PIN[19] with any other person etc. Furthermore, during the registration process, the IRMA app offers brief explanation about some security aspects that users should be aware of, for example that one needs to enter their PIN during each IRMA session, and that one can remotely block their IRMA account if they provide their email address to the IRMA app that the app can verify (by sending a clickable link to that email).

3. *Collect the relevant identity data required for identity proofing and verification.*

   In the case of conventional identification documents such as national identity cards, the issuing authority collects the relevant identity data required to verify the identity of the person beyond doubt at the time of application. In particular, this includes the data that gets stored on the identity cards. The guidance document interprets this requirement as follows: the least information for the minimum data set that is not known to or generated by the eID scheme needs to be collected by the issuing authority from the applicant or other sources.

   In the current context, the eID scheme is IRMA and the electronic identification means are ABCs which can be obtained by a user on her IRMA app after undergoing online self-enrolments described in this chapter. So the enrolment of users for an eIDM means enrolment for an ABC. At the time of registration, a user creates an IRMA account. The IRMA app itself does not collect any identity information from the user as identification of users at this stage is not necessary. The IRMA user only receives a random username (pseudonym) and she can choose to link this pseudonym to her (existing or newly created) email address. Linking the IRMA account with an email address allows the user to later block the account if the phone is lost or stolen.

   Now let us consider the example of an ABC with name and address attributes of the user. Several trusted parties such as bank, university, municipality already possess such attributes of a user and can issue these attributes in the form of ABCs to the user. So the user has to authenticate to any such potential attribute issuer during the identity-proofing stage of the enrolment for ABCs so

[19] The users of the IRMA app have to choose a PIN containing at least 5 digits and remember it to authorise any action (e.g. credential issuance or disclosure) on the app.

that the issuer can reliably recognise the user, retrieve her identity attributes: name and address from its records and issue the attributes to the user. So in the IRMA context, the requirement by eIDAS that an issuing party needs to collect relevant identity data from the user for identity-proofing is equivalent to saying that the issuing party needs to authenticate the user so that it can reliably identify the user, retrieve her attributes from its records at the time identity-proofing.

Identity proofing and verification. The eIDAS regulation states the requirements with respect to the identity proofing and verification of natural persons for the electronic identification means to reach the assurance level 'high' (Refer Section 2.1.2, [128]). User authentication based on ABCs is strong only when it is based on highly identity-proofed credentials. In the context of ABCs in the IRMA system, we consider two sets of eIDAS requirements that are stated below. Upon meeting either set of requirements, the ABCs resulting from the self-enrolments can achieve the assurance level 'high'.

eIDAS requirements set 1.

(1a) *The person has been verified to be in possession of evidence recognised by the Member State in which the application for the electronic identity means is being made and representing the claimed identity*
and

(1b) *the evidence is checked to determine that it is genuine; or, according to an authoritative source, it is known to exist and relates to a real person*
and

(1c) *steps have been taken to minimise the risk that the person's identity is not the claimed identity, taking into account for instance the risk of lost, stolen, suspended, revoked or expired evidence*
and

(1d) *the applicant is identified as the claimed identity through comparison of one or more physical characteristics of the person with an authoritative source.*

The ABCs can be assigned the assurance level 'substantial' if the identity-proofing done during a self-enrolment satisfies the first three requirements in the above list. By satisfying all four requirements, the ABCs will reach the assurance level 'high'.

eIDAS requirements set 2.

(2a) *Where procedures used previously by a public or private entity in the same Member State for a purpose other than the issuance of electronic identification means provide for an equivalent assurance to those set out in the above requirement set for the assurance level high, then the entity responsible for registration need not repeat those earlier procedures, provided that such equivalent assurance is confirmed by a conformity assessment body referred to in Article 2(13) of Regulation (EC) No 765/2008 or by an equivalent body*
and

(2b) *steps are taken to demonstrate that the results of the earlier procedures remain valid.*

If the identity-proofing done during a self-enrolment meets the first requirement in the above set, the resulting ABCs gain the assurance level 'substantial' and if both the requirements are satisfied, the ABCs gain the assurance level 'high'.

Now, we map the identity proofing steps carried out for a user to obtain specific IRMA ABCs to the above eIDAS requirements, and assign the corresponding assurance levels to the issued ABCs in Table 6.1. In the IRMA system, the above requirements (1a-1d, 2a-2b) are interpreted as follows:

(1a,2a) We consider the evidence presented by the user during enrolment to be the trust anchor (aka. authoritative source) that has been previously issued to the user by some well-known authority; for example, a government-issued identity card, bank-issued credentials. Such trust anchors have an assurance level 'high' as the face-to-face identity proofing of applicants performed by the issuing authorities (e.g. government, bank) before their issuance satisfies all the requirements in the first set of requirements (1a-1d). In the case of self-enrolments for ABCs, the IRMA enroller verifies the user's possession of such a trust anchor during identity-proofing. Here, the main consideration is if the enroller can rely on the enrolment that was done for the issuance of the trust anchor for authenticity and strong binding of the identity data in the trust anchor to the identity of the enrolling user;

(1b) The IRMA enroller validates the trust anchor (i.e. determines if it is genuine) as a part of the self-enrolment;

(1c) The IRMA enroller checks the revocation status of the presented trust anchor;

(1d) The IRMA enroller compares one or more physical characteristics of the user with the trust anchor (e.g. biometric check with an eID);

(2b) The IRMA enroller checks validity and freshness of certain identity attributes of the user that might have changed over time (e.g. name, address). This can be done by matching such attributes of the user in its own registers with the attributes present in other reliable sources (e.g. national population registers, national identity cards).

Now we analyse some of the self-enrolment methods described in this chapter in Table 6.1. Based on the interpretation above, we describe how each self-enrolment method handles the eIDAS requirements (1a-1d or 2a-2b) in the identity-proofing procedure column of the table. We note that in the case of enrolments for obtaining ABCs, it is possible to satisfy all the requirements (either in set 1 or set 2) by combining two or more self-enrolments and attain assurance level 'high' for the resulting ABC. This possibility is described in Section 6.5 and we also illustrate it in the last two rows of Table 6.1.

Self-enrolment Type	**Identity-proofing Procedure**	**Resulting ABC & its attributes**	**eIDAS Assurance Level**
Basic eID-based enrolment	(1a) The possession of a government-issued eID is verified remotely; furthermore, this self-enrolment relies on the face-to-face enrolment done for the eID issuance (1b) The eID is electronically validated to determine if its genuine; (1c) Revocation status is verified; (1d) no biometric check	*eID ABC*: name, date of birth, nationality, eID type, number, expiry date	Substantial
Bank-based enrolment via iDIN	(1a) The possession of a valid bank account is verified. This remote identity-proofing relies on the face-to-face enrolment done for opening the bank account (bound by regulation such as KYC[20]) (1b) User is authenticated via bank credentials (mostly, two-factor authentication), and identity and account-related data is validated; (1c) Verification of the transaction history and bank account revocation status is possible (1d) no biometric check	*Bank ABC*: name, date of birth, gender, address	Substantial
University-based enrolment via Surfconext	(1a) User's university ID is verified; this remote identity-proofing relies on the previous university enrolment (done face-to-face when employment contract is signed); (1b) User is authenticated based on university credentials (mostly involves one factor: username-password); Identity and affiliation information (e.g. student, employee) checked; (1c) User's ID is checked against expired/revoked university IDs (1d) no biometric check	*Surf ABC*: name, university name, employee or student ID, email address	Substantial
Facebook enrolment	Facebook is not an authoritative source for a person's identity credentials as it has only the self-asserted identity data of users	*Facebook ABC*: name, email address, date of birth	None
Email/Phone number credential issuance	Although email and phone number attributes are verified via one-time codes, they do not originate from an authoritative source	*Email/Phone ABC*: email address/phone number	None
Enhanced eID enrolment	Basic eID enrolment plus biometric check	*eID*$^{+}$ *ABC*: same as *eID ABC* attributes	High

Bank + Basic eID enrolment	(2a) If the enrolling user has bank and eID ABCs, then the bank can verify both ABCs and rely on the enrolments done for both of them. Alternatively, bank could also perform all the identity-proofing steps of bank-based SSE and basic eID SSE; (2b) the user's identity information such as, name mentioned in the eID is matched with the user's name in the bank's records (and also the freshness of this data is checked) – this is to ensure that only up-to-date data of the user is issued as an attribute in the ABC	$Bank^{+}\ ABC$: $Bank\ ABC \cup eID\ ABC$ attributes	High
Bank + Enhanced eID enrolment	(1a-1d) This combined enrolment involves the verification of the user's identity information existing at the bank, basic eID checks and the biometric check	$Bank^{++}\ ABC$: $Bank\ ABC \cup eID\ ABC$ attributes	High

Table 6.1: Summary of self-enrolment methods with specific identity proofing steps carried out in the methods and their approximate mapping to eIDAS assurance levels. The first five methods are implemented in IRMA; last three methods are mentioned in the table to show the assurance levels for the resulting ABCs elevates by including a biometric check and/or by combining two SSE protocols.

The main observations from Table 6.1 are as follows.

- Even for a low assurance level, eIDAS requires that an authoritative source knows that the claimed identity of the user exists and it is valid. Facebook, email and mobile number credentials cannot be assigned any assurance level as the attributes within those credentials do not come from an authoritative source. Users are not identity proofed by enrollers in these cases and the final ABCs are based on some data provided by the users themselves at the time of Facebook or email account creation. In the context of this chapter, we do not consider these self-enrolments as secure self-enrolments and thus, the resulting ABCs cannot be assigned with any assurance level.
- Addition of a biometric factor verification step during identity proofing increases the assurance level of resulting ABC from 'substantial' to 'high' (e.g. Enhanced eID SSE).
- From the last two entries in the table, we see that we can achieve higher levels of assurance by combining two or more self-enrolment protocols than we could have got from an individual self-enrolment protocol.
- We see that some self-enrolments can result in ABCs with the same attributes but different assurance levels; for example, basic and enhanced eID self-enrolments. There needs to be a way for verifiers (service providers) to distinguish between these ABCs and know their exact assurance level. For this purpose, as shown in the table, the ABCs with higher assurance levels are named differently (e.g. eID^{+} ABC) from the ABCs with same attributes but slightly lower assurance level (e.g. eID ABC). We envision a public policy document associated with an ABC that contains the description of the attributes

contained in the ABC and its naming convention. This document is signed by the issuer to ensure authenticity and it enables the service providers to know the assurance level of the ABC from which the attributes are disclosed during an authentication just by looking at the name of the ABC. Alternatively, the assurance level could be issued as one of the attributes that is disclosed by default to all the service providers when users authenticate with ABCs.

6.7.2 Electronic identification means management

In this subsection, we state the eIDAS requirements for the assurance level 'high' with respect to the management of the electronic identification means, interpret them in the IRMA context and describe how ABCs in the IRMA system can satisfy them.

Electronic identification means characteristics and design. This category consists of requirements with regard to (i) the number of authentication factors supported by the eID means and, (ii) measures taken by the eID scheme during the design of the eIDM to protect it against duplication, guessing, replay and manipulation of communication threats.

1. *The electronic identification means utilises at least two authentication factors from different categories.*

   The eIDAS document 1502 [128] defines 'authentication factor' as a factor confirmed as being bound to a person, which falls in any of the three categories: knowledge-based factors (e.g. PIN, password); possession-based factors (e.g. a cryptographic secret key stored on a smartcard or smartphone); inherent factors (e.g. biometrics). An identification means that utilises more than one factor from different categories is called multi-factor, for example: a smartcard (possession) that is activated via a PIN (knowledge) is a multi-factor identification means.

   The IRMA ABCs utilise two authentication factors: possession-based authentication factor (secret key) and knowledge-based authentication factor (IRMA PIN). The secret key is bound to all the ABCs and it is required to successfully authenticate a user to a service provider. The IRMA PIN is required to perform any action on the IRMA app such as acquire or use ABCs. This PIN contains at least 5 digits and is chosen by the user at the time of registration. Even if the smart phone is stolen, the ABCs stored in the IRMA app cannot be used by anyone who does not know the PIN.

2. *The electronic identification means protects against duplication and tampering as well as against attackers with high attack potential.*

   In the current analysis, we interpret this requirement as follows. It must not be possible for an adversary to copy or modify the ABCs stored in the IRMA app. The IRMA ABCs are protected against unauthorised access and tampering by the security enforcements of the smartphone's operating system. Nobody can access or modify the contents of the app unless they have root access to the user's smartphone. If the adversary has root access, then she can copy the entire app along with its contents (ABCs, secret key, issuer's signatures) onto another phone. However she cannot use the ABCs without knowing the user's IRMA PIN. The PIN is not stored anywhere in the app. So the adversary will still have to gain the knowledge of the PIN to access the app and use the ABCs. Furthermore, it is not possible to change ABCs and use them at service providers because the issuer's signature on the ABCs will become invalid upon modification. ABCs provide this integrity guarantee by design.

3. *The electronic identification means is designed so that it can be reliably protected by the person to whom it belongs against use by others.*

   The guidance document mentions that 'reliably protected' refers to the efforts undertaken to prevent the electronic identification means from being used without the subject's knowledge and active consent. IRMA ABCs can be protected by the user (i.e. credential owner) by safeguarding her phone and the PIN. The cryptographic key bound to the ABCs. The key is secured by distributing its shares among the user's IRMA app and a central server called the key-share server (See Section 5.2.1 for details). Both the app and server have to cooperate for the user to successfully use an ABC at a verifier. The key cannot be used by any entity without user's active consent which is given by entering the PIN code. In fact, PIN authenticates the user to the key-share server. To counter the PIN guessing threat, the key-share server is configured to temporarily block access to the key-share if an incorrect PIN is entered three consecutive times. Thereafter, the key-share server incrementally slows down the PIN attempts. Furthermore, in the case of a lost or stolen phone, the user can also instantaneously revoke her secret key by blocking the key-share at the key-share server. The exact procedure to accomplish user-initiated revocation is described under the revocation process.

Issuance, delivery and activation. The measures taken to ensure that the electronic identification means is delivered to and can be activated by only the legitimate owner of the identity are considered while deciding the assurance level. The eIDAS regulation requires a secure delivery of electronic identification means and an activation process that involves user intervention to reach an assurance level 'high'. Formally, this requirement is stated in the document 1502 [128] as follows.

1. *The activation process verifies that the electronic identification means was delivered only into the possession of the person to whom it belongs.*

   The eIDAS regulation mostly considers the physical delivery of electronic identification means and activation codes via regular mail to the applicant or that the applicant collects them from the trusted parties after verifying her identity in person [137]. This consideration seems strongly inspired by physical electronic identification means and does not include the possibility of non-physical eIDM such as, ABCs that are issued online by an issuer to the IRMA application on the user's smart phone. In the IRMA context, we interpret the above requirement as follows.

   Identity proofing and issuance of ABCs happen within a single secure session over HTTPS. So, the issuer can verify that the ABC was issued (or delivered) only to the identity-proofed user. IRMA relies on HTTPS for the secure delivery of ABCs to the legitimate owner. While receiving the attributes from an issuer, the user has to enter the correct IRMA PIN. The correct PIN is needed for every action involving an ABC on the IRMA app. Entering of the correct PIN acts as an activation step in IRMA. In sum, the IRMA app not only ensures that the ABCs are delivered to the correct user although done online but also involves an explicit activation step from the user in order to take ownership of the ABCs.

Suspension, revocation and reactivation. According to the guidance document, eIDAS mostly considers the suspension and revocation of users that is initiated by the verifier or the public authorities in general. At present, IRMA supports only the user-driven revocation of the ABCs stored on her smartphone via the key-share server. Schemes for issuer- or verifier-driven revocation of credentials are already present in the literature [58, 138, 57] but implementation of such a scheme within the IRMA system is subject to future work. Below we state the specific requirements for suspending, revoking and reactivating users under eIDAS regulation to reach the assurance level 'high' and describe how these requirements are met under the current implementation of the IRMA ABCs.

1. *It is possible to suspend and/or revoke an electronic identification means in a timely and effective manner.*
   IRMA allows a user to revoke the use of the ABCs stored on her smartphone's IRMA app in a timely and effective manner in case the phone is lost, stolen or corrupted. The revocation in the IRMA system is accomplished through sharing of the secret key between her smart phone and the key-share server. By this, the user can instantly revoke the use of ABCs by blocking the key share on the server. The exact procedure to block the use of IRMA ABCs is as follows. To enable blocking, the user should have provided a valid email address that she controls to the key-share server (via the IRMA app) during or after registration. If the user loses her phone, then first she authenticates to the key-share server by proving the ownership of the email address linked to her IRMA account. The authentication is carried out via the user receiving one-time password (OTP) on her registered email address. When the user enters the correct OTP, the key-share server blocks the user's key share that it has, thereby preventing any further use of ABCs from the user's lost/stolen phone. Currently, the key-share server uses email address to authenticate the user who cannot authenticate with her PIN anymore, for instance, due to loss of her phone. In future, the server can use other authentication mechanisms for this purpose.

2. *The existence of measures taken to prevent unauthorised suspension, revocation and/or reactivation.*
   Currently in the IRMA system, only a user who owns IRMA ABCs and has linked her email address to her IRMA account can initiate revocation. This involves the user to authenticate to the key-share server by proving the ownership of the pre-registered email address (as explained above).

3. *Reactivation shall take place only if the same assurance requirements as established before the suspension or revocation continue to be met.*

   Reactivation involves the reissuance of all the ABCs on the IRMA app. This mandates the user to undergo identity proofing again. Other assurance requirements remain the same as before the revocation.

Renewal and replacement. For an assurance level 'high', eIDAS sets out the following requirements with respect to the renewal and replacement of the eID means if they are expired or lost. We mention how this aspect is dealt with in the IRMA system.

1. *Taking into account the risks of a change in the person identification data, renewal or replacement needs to meet the same assurance requirements as initial identity proofing and verification*

   and

   Where renewal and replacement is based on a valid electronic identification means, the identity data is verified with an authoritative source.

   Every ABC in IRMA has an expiry date. Once it expires, it has to be reissued by repeating the secure self-enrolment. The same requirement applies even when a user loses her phone and needs to replace her old ABCs. So, the ABCs in the IRMA system cannot be renewed or replaced without a new enrolment. This allows the enroller and the issuer to verify the most recent person identification data of the user and issue them as new ABCs.

6.7.3 Authentication

This subsection focuses on the technical requirements stated in the eIDAS document 1502 (Section 2.3 in [128]) that an eID scheme should satisfy to mitigate the threats associated with the use of the authentication mechanism in order to reach a given assurance level.

Authentication mechanism In the IRMA system, the authentication mechanism uses ABCs. It consists of disclosing attributes from the credentials and dynamically proving the validity of the attributes to the verifiers. The eIDAS document 1502 defines 'dynamic authentication' as an electronic process using cryptography or other techniques to provide a means of creating on demand an electronic proof that the subject is in control or in possession of the identification data and which changes with each authentication between the subject and the system verifying the subject's identity. In IRMA, dynamic authentication is accomplished by a fresh electronic authentication proof that is computed over the ABCs, secret key of the user and an unpredicatable random nonce (for freshness) sent by the verifier. The cryptographic details of this mechanism is given in Section 2.4 of the Preliminaries chapter. Now we list the eIDAS requirements for the assurance level 'high' under this category and describe how IRMA ABCs satisfy them.

1. *The release of person identification data is preceded by reliable verification of*

the electronic identification means and its validity.

The guidance document mentions that the release of person identification information is about transmitting the minimum data set to the relying party. In IRMA, the attributes that are disclosed during an authentication constitute the data set requested by the verifier (aka. the relying party). As these attributes are required to verify the authentication proof, the attributes and the proof are sent together to the service providers (relying parties).

2. *Where person identification data is stored as part of the authentication mechanism, that information is secured in order to protect against loss and against compromise, including analysis offline.*

   In IRMA, the person identification data are the personal attributes of a user that are contained in the ABCs. If the user loses her phone or finds out that her phone is compromised, then she can immediately block the use of ABCs by contacting the key-share server. Furthermore, the ABCs are protected against against unauthorised access and use by the IRMA PIN, the smartphone's operating system and the phone's screen unlock mechanism. The IRMA app and the key-share server together ensure that the user has given her consent to the use of the ABCs and to the release of the attributes by entering the PIN. The attributes in the ABCs cannot be modified after issuance even by the user herself – the issuer's signature on the ABCs prevents such modification attacks. This is how IRMA protects the attributes against misuse upon loss or compromise of the phone.

3. *The authentication mechanism implements security controls for the verification of the electronic identification means, so that it is highly unlikely that activities such as guessing, eavesdropping, replay or manipulation of communication by an attacker with high attack potential can subvert the authentication mechanisms.*

   All cryptographic protocols in IRMA (and in Idemix) are resistant against guessing, eavesdropping, replay and manipulation attacks [18, 46]. The IRMA PIN consists of minimum 5 digits and is protected against guessing attacks by attackers with high attack potential. As the communication between the IRMA app and a verifier (service provider) takes place over secure HTTPS sessions, the service provider is also authenticated to the app through the verification of the SP's public key certificate. The attributes and the authentication proof are transmitted in an end-to-end protected secure HTTPS session that is established between the user's IRMA app and the SP which protects them from eavesdropping attacks. The dynamic authentication mechanism of IRMA ABCs prevents replay attacks. The ABCs stored in the IRMA app cannot be modified after issuance; the issuer's signature on the ABCs prevents such modification or manipulation attacks. IRMA uses appropriate key lengths for the user's secret key and the issuer's issuing keys. The key-lengths currently used in IRMA can be found in the source code[21]. IRMA also includes secure management of the user's key via secret sharing.

[21] `https://github.com/mhe/gabi/blob/master/sysparams.go#L33`

6.8 Conclusion and Future Work

In this chapter, we describe secure self-enrolment methods which allow an ABC user with several trust anchors, for example, an eID document, an active bank account, a valid SIM subcription, university ID to enrol herself from any location (e.g. sitting at home) by connecting to the respective enroller. Subsequently, she can obtain attribute-based credentials that are derived from her trust anchor on her smart phone from an issuer. This type of enrolment is convenient, cheap and commercially viable when compared to traditional face-to-face enrolments. Based on the experiences with the IRMA smartphone implementation of some of the described self-enrolment protocols in an ABC framework, we can claim that such approaches are practical and efficient. Furthermore, the approaches are simple to use and, convenient for users as they can do it from anywhere to get trustworthy credentials in a short time.

Combining several self-enrolment methods may help in elevating the levels of trust in the resulting ABCs. For example, by combining the basic eID and bank-based SSE, we add a security layer (eID check) on top of the security provided by the user authentication (identity-proofing) at the bank. Furthermore, the bank can verify if certain identity attributes such as name of the user is consistent in both eID and in its records. This results in the issuance of a highly trusted bank ABC with up-to-date identity attributes. Which solution works best in which situation depends on various factors, such as costs, effort, and willingness of different parties to cooperate. The preliminary assessment on the assurance levels of IRMA ABCs show that the ABCs have the potential to achieve a high level of assurance within the eIDAS framework.

The study performed in this chapter focuses only on self-enrolment methods due to their advantages over face-to-face enrolments. But in situations where face-to-face enrolments can be organised easily, they may be carried out for enrolling users. Some future studies on user experience with the types of remote enrolments as well as face-to-face enrolment may indicate which direction to take. Future work also includes implementation of combined forms of self-enrolments.

Chapter 7

Conclusion and Recommendations

In the current digital world, we are constantly witnessing data breaches, unintended user-data disclosures from company or government databases, privacy scandals involving misuse of personal information of millions of users. Apparently, it is crucial that the technologies that promote data minimisation and user control over personal data are deployed and used in real-world use cases. Data protection regulations such as the European GDPR lay down many such requirements for digital technologies and processes to protect the privacy of users.

Attribute-based credentials (ABCs) are privacy enhancing-technologies that provide fine-grained control to users over the disclosure of personal attributes (enforce data minimisation and contextual identity usage) and can be realised in a decentralised setting when stored on users' personal devices such as, smartphones (enforce user control). So, ABCs are very good alternatives to the technologies (widely) used in identity management that are not built with privacy of users in mind. The research project, IRMA at Radboud University, has long strived to make the use of ABCs more practical and has led to the development of an efficient implementation of ABCs on smartphones. This thesis is a part of the academic research carried out within the IRMA project and focuses on finding new ways to increase the uptake of ABCs among users, credential issuers and service providers. At the beginning of the thesis, we set out to achieve two goals: first, to explore new applications of attribute-based credentials that are relevant in the real-world use case scenarios; second, to strengthen the security and trustworthiness of ABCs stored on users' smartphones. These goals help to achieve the final objective of putting privacy-friendly contextual identities via ABCs in practice.

In the first part of the thesis, ABC applications are considered. Chapter 3 introduces timestamped attribute-based signatures which can be securely used in practice along with attribute-based authentication with the same set of attributes. We call them IRMA signatures. The targeted application for IRMA signatures is conventional digital signing in which immediate verifiers need not be present. They can be used for contextual or role-based signing which may or may not identify the signer. Thus, they provide more flexibility than the existing counterparts such as public-key signatures while providing similar security guarantees. Chapter 4 shows how an attribute-based approach can be applied to Internet transactions such as webshopping to achieve minimum data disclosure from purchasers. Under this approach, the interactions between the participants of a transaction involve either issuance of or use of ABCs. The number and nature of attributes contained in the credentials are tailored for the interaction context to ensure that the minimum data of the purchaser is issued and disclosed. A purchaser can use ABCs for creating attribute-based signatures that will authenticate her to the verifier, preserve the integrity of the signed message and prevent repudiation by the purchaser. As there are immediate verifiers in transactions similar to authentication sessions, ABSs used in this chapter do not require external timestamps like IRMA signatures do. With these applications, we show that attribute-based credentials can be used for more than just authenticating users.

In the second part of the thesis, the techniques to strengthen the security and trust assurance of ABCs in real-world use cases are considered. When users' ABCs are stored and used on smartphones (and not on tamper-resistant platforms like smartcards), they need to provide security assurances both to users and to verifiers. Security assurance to users means two things: (i) users must be aware of every use of an ABC from her phone and, (ii) users must be able to block the usage at any time, for instance, when she loses her phone or notices some anomolous use of ABCs from her phone. Security assurance for verifiers means that the verifiers must be sure that the attributes received by a user during an authentication session or in a signature truly belong to that user. We explored some technical solutions that can provide such assurances to the users and verifiers in an attribute-based credential system. Chapter 5 proposes Tandem to secure the users' secret keys (bound to ABCs) using a central server while preserving the privacy of users towards this server. This allows users to instantaneously block their keys at the server upon loss or compromise of their phones. Furthermore, the server can limit the number of times a user's key is used in a specific period. When all the users in the system use this central server during credential issuance and verification, verifiers also get some level of assurance about the authentications and signatures made by the users with their ABCs. This is because, when the server is involved, a blocked user, a rate-limited user or any user without the knowledge of correct PIN of the ABC owner cannot complete an authentication successfuly. Chapter 6 focuses on secure self-enrolment (SSE) of users in ABC systems. The proposed SSE methods involve the online conversion of the existing identity information of users at public or private parties (e.g. banks, universities) into trustworthy ABCs on users' IRMA apps after identity proofing of the users. These methods are not only convenient for users and cost-efficient for the enrollers and issuers but they also achieve strong binding between the user's real identity and the ABC issued to the smartphone of the user. Such enrolments establish confidence in the claimed identity of the users (via the disclosed attributes during authentication or signing) among verifiers.

The proposals made in the thesis such as attribute-based signatures for digital signing, basic key-sharing solution for protecting the users ABC-secret-keys using a trusted central server and some of the secure self-enrolment methods are implemented and put into practice within the IRMA ecosystem by the Privacy by Design foundation. The research made by the author in this thesis has widened the applicability of ABCs (via ABSs and its innovative use in transactions) and has resulted in improving their practicality (via SSE and user-driven key blocking features). With this, we hope to achieve a wider use of contextual identities through ABCs in online environments.

As a general conclusion, IRMA is a flexible privacy-enhancing technology that has benefits to all the stakeholders in the system. An IRMA app hosts authentic ABCs containing a variety of context-dependant, personal attributes of a user issued by different (authorised) issuers. Users can conveniently use their IRMA apps to authenticate or sign with their ABCs in different contexts. The IRMA app is designed to enable users' awareness and control over the disclosure of their personal attributes. It embeds our concept of privacy that is described in Chapter 1. IRMA also benefits verifiers because they receive reliable and authentic user information via ABCs. The issuers who possess trustworthy identity information about users can perform both as enroller and issuer and carry out secure self-enrolment. This can be offered by the issuers as a trusted service to IRMA users and verifiers. In sum, practical and privacy-preserving technologies such as IRMA are very relevant in the current digital arena where the public and laws are demanding privacy and user control. It would be beneficial to businesses if they could take these demands into account and use such technologies in their operations.

7.1 Recommendations for Future Work

In this section, we provide some chapter-specific recommendations and also some general future directions for IRMA.

Chapter 3 presents the concept of attribute-based signatures (ABSs) and a practical set-up for putting them in practice. This chapter envisions the use of ABSs in conventional digital signing such as signing agreements, application forms etc. In such applications, the signature itself and the disclosed attributes are verifiable by any verifier at any time after the signature is made. That is why we include a customised timestamp token in every ABS. ABSs provide all the security guarantees of a typical digital signature (similar to public-key signatures) along with privacy and flexibility to signers, and authentic attributes to verifiers. Thus, they become very interesting in a legal context. As a possible future interdisciplinary research, ABSs could be analysed from a legal perspective to see how they can fit into standard regulations such as eIDAS for electronic signatures. Furthermore, for wide adoption of IRMA signatures in the real world, we need some support infrastructure other than the IRMA signature generation by the IRMA app and signature verification by a verifier terminal. Here we provide two examples for such support infrastructure: (i) a signature requestor app that can be used by the parties who wish to request signers to sign messages with certain identity attributes, and, (ii) signature verification features embedded in email clients: this allows a signature requestor's email client to verify the IRMA signatures received by signers via email. An example scenario where such a requestor app is useful is when a secretary requests his boss to sign an official document with the boss's name and designation attributes. The boss signs the document with her IRMA app and sends to the secretary or to any other third party via email. It is convenient for anyone who receives this IRMA signature if their email client verifies the signature and tells them if it is valid, similarly to PGP implementations[1]. The development of such features is subject to future work.

Chapter 4 has a detailed list of future work to be done in the context of attribute-based webshopping transactions. But in general, the attribute-based approach could be applied and analysed for other transactions such as ridesharing, online donations etc., to encourage a data-minimised and a privacy-friendly way of carrying out transactions on the Internet.

Chapter 5 presents two key-sharing solutions (basic key-sharing and Tandem) to protect a user's secret key as an alternative to storing entire secret keys on the user's (insecure) device. Both solutions share the key between the user's smart phone and a central server. Currently, the basic key-sharing is implemented in IRMA. However, when IRMA is used in real-world use cases in which the key-sharing central server cannot be fully trusted with respect to the privacy of users, then the basic key-sharing should be replaced by a more privacy-friendly solution such as Tandem. In this chapter, we have assumed that the central server is always available to run threshold protocols with the user's phone and to enforce blocking whenever requested by the user. To ensure availability in the absence of this assumption, the current setup can be extended such that the user's key is shared between a user's device and multiple servers. The adaptation of the key-sharing solutions to the new distributed setup and subsequent security analysis on them are subject to future work.

Chapter 6 proposes many ways for users to remotely get ABCs that are strongly bound to their real identity. It would be interesting to have public authorities such as muncipalities, banks, MNOs etc., carry out the identity-proofing and issuance of attribute-based credentials to users. In principle, authentication with ABCs in an ecosystem such as IRMA can achieve the highest level of assuarance (LoA). So, when IRMA reaches a stage at which all the management and organisational requirements for high assurance levels are met according to eIDAS standard, a complete analysis and LoA mapping needs to be performed to assess the overall authentication assurance level of IRMA ABCs.

7.1.1 Other future directions for IRMA

IRMA identity for 'things'. Internet of things (IoT) embodies the idea that many devices that are used in a specific context (e.g. inside a house, inside a factory) are connected to each other and to the Internet. The sensors in IoT devices in houses can collect a lot of privacy-sensitive data. It would be preferable if house owners have complete control over the sharing of such data [139]. Below we discuss some scenarios in which attribute-based identities can be assigned to IoT devices to enable user control and minimise privacy harms to the humans who own these devices or

[1] `https://www.enigmail.net/index.php/en/user-manual/signature-and-encryption` [last accessed: August 27, 2018]

are living in their vicinity.

- Robots: Home robots are becoming commonplace nowadays, they are used for many applications ranging from cleaning to elderly care. There are many situations in which a robot communicates with external information systems and servers over the Internet. For instance, if the robot needs a cloud server's assistance to perform some complex computation or it needs maintenance from its manufacturer. Then it has to first authenticate itself so that the external system (verifier) is convinced of the robot's true identity. If the robot identifies itself as well as its human user to the verifier during authentication, then all the queries, data collected from the user or from the living environment get linked to the user's identity at the verifier. This shows that the human user can easily be profiled by the verifier based on the information shared by her robot and this raises privacy concerns [140]. According to general robotic ethics, security and privacy are cornerstones for the wide adoption of cloud robotics [141]. It is possible to minimise privacy harms by enforcing data minimisation and purpose limitation during data collection as well as during authentication. When a context does not require identification of the user or the robot, then the unique identifiers should be stripped off the data and the authentication credentials that is sent out by the robot. That is, the data should have no references to the user's identity and the authentication credentials of the robot should not uniquely identify the robot to the verifier.

  Privacy-preserving ABCs could be an option to represent robot identity data and enforce context-based authentication. We present an example scenario to illustrate the idea above. Let us consider a home robot that is manufactured by the company Kuka is purchased by Mr.X. At the time of purchase, Kuka issues an identity credential to the robot with a random robot ID = 1234, manufacturer name = 'Kuka', owner's name = 'Mr.X' and contact = 'mrx@uhoo.com' attributes. Now this robot identity can be used for the following purposes. If the robot needs to prove that it belongs to Mr.X, then it authenticates by disclosing the owner's name attribute. If the robot contacts Kuka requesting maintenance, then Kuka may require the robot to authenticate with its robot ID and the owner's contact attribute. The ID attribute may be used to log the issues and maintenance status of the robot and the contact attribute can be used to convey the maintenance costs to the owner. The above two scenarios involve identified authentication. If the robot has to authenticate to Kuka's cloud server to query some information or outsource some data processing then it only authenticates with the manufacturer-name attribute (assuming for instance that the Kuka cloud server serves only their own robots free of charge). This is a case of non-identifying authentication. As we can see from this example, ABCs encourage data-minimised interaction of home robots with external entities on the Internet, thereby preserving the privacy of the users.

- Quite similar to the use case above in which robots are assigned with identity attributes, electronic equipments such as energy meters installed in houses may be assigned with some attributes such as meter ID, house address, house owner contact details. The meter can periodically send signed messages with meter readings and meter ID attributes to the energy company for the purpose of billing. This is a potential application of attribute-based signatures. The ABSs sent by meters allow the company to verify the authenticity of the readings as well as the attributes of the energy meter. If there are abnormal readings coming from the meter, then the company can request the meter to disclose the owner's contact attribute so that it can inform the owner about the issues with the energy meter.

Standardising ABCs. The use of ABCs has now become simple, practical, accessible and user-friendly through the IRMA's identity platform that consists of the smartphone application and the infrastructure to securely issue, verify and block the use of ABCs. Some efforts could be undertaken in future towards the inclusion of IRMA within a standard identity management framework (e.g. eIDAS [127]) and the establishment of best practices for this attribute-based technology. This could provide the much needed affirmation to public organisatons and businesses to adopt attribute-based authentication and signatures in their use cases.

Usability studies. Several usability evaluations of the IRMA application have already taken place so far. In particular, there have been expert evaluations, where one or more usability experts inspect the user interface of the app for potential issues. Furthermore, two usability tests with users have been conducted as a part of the IRMA project.

- The first is an informal user test that was conducted with students at the Computer Science department at the Radboud University as the test subjects. The students used the IRMA app to prove their attendance for a course's lectures at the university (replacing the need for a physical signature on a paper attendance list). To be able to do this, the students had to first register to the app, obtain credentials and use the credentials to authenticate to a test verifier to prove their attendance. After the completion of the course, students were invited to fill in a SUS questionnaire [142] and 30 out of approximately 90 actively participating students responded. The resulting SUS score was a 75.9. We can interpret this score in the light of the research by Bangor et al. [143], who have investigated the relationship between SUS ratings and ratings in the form of adjectives such as 'good','ok', or 'poor'. In their study, they found that a mean score of 75.9 would correspond to an adjective rating between 'good' and 'excellent' – although, closer to 'good'.
- Second, a more formal usability test on the IRMA app was conducted recently, with 5 people who were observed while completing several tasks pertaining to the app functionality and subsequently interviewed. As a part of the interview, the users were asked some questions to find out what kind of mental models are evoked in users based on their experience with the IRMA app. Basically, this was done to check if the app interface evokes the right mental model for privacy-enhancing features of IRMA such as selective disclosure and data minimisation [144]. These test results are currently being analysed by a usability expert who is actively involved in the IRMA project.

However, the usability of the IRMA app with both its authentication and digital signing functions must be evaluated on a larger and even more diverse group of users in future. Such usability tests and studies will give some estimates of user acceptance rates and also ideas to further improve the usability of the application.

Bibliography

[1] Monica Chew and Sid Stamm. Contextual identity: Freedom to be all your selves. In *Proceedings of the Workshop on Web*, volume 2. Citeseer, 2013.

[2] Helen Nissenbaum. Privacy as contextual integrity. *Washington Law Review*, 79:119, 2004.

[3] Daniel J Solove. I've got nothing to hide and other misunderstandings of privacy. *San Diego Law Review*, 44:745–772, 2007.

[4] Ann Cavoukian. Privacy by Design - The 7 Foundational Principles. *Take the challenge. Information and privacy commissioner of Ontario, Canada*, 2010.

[5] J Backhouse. D4.1: Structured account of approaches on interoperability. *FIDIS Deliverables*, 1:108, 2005.

[6] Claudia Diaz and Seda Gürses. Understanding the landscape of privacy technologies. *Extended abstract of invited talk in proceedings of the Information Security Summit*, pages 58–63, 2012.

[7] Martina Angela Sasse. Usability and trust in information systems. Edward Elgar, 2005.

[8] European Parliament and European Council. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). *Official Journal of the European Union 119 (Apr.2016)*, pages 1– 88, 2016.

[9] Lawrence Lessig. Code is law: On liberty in cyberspace. *Harvard Magazine*, (January-February):1–2, 2000.

[10] Anne Adams and Martina Angela Sasse. Privacy in multimedia communications: Protecting users, not just data. In *People and Computers XV—Interaction without Frontiers*, pages 49–64. Springer, 2001.

[11] GW Van Blarkom, JJ Borking, and JGE Olk. Handbook of Privacy and Privacy-Enhancing Eechnologies. *Privacy Incorporated Software Agent (PISA) Consortium, The Hague*, 198, 2003.

[12] Luís T. A. N. Brandão, Nicolas Christin, George Danezis, and anonymous. Toward mending two nation-scale brokered identification systems. *PoPETs*,

2015(2):135–155, 2015.

[13] Stefan Brands and Christian Paquin. U-Prove cryptographic specification v1.0. Technical report, Microsoft Corporation, March 2010.

[14] IBM Research Zürich Security team. Specification of the Identity Mixer Cryptographic Library – Version 2.3.4. Research report, IBM Research, Zürich, Feb 2012.

[15] Jan Camenisch and Els Van Herreweghen. Design and implementation of the *Idemix* anonymous credential system. In *CCS 2002*, pages 21–30. ACM, 2002.

[16] Stefan A Brands. *Rethinking public key infrastructures and digital certificates: building in privacy.* MIT Press, 2000.

[17] Eric R. Verheul. Self-Blindable Credential Certificates from the Weil Pairing. In *Asiacrypt 2001*, volume 2248 of *LNCS*, pages 533–551. Springer, 2001.

[18] Jan Camenisch and Anna Lysyanskaya. A Signature Scheme with Efficient Protocols. In *SCN 2002*, volume 2576 of *LNCS*, pages 268–289. Springer, 2002.

[19] Pim Vullers and Gergely Alpár. Efficient selective disclosure on smart cards using Idemix. In *IDMAN 2013*, volume 396 of *IFIP AICT*, pages 53–67. Springer, 2013.

[20] Gergely Alpár, Fabian van den Broek, Brinda Hampiholi, Bart Jacobs, Wouter Lueks, and Sietse Ringers‡. IRMA: practical, decentralized and privacy-friendly identity management using smartphones. *HotPETs 2017*, 2017.

[21] Jan Camenisch, Abhi Shelat, Dieter Sommer, Simone Fischer-Hübner, Marit Hansen, Henry Krasemann, Gérard Lacoste, Ronald Leenes, and Jimmy C. Tseng. Privacy and identity management for everyone. In *Proceedings of the 2005 Workshop on Digital Identity Management, Fairfax, VA, USA, 2005*, pages 20–27. ACM, 2005.

[22] Brinda Hampiholi, Gergely Alpár, Fabian van den Broek, and Bart Jacobs. Towards practical attribute-based signatures. In *Security, Privacy, and Applied Cryptography Engineering - 5th International Conference, SPACE 2015, Jaipur, India, October 3-7, 2015, Proceedings*, volume 9354 of *LNCS*, pages 310–328. Springer, 2015.

[23] Brinda Hampiholi and Gergely Alpár. Privacy-preserving webshopping with attributes. In *IEEE Symposium on Privacy-Aware Computing, PAC 2017, Washington, DC, USA, August 1-4, 2017*, pages 25–36. IEEE, 2017.

[24] Wouter Lueks, Brinda Hampiholi, Gergely Alpár, and Carmela Troncoso. TANDEM: Securing Keys by Using a Central Server While Preserving Privacy. Available at `https://www.dropbox.com/s/48u8s04c6kdcop2/tandem.pdf` and the updated version is available at `https://arxiv.org/abs/1809.03390`.

[25] Fabian van den Broek, Brinda Hampiholi, and Bart Jacobs. Securely derived identity credentials on smart phones via self-enrolment. In *Security and Trust Management - 12th International Workshop, STM, 2016, Heraklion, Crete,*

Greece, September 26-27, 2016, Proceedings, volume 9871 of *LNCS*, pages 106–121. Springer, 2016.

[26] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In Jr. Kaliski, BurtonS., editor, *Advances in Cryptology - CRYPTO 1997*, volume 1294 of *LNCS*, pages 410–424. Springer Berlin Heidelberg, 1997.

[27] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[28] David Pointcheval. The composite discrete logarithm and secure authentication. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1751 of *LNCS*, pages 113–128. Springer Berlin Heidelberg, 2000.

[29] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO '86*, volume 263 of *LNCS*, pages 186–194. Springer, 1986.

[30] Anna A. Lysyanskaya. *Signature schemes and applications to cryptographic protocol design*. PhD thesis, Massachusetts Institute of Technology, September 2002.

[31] David Chaum. Blind signatures for untraceable payments. In *Advances in cryptology*, pages 199–203. Springer, 1983.

[32] Pim Vullers. *Efficient implementations of attribute-based credentials on smart cards*. PhD thesis, Radboud University, Nijmegen, The Netherlands, 2014.

[33] Jonathan Katz. *Digital signatures*. Springer Science & Business Media, 2010.

[34] D Cooper, S Santesson, S Farrell, S Boeyen, R Housley, and W Polk. RFC 5280 - Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile. Technical Report May, IETF, 2008.

[35] S Farrell, R Housley, and S Turner. Rfc 5755 - an internet attribute certificate profile for authorization. Technical Report January, IETF, 2010.

[36] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011*, volume 6558 of *LNCS*, pages 376–392. Springer, 2011.

[37] Privacy by design foundation. `https://privacybydesign.foundation/irma-en/`. Accessed: 2017-11-07.

[38] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 614–629. Springer, 2003.

[39] Jonathan Katz, Rafail Ostrovsky, and Michael O. Rabin. Identity-based zero knowledge. In *SCN 2004*, volume 3352 of *LNCS*, pages 180–192. Springer, 2004.

[40] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, 2006.

[41] Tatsuaki Okamoto and Katsuyuki Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 35–52. Springer, 2011.

[42] Javier Herranz, Fabien Laguillaumie, Benoît Libert, and Carla Ràfols. Short attribute-based signatures for threshold predicates. In *Topics in Cryptology - CT-RSA 2012*, volume 7178 of *LNCS*, pages 51–67. Springer, 2012.

[43] Guo Shaniqng and Zeng Yingpei. Attribute-based signature scheme. In *Information Security and Assurance, 2008. ISA 2008*, pages 509–511. IEEE, 2008.

[44] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT '91*, volume 547 of *LNCS*, pages 257–265. Springer, 1991.

[45] Sarah Meiklejohn. An exploration of group and ring signatures. *UCSD Research Exam*, 2011.

[46] Gergely Alpár and Jaap-Henk Hoepman. A secure channel for attribute-based credentials: [short paper]. In *DIM'13, Proceedings of the 2013 ACM Workshop on Digital Identity Management, Berlin, Germany, November 8, 2013*, pages 13–18. ACM, 2013.

[47] Dalia Khader. Attribute based group signatures. *IACR Cryptology ePrint Archive*, 2007:159, 2007.

[48] Jan Camenisch, Ioannis Krontiris, Anja Lehmann, Gregory Neven, Christian Paquin, Kai Rannenberg, and Harald Zwingelberg. D2. 1 Architecture for Attribute-based Credential Technologies–Version 1. *ABC4Trust Deliverable D*, 2, 2011.

[49] Jan Camenisch, Stephan Krenn, Anja Lehmann, Gregory Neven, Gert Læssøe Mikkelsen, and Michael Østergaard Pedersen. D3.1 Scientific Comparison of ABC Protocols – Part I – Formal Treatment of Privacy-Enhancing Credential Systems. *ABC4Trust Deliverable D*, 2014.

[50] Jan Camenisch, Stephan Krenn, Anja Lehmann, Gert Læssøe Mikkelsen, Gregory Neven, and Michael Østergaard Pedersen. Formal treatment of privacy-enhancing credential systems. In *International Conference on Selected Areas in Cryptography*, pages 3–24. Springer, 2015.

[51] Federal Information Processing Standards. FIPS 202 - SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. Technical Report August, FIPS, 2015.

[52] Keccak team. Note on keccak parameters and usage. `http://keccak.noekeon.org/NoteOnKeccakParametersAndUsage.pdf`. Online; accessed 6-July-2015.

[53] Barker, Elaine. NIST Special Publication 800-102 – Recommendation for Digital Signature Timeliness. Technical report, NIST, 2009. Available at `https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-102.pdf`.

[54] ANSI ASC X9.95 Standard: Trusted Timestamp Management and Security. Technical Report X9.95, Accredited Standards Committee, 2005. Available at `http://standards.globalspec.com/std/750866/asc-x9-x9-95`.

[55] Jan Camenisch and Anna Lysyanskaya. An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer Berlin / Heidelberg, 2001.

[56] S Santesson, M Myers, R Ankney, A Malpani, S Galperin, and C Adams. Rfc 6960 - x.509 internet public key infrastructure online certificate status protocol (ocsp). Technical Report June, IETF, 2013.

[57] Jorn Lapon, Markulf Kohlweiss, Bart De Decker, and Vincent Naessens. Analysis of Revocation Strategies for Anonymous Idemix Credentials. In *Communications and Multimedia Security, 12th IFIP TC 6 / TC 11 International Conference, CMS 2011, Ghent, Belgium, 2011. Proceedings*, volume 7025 of *LNCS*, pages 3–17. Springer, 2011.

[58] Wouter Lueks, Gergely Alpár, Jaap-Henk Hoepman, and Pim Vullers. Fast revocation of attribute-based credentials for both users and verifiers. *Computers & Security*, 67:308–323, 2017.

[59] Gergely Alpár and Bart Jacobs. Credential design in attribute-based identity management. 2013.

[60] Paul M Schwartz and Daniel J Solove. The pii problem: Privacy and a new concept of personally identifiable information. *New York University Law Review*, 86:1814, 2011.

[61] Kyle Soska and Nicolas Christin. Measuring the longitudinal evolution of the online anonymous marketplace ecosystem. In *24th USENIX Security Symposium, Washington, D.C., USA, August 12-14, 2015.*, pages 33–48. USENIX Association, 2015.

[62] Rhys Smith and Jianhua Shao. Privacy and e-commerce: a consumer-centric perspective. *Electronic Commerce Research*, 7(2):89–116, 2007.

[63] Jesus Diaz, Seung Geol Choi, David Arroyo, Angelos D Keromytis, Francisco B Rodriguez, and Moti Yung. Privacy threats in e-shopping (position paper). In *International Workshop on Data Privacy Management*, pages 217–225. Springer, 2015.

[64] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *CRYPTO '88*, volume 403 of *LNCS*, pages 319–327. Springer, 1990.

[65] Stefan Brands. Electronic cash on the Internet. In *NDSS '95*, pages 64–84. IEEE Computer Society, 1995.

[66] Gesine Hinterwälder, Felix Riek, and Christof Paar. Efficient E-cash with Attributes on MULTOS Smartcards. In *International Workshop on Radio Frequency Identification: Security and Privacy Issues*, pages 141–155. Springer, 2015.

[67] Georg Fuchsbauer and Markulf Kohlweiss. Anonymous transferable e-cash. In

PKC 2015: 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, 2015, Proceedings, volume 9020, page 101. Springer, 2015.

[68] Qing Zhang, Konstantinos Markantonakis, and Keith Mayes. A practical fair-exchange e-payment protocol for anonymous purchase and physical delivery. In *IEEE International Conference on Computer Systems and Applications, 2006.*, pages 851–858. IEEE, 2006.

[69] Fahad A Alqahtani. A fair exchange & customer anonymity protocol using a trusted third party for electronic commerce transactions & payments. *International Journal of Network Security & Its Applications*, 6(1):59, 2014.

[70] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 302–321. Springer, 2005.

[71] Ian Miers, Christina Garman, Matthew Green, and Aviel D. Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *S&P 2013*, pages 397–411. IEEE Computer Society, 2013.

[72] Jan-Erik Ekberg, Kari Kostiainen, and N. Asokan. The Untapped Potential of Trusted Execution Environments on Mobile Devices. *IEEE Security & Privacy*, 12(4):29–37, 2014.

[73] Claudio Marforio, Nikolaos Karapanos, Claudio Soriente, Kari Kostiainen, and Srdjan Capkun. Secure enrollment and practical migration for mobile trusted execution environments. In *SPSM '13*, pages 93–98. ACM, 2013.

[74] Ravi S. Sandhu and Xinwen Zhang. Peer-to-peer access control architecture using trusted computing technology. In *SACMAT 2005*, pages 147–158. ACM, 2005.

[75] Brian McGillion, Tanel Dettenborn, Thomas Nyman, and N. Asokan. Open-TEE - An Open Virtual Trusted Execution Environment. In *IEEE TrustCom*, pages 400–407. IEEE, 2015.

[76] Android security website. Developing third party applications with Trusty TEE. `https://source.android.com/security/trusty/#third-party_trusty_applications`.

[77] Sanchari Das, Andrew Dingman, and L Jean Camp. Why Johnny Doesn't Use Two Factor A Two-Phase Usability Study of the FIDO U2FSecurity Key. In *FC 2018*, LNCS. Springer, 2018.

[78] Alex Hern. Stagefright: new Android vulnerability dubbed 'heartbleed for mobile'. *The Guardian*, 2015.

[79] Victor van der Veen, Yanick Fratantonio, Martina Lindorfer, Daniel Gruss, Clémentine Maurice, Giovanni Vigna, Herbert Bos, Kaveh Razavi, and Cristiano Giuffrida. Drammer: Deterministic Rowhammer Attacks on Mobile Platforms. In *CCS 2016*, pages 1675–1689. ACM, 2016.

[80] Kim Zetter, WIRED magazine. How the top 5 PC makers open your laptop to hackers. `https://www.wired.com/2016/05/2036876/`.

[81] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas,

Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown. *ArXiv e-prints*, Jan 2018.

[82] Husam Al Jawaheri, Mashael Al Sabah, Yazan Boshmaf, and Aiman Erbad. When A small leak sinks A great ship: Deanonymizing tor hidden service users through bitcoin transactions analysis. *CoRR*, abs/1801.07501, 2018.

[83] Man Ho Au, Willy Susilo, and Yi Mu. Constant-Size Dynamic k-TAA. In *SCN 2006*, volume 4116 of *LNCS*, pages 111–125. Springer, 2006.

[84] Man Ho Au, Patrick P. Tsang, and Apu Kapadia. PEREA: Practical TTP-free revocation of repeatedly misbehaving anonymous users. *ACM TISSEC*, 14(4):29:1–29:34, 2011.

[85] Patrick P. Tsang, Man Ho Au, Apu Kapadia, and Sean W. Smith. BLAC: Revoking Repeatedly Misbehaving Anonymous Users without Relying on TTPs. *ACM TISSEC*, 13(4):39:1–39:33, 2010.

[86] Jan Camenisch, Susan Hohenberger, Markulf Kohlweiss, Anna Lysyanskaya, and Mira Meyerovich. How to win the clone wars: efficient periodic n-times anonymous authentication. In *CCS 2006*, pages 201–210. ACM, 2006.

[87] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems. *J. Cryptology*, 20(1):51–83, 2007.

[88] Victor Shoup. Practical threshold signatures. In *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 207–220. Springer, 2000.

[89] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO 1984*, volume 196 of *LNCS*, pages 10–18. Springer, 1984.

[90] Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *J. Cryptology*, 15(2):75–96, 2002.

[91] Philip D. MacKenzie and Michael K. Reiter. Two-party generation of DSA signatures. *International Journal for Information Security*, 2(3-4):218–239, 2004.

[92] Rosario Gennaro, Steven Goldfeder, and Arvind Narayanan. Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security. In *ACNS 2016*, volume 9696 of *LNCS*, pages 156–174. Springer, 2016.

[93] Timothy G. Abbott, Katherine J. Lai, Michael R. Lieberman, and Eric C. Price. Browser-Based Attacks on Tor. In *PETS 2007*, volume 4776 of *LNCS*, pages 184–199. Springer, 2007.

[94] Lasse Øverlier and Paul F. Syverson. Locating Hidden Servers. In *S&P 2006*, pages 100–114. IEEE Computer Society, 2006.

[95] Ero Balsa, Carmela Troncoso, and Claudia Díaz. OB-PWS: Obfuscation-Based Private Web Search. In *IEEE S&P*, pages 491–505. IEEE Computer Society, 2012.

[96] Richard Chow and Philippe Golle. Faking contextual data for fun, profit, and

privacy. In *WPES*, pages 105–108. ACM, 2009.

[97] Ania M. Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. The Loopix Anonymity System. In *USENIX 2017*, pages 1199–1216. USENIX Association, 2017.

[98] Roger Dingledine, Nick Mathewson, and Paul F. Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, pages 303–320. USENIX Association, 2004.

[99] Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *EUROCRYPT '99*, volume 1592 of *LNCS*, pages 223–238. Springer, 1999.

[100] Marc Joye and Benoît Libert. Efficient Cryptosystems from 2^k-th Power Residue Symbols. In *EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 76–92. Springer, 2013.

[101] Taher ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[102] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO 1991*, volume 576 of *LNCS*, pages 129–140. Springer, 1991.

[103] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In *Advances in Cryptology – ASIACRYPT96*, volume 1163 of *LNCS*, pages 252–265. Springer Berlin Heidelberg, 1996.

[104] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.

[105] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *Annual Symposium on Foundations of Computer Science 1986*, pages 162–167. IEEE Computer Society, 1986.

[106] Xiao Wang, Samuel Ranellucci, and Jonathan Katz. Authenticated garbling and efficient maliciously secure two-party computation. In *ACM SIGSAC 2017*, pages 21–37. ACM, 2017.

[107] Carlos Aguilar-Melchor, Joris Barrier, Laurent Fousse, and Marc-Olivier Killijian. XPIR : Private Information Retrieval for Everyone. *PoPETs*, 2016(2):155–174, 2016.

[108] Raphael R. Toledo, George Danezis, and Ian Goldberg. Lower-Cost ϵ-Private Information Retrieval. *PoPETs*, 2016(4):184–201, 2016.

[109] Chris J. Mitchell. What is trusted computing? In *Trusted Computing*, volume 6. Institution of Engineering and Technology, 2005.

[110] Yvo Desmedt. Society and Group Oriented Cryptography: A New Concept. In *CRYPTO 1987*, volume 293 of *LNCS*, pages 120–127. Springer, 1987.

[111] Colin Boyd. Digital multisignatures. *Cryptography and Coding*, pages 241–246,

1989.

[112] Yvo Desmedt and Yair Frankel. Shared Generation of Authenticators and Signatures (Extended Abstract). In *CRYPTO '91*, volume 576 of *LNCS*, pages 457–469. Springer, 1991.

[113] Tal Rabin. A Simplified Approach to Threshold and Proactive RSA. In *CRYPTO '98*, volume 1462 of *LNCS*, pages 89–104. Springer, 1998.

[114] Rosario Gennaro, Tal Rabin, Stanislaw Jarecki, and Hugo Krawczyk. Robust and Efficient Sharing of RSA Functions. *Journal of Cryptology*, 13(2):273–300, 2000.

[115] Jesús F. Almansa, Ivan Damgård, and Jesper Buus Nielsen. Simplified Threshold RSA with Adaptive and Proactive Security. In *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 593–611. Springer, 2006.

[116] Roel Peeters, Svetla Nikova, and Bart Preneel. Practical RSA threshold decryption for things that think. In *WISSec 2008*, 2008.

[117] Carmit Hazay, Gert Læssøe Mikkelsen, Tal Rabin, and Tomas Toft. Efficient RSA Key Generation and Threshold Paillier in the Two-Party Setting. In *CT-RSA 2012*, volume 7178 of *LNCS*, pages 313–331. Springer, 2012.

[118] Erinn Atwater and Urs Hengartner. Shatter: Using Threshold Cryptography to Protect Single Users with Multiple Devices. In *WISEC 2016*, pages 91–102. ACM, 2016.

[119] Marcel Keller, Gert Læssøe Mikkelsen, and Andy Rupp. Efficient Threshold Zero-Knowledge with Applications to User-Centric Protocols. In *ICITS 2012*, volume 7412 of *LNCS*, pages 147–166. Springer, 2012.

[120] Philip D. MacKenzie and Michael K. Reiter. Networked Cryptographic Devices Resilient to Capture. In *S&P 2001*, pages 12–25. IEEE Computer Society, 2001.

[121] Jan Camenisch, Anja Lehmann, Gregory Neven, and Kai Samelin. Virtual Smart Cards: How to Sign with a Password and a Server. In *SCN 2016*, volume 9841 of *LNCS*, pages 353–371. Springer, 2016.

[122] Dan Boneh, Xuhua Ding, Gene Tsudik, and Chi-Ming Wong. A Method for Fast Revocation of Public Key Certificates and Security Capabilities. In *10th USENIX Security Symposium, August 13-17, 2001, Washington, D.C., USA*. USENIX Association, 2001.

[123] Dan Boneh, Xuhua Ding, and Gene Tsudik. Fine-grained control of security capabilities. *ACM TOIT*, 4(1):60–82, 2004.

[124] Benoît Libert and Jean-Jacques Quisquater. Efficient revocation and threshold pairing based cryptosystems. In *PODC 2003*, pages 163–171. ACM, 2003.

[125] Ahto Buldas, Aivo Kalu, Peeter Laud, and Mart Oruaas. Server-supported RSA signatures for mobile devices. In *ESORICS 2017*, volume 10492 of *LNCS*, pages 315–333. Springer, 2017.

[126] William E Burr, Donna F Dodson, Elaine M Newton, Ray A Perlner,

W Timothy Polk, Sarbari Gupta, and Emad A Nabbus. Electronic authentication guideline. NIST Special Publication 800-63-1. Technical report, 2011. Available at http://csrc.nist.gov/publications/nistpubs/800-63-1/SP-800-63-1.pdf.

[127] The European Parliament and the Council of the European Union. eIDAS - EU Regulation No 910/2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC, July 2014. Avaliable at http://eur-lex.europa.eu/eli/reg/2014/910/oj.

[128] European Commission. Ccommission implementing Regulation (EU) 2015/1502 on setting out minimum technical specifications and procedures for assurance levels for electronic identification means pursuant to Article 8(3) of Regulation (EU) No 910/2014 of the European Parliament and of the Council on electronic identification and trust services for electronic transactions in the internal market (Text with EEA relevance), September 2015. Avaliable at http://eur-lex.europa.eu/eli/reg_impl/2015/1502/oj.

[129] Antonia Rana and Luigi Sportiello. Implementation of security and privacy in ePassports and the extended access control infrastructure. *IJCIP*, 7(4):233–243, 2014.

[130] International Civil Aviation Organization. Document 9303 - Part 1 - Machine Readable Travel Documents. Technical Report 7th Edition, ICAO, 2015. Available at https://www.icao.int/publications/Documents/9303_p3_cons_en.pdf.

[131] International Civil Aviation Organization. Document 9303 - Security Mechanisms for Machine Readable Travel Documents (Part 11) . Technical Report 7th Edition, ICAO, 2014. Available at https://www.icao.int/Meetings/TAG-MRTD/TagMrtd22/TAG-MRTD-22_WP03-rev.pdf.

[132] BaFin - German Federal Financial Supervisory Authority. Circular 3/2017 (GW) - Video Identification Procedures (Reference Number: GW 1-GW 2002-2009/0002), April 2017. Avaliable at https://www.bafin.de/SharedDocs/Veroeffentlichungen/EN/Rundschreiben/2017/rs_1703_gw_videoident_en.html.

[133] BaFin - German Federal Financial Supervisory Authority. Money Laundering Act, July 2013. Avaliable at https://www.bafin.de/SharedDocs/Veroeffentlichungen/EN/Aufsichtsrecht/Gesetz/GwG_en.html.

[134] Defense Human Resource Activity (DHRA). Common Access Card (CAC) Security. Avaliable at http://www.cac.mil/Common-Access-Card/CAC-Security/.

[135] BSI - Federal Office for Information Security. German eID based on Extended Access Control v2 – Overview of the German eID system, February 2017. Avaliable at https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/EIDAS/German_eID_Whitepaper.pdf.

[136] Entrust Datacard. Mobile Derived PIV/CAC Credential - A Complete Solution For NIST 800-157, 2014. Avaliable at https:

//www.entrust.com/wp-content/uploads/2014/10/Mobile-Derived-Credential-WEB2-Nov15.pdf.

[137] European Commission. Guidance for the application of the levels of assurance which support the eIDAS Regulation. Avaliable at https://ec.europa.eu/cefdigital/wiki/download/attachments/40044784/Guidance%20on%20Levels%20of%20Assurance.docx?version=1&modificationDate=1488295895839&api=v2.

[138] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. Solving Revocation with Efficient Update of Anonymous Credentials. In *Security and Cryptography for Networks, 7th International Conference, SCN 2010, Amalfi, Italy, September 13-15, 2010. Proceedings*, volume 6280 of *LNCS*, pages 454–471. Springer, 2010.

[139] Gergely Alpár, Lejla Batina, Lynn Batten, Veelasha Moonsamy, Anna Krasnova, Antoine Guellier, and Iynkaran Natgunanathan. New directions in IoT privacy using attribute-based authentication. In *Proceedings of the ACM International Conference on Computing Frontiers*, pages 461–466. ACM, 2016.

[140] Ugo Pagallo. Robots in the cloud with privacy: A new threat to data protection? *Computer Law & Security Review*, 29(5):501–508, 2013.

[141] Patrick Lin, Keith Abney, and George A Bekey. Robot Ethics: The Ethical and Social Implications of Robotics (Intelligent Robotics and Autonomous Agents series), 2011.

[142] John Brooke et al. SUS-A quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[143] Aaron Bangor, Philip Kortum, and James Miller. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of usability studies*, 4(3):114–123, 2009.

[144] Erik Wästlund, Julio Angulo, and Simone Fischer-Hübner. Evoking comprehensive mental models of anonymous credentials. In *IFIP WG 11.4 International Workshop, iNetSec 2011, Lucerne, Switzerland, 2011, Revised Selected Papers*, volume 7039 of *LNCS*, pages 1–14. Springer, 2011.

Curriculum Vitae

Brinda Hampiholi

June 20, 1989 Born in Bangalore, India

July 2005 - June 2007 Pre-university education
Physics, Chemistry, Mathematics and Electronics (PCME) Profile
Vijaya Composite PU College, Bangalore, India

July 2007 - August 2011 Bachelor of Engineering (B.E)
Computer Science and Engineering
Visvesvaraya Technological University, Bangalore, India

September 2011 - August 2012
Network Consultant Engineer
Cisco systems Ltd, Bangalore, India

September 2012 - August 2014 Master of Science
Computer Science and Engineering, Security and Privacy track
EIT Digital Dual Degree Masters Programme
University of Trento, Italy and University of Twente, The Netherlands

September 2014 - August 2018 PhD
Digital Security
Radboud University, Nijmegen

September 2018 - Present
Research Scientist
Philips Research, Eindhoven, The Netherlands