ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Model-Based Analysis of Privacy in Electronic Services

Koen Decroix

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

October 2015

Model-Based Analysis of Privacy in Electronic Services

Koen DECROIX

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

October 2015

Acknowledgements

This Ph.D thesis presents the results of my doctoral research. The research results were only made possible by the contributions of other people. I would like to thank all of them.

First of all, I would like to thank my co-supervisor Prof. Vincent Naessens who convinced me to leave industry to start a Ph.D. I truly appreciate his advice, guidance and motivational discussions.

Secondly, I woud like to thank my supervisor Prof. Bart De Decker for his advice and feedback which helped me to improve my research.

Next, I would like to express my appreciation to Prof. Lieven De Strycker, Prof. Bart Preneel, Prof. Hans Vangheluwe, Prof. Marc Denecker and Daniel Le Métayer for participating in my jury and to give me feedback that improved the quality of my dissertation. Also, I thank Prof. Willy Sansen and Prof. Yves Willems for chairing my preliminary and public Ph.D defense, respectively.

I want to thank Erik Van Achter for proofreading my thesis and giving me advice that improved the quality of my thesis. My thanks also go to Milica Milutinovic, Denis Butin, and Joachim Jansen for our joint work. I also thank my colleagues and former colleagues of the MSEC research group Jorn Lapon, Faysal Boukayoua, Jan Vossaert, Laurens Lemaire, Michiel Willocx, and Vincent Raes for their entertaining contributions during the breaks and for the help during my Ph.D. Thank you to the members of the CODeS and SecAnon iMinds-DistriNet research groups as well.

Finally, I thank my family for supporting me during the last four years. Especially, I would like to express my gratitude towards my girlfriend Gerlinde, for her continuous support during the final hectic months of my Ph.D.

Koen Decroix, Ghent, September 2015

Abstract

Many stakeholders are involved in complex electronic services. Individuals release personal information to front-ends in advanced web services. However, to what extent the acquired information is distributed and processed, what other data is included and/or merged in user profiles, remains unclear. Much more personal information is often released than strictly necessary. As is well known, many authentication technologies release personal data and make transactions linkable. Service providers can accurately profile individuals and trade profiles with external commercial entities. Moreover, service providers delegate their responsibilities for the data they collected to external entities who receive these data, while consequently, individual users lose control of their personal information. The present dissertation presents a logic based modeling approach to inspect privacy in composite electronic services from different viewpoints in data protection.

The first part provides the background for the major concepts on privacy complemented by an overview of the different privacy analysis approaches currently available.

The second part presents a logic based modeling approach for inspecting the user privacy. First of all, the key concepts necessary to reason on privacy in composite electronic services are identified. Examples are privacy policies, authentication technologies; but also, the privacy preferences and trust assumptions of users. These concepts are mapped to components of a logic based framework supporting automated privacy inspection of composite electronic services. The approach extracts profiles, containing a set of personal data that can be compiled by service providers after a user has interacted with a particular service provider. The resulting profiles constitute the input for a qualitative privacy analysis capable of providing meaningful feedback to both end-users and service designers. The former can select a service based on the minimal data disclosure principle. The latter can use the framework to select privacy-enhancing technologies and define consistent privacy policies from the earliest design stage.

In the third part, the approach is validated on two electronic public transport ticketing systems. The case study shows the extensibility of the framework.

The final part focuses on service provider accountability, one of the core privacy protection principles of the upcoming European General Data Protection Regulation. Service provider accountability refers to the obligation of data controllers to demonstrate compliance of their data practices with the rules and regulations. In this part, an inference model providing individuals with global accountability guarantees of multi-component systems is introduced. It considers different user expectations on provided evidence (i.e. log evidence) of declared data handling practices. The feedback provided by this approach is useful for both end-users and auditors who conduct privacy assessments on behalf of the end-users.

Beknopte samenvatting

In deze thesis worden complexe elektronische services beschouwd waarin personen interageren met een publieke interface. Bij deze services zijn verschillende stakeholders betrokken die elk persoonlijke gegevens verzamelen van diegene die de services gebruiken. Het is echter onduidelijk welke persoonlijke gegevens er allemaal worden verspreid en verwerkt, en welke er voor het profileren van personen worden bijgehouden. In veel gevallen worden meer persoonlijke gegevens verzameld dan strikt noodzakelijk is. Vaak is dit te wijten aan authenticatietechnologieën die persoonlijke gegevens vrijgeven met als gevolg dat mogelijks verschillende service transacties met elkaar kunnen worden gelinkt. De verzamelde gegevens zorgen ervoor dat service providers gebruikersprofielen kunnen aanmaken die personen heel nauwkeurig beschrijven. Deze profielen kunnen dan worden verhandeld aan externe partijen zoals adverteerders. Bovendien delegeren service providers taken aan derde partijen die gespecialiseerd zijn in een bepaalde deeltaak (bijv. de authenticatie wordt verzorgd door de Facebook authenticatieservice). Het gevolg van deze gedistribueerde services is dat personen controle verliezen over hun persoonlijke gegevens. Het doel in deze thesis is om een privacy analyse methode te ontwikkelen die voor zowel eindgebruikers als service designers nuttige feedback oplevert.

In het eerste gedeelte van deze thesis worden de belangrijkste privacyconcepten besproken en wordt een overzicht gegeven van bestaande privacy analyse methodes.

Het tweede gedeelte in de thesis focust zich op het uitwerken van een op logica gebaseerde methode voor het analyseren van privacy in elektronische services. Eerst worden de concepten die cruciaal zijn voor het beschrijven van privacy in services geïdentificeerd. Voorbeelden hiervan zijn privacy policy's en authenticatietechnologieën, maar ook de persoonlijke privacyvoorkeuren en de vertrouwensperceptie van personen. Daarna worden de verschillende componenten, waartoe de bovenvermelde concepten behoren, van een op logica gebaseerd modelleerraamwerk gedefinieerd. Het raamwerk laat toe om automatisch te redeneren over de privacy in een gemodelleerde service. Hiervoor worden er profielen die een individu's persoonlijke gegevens bevat gecompileerd. Op basis van deze profielen wordt er een privacy analyse uitgevoerd die kwalitatieve feedback oplevert die nuttig is voor eindgebruikers en service designers. De eindgebruikers kunnen een service selecteren op basis van welke service het minst persoonlijke gegevens verzamelt. Service designers kunnen het raamwerk gebruiken voor het selecteren van een geschikte privacy verbeterende technologie (PET) en om consistente privacy policy's te bepalen vanaf de vroegste ontwerpfase.

In het derde gedeelte van de thesis wordt de privacy analyse methode gevalideerd m.b.v. ticketsystemen die in publiek transport worden gebruikt. De gevalstudie toont de uitbreidbaarheid van het modelleerraamwerk aan.

Het laatste gedeelte in de thesis focust zich op de accountability van service providers. Accountability is één van de pijlers is van de binnenkort geratificeerde Europese Algemene Data Protectie Richtlijn voor het beschermen van de privacy. Dit dataprotectieprincipe maakt datacontrollers verantwoordelijk om aan te tonen dat hun datapraktijken overeenstemmen met hun privacy policy's en met de privacywetgeving. In dit gedeelte van de thesis wordt er een analyse methode uitgewerkt die automatisch een overzicht genereert voor eindgebruikers over de globale accountability garanties in een systeem. De analyse houdt rekening met het vertrouwen van verschillende prototypische gebruikers over het geleverde bewijs (i.e. logs) van de dataverwerkingspraktijken van de service providers. De feedback die door deze methode wordt gegenereerd is nuttig voor eindgebruikers en auditeurs die in opdracht van een eindgebruiker privacy assessments uitvoeren.

Abbreviations

ABC	Attribute-Based Credential
AD	Advertiser
ASP	Answer Set Programming
CA	Certificate Authority
CARL	Credential-based Authentication Requirements Language
CP	Constraint Programming
CRL	Certificate Revocation List
CTL	Computation Tree Logic
DoB	Date of Birth
eID	Electronic Identity Card
ENISA	European Union Agency for Network in Information Security
EPAL	Enterprise Privacy Authorization Language
FIPP	Fair Information Practice Principles
FO	First-Order logic
GAP	Global Accountability Profile
GDPR	General Data Protection Regulation
Gov	Government
GS	Grocery Store
HIPAA	Health Insurance Portability and Accountability Act
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure

ID	Identifier
IP	Internet Protocol
IRMA	I Reveal My Attributes
ITU-T	International Telecommunication Union - Telecommunications sector
LA	Loyalty App provider
LP	Loyalty Program provider
LTC	Linear Time Calculus
MAC	Media Access Control
NAF	Negation As Failure
NFC	Near Field Communication
OCSP	Online Certificate Status Protocol
OECD	Organization for Economic Co-operation and Development
OR	Operation Research
OV-card	Dutch public transport card (Openbaar Vervoer kaart)
P3P	Platform for Privacy Preferences Project
PET	Privacy Enhancing Technology
PII	Personal Identifiable Information
PKI	Public Key Infrastructure
PPTS	Privacy-preserving Public transport Ticketing System
PS	Postal Service company
PTO	Public Transport Operator
QR code	Quick Response code
SAML	Security Assertion Markup Language
SAT	Propositional Satisfiability Problem
SDK	Software Development Kit
SDL	Software Development Life cycle
SM	SMartphone
SSN	Social Security Number
TCP	Transmission Control Protocol

TSO	Ticketing System Operator
URL	Uniform Resource Locater
WS	Web Shop
XACML	eXtensible Access Control Markup Language

Contents

List of Figures

List of Tables

Chapter 1

Introduction

At present, web services constitute an integral part of our daily lives. Initially, they were stand-alone applications offered by single service providers. Over time, they have evolved to complex web mashups, where individuals interact with a service front-end calling on underlying sub-services to perform specific tasks. Personal information revealed by individuals via the service front-end can be spread across multiple parties that collaborate to offer these services. Service providers can use the collected data to profile individuals, e.g. to offer personalized service content. Before individuals can use services, they must first agree with the privacy policies of service providers, and give them their explicit consent for the declared data handling practices. For example, by giving his consent when registering, the individual grants permission to the service provider to link his Facebook account to the service and collect personal information from Facebook. Because privacy policies typically consist of ambiguous declarations, individuals are left oblivious about the processing of their personal information. Moreover, privacy policies are mere declarations of intent and offering no guarantee of compliance with the actual data handling practices of service providers. Users fall back on trust perceptions [158, 77, 192] when selecting a particular web service. They are willing to release more personal information to service providers that are trustworthy.

Service providers may inadvertently collect more personal information than strictly necessary because of the technology used for authenticating users, such as in the case of an individual who proves he is older than 18 using his X.509 certificate based electronic identity card. Not only does he reveal his date of birth, but also his name, address, social security number and all other attributes that are included in the certificate. Service providers – driven by economical motives – may collaborate with external parties to intentionally collect and store more personal information than declared to accurately profile their users. They can trade the data with external parties, such as advertisers buying the data to spam individuals with offers. Another example are insurance companies who can refuse individuals based on health information they collected about them. The concerns of individuals and service providers are conflicting. It is important for individuals to have more control over their personal information revealed for protecting their privacy. Disclosing only the personal information which is strictly necessary, i.e. data minimization, is one strategy to protect an individual's privacy. It is a key data protection principle considered by the computer science and privacy law community.

One way to achieve minimal data disclosure are *privacy-enhancing technologies* (PETs), which originate from computer science. It is an umbrella term of cryptographic systems that assure individuals that only necessary personal attributes are disclosed. An example of a PET is Tor [74]. It is an anonymity network that ensures anonymity for those surfing the Internet. In a normal setting, individuals just browse on the Internet. Typically, they are not aware that when they access a website, they leave traces behind inherent to the underlying TCP/IP protocol used by the HTTP(S) protocol that is used by browsers. For example, the website can store the individual's IP address, or their Internet Provider can store the addresses they visit. Anonymity networks – such as Tor – prevent this by offering an anonymous overlay channel that hides the underlying linkable protocol data. Another example is U-Prove [139], which is an anonymous attribute based credential technology. U-Prove credentials are attribute containers certified by a trusted authority which support privacy-preserving user authentication – e.g. only selected attributes are disclosed – while offering strong authentication guarantees to service providers. For now, PETs are not widespread in current systems; they have a steep learning curve based on complex cryptographic operations. As a result, most system designers have little knowledge of PETs. Moreover, data is a crucial asset in many business models of commercial organizations, making PETs unattractive. Nevertheless, PETs alone are not effective methods for protecting privacy because in many cases service functionality requires that individuals reveal parts of their personal information. In such cases, privacy regulations are required as well.

The non-binding OECD's privacy guidelines [138] aim at harmonizing privacy regulations of member states and are adopted by the upcoming *European General Data Protection Regulation* (GDPR) [8]. The framework contains recommendations on privacy principles that should be adopted by data protection regulations of member states. Like PETs, the framework also considers *data minimization* as a data protection principle. More specifically, it restricts data controllers, e.g. service providers, to collect only the data that is necessary for the specified purpose. Furthermore, the framework also includes the *openness* and the *individual's participation* principle. The former implies that individuals must be informed about the data handling practices, e.g. via readily available privacy policies which contain data handling practice declarations. The latter includes the individual's right to access his data. An example is Google Dashboard [10] which provides an interface to individuals to manage the personal data that Google collected about them. Another recommendation is *accountability of data controllers*, which makes data controllers responsible for having data handling practices compliant with regulations and for demonstrating this compliance. As electronic services may involve multiple parties that are located in different member states, the OECD recommends member states to restrict *cross-border data flows* to countries that have sufficient privacy protection regulations. For example, cross-border data flows from EU companies to US companies are regulated by Safe Harbor [174, 174], which bridges the differences in data protection regulations between EU and US. US companies can only receive data from EU companies if they participated in the Safe Harbor program. To protect the personal information of an individual even more, the OECD's privacy guidelines and EU's GDPR add the *privacy by design* principle, which enforces data controllers to build in privacy safeguards throughout the design process from the earliest design stage, and not consider privacy as an afterthought. However, the regulations are too high level to be useful for designers [93]. Methods that assist designers are required that bridge the gap between regulations and the design/development of electronic services. Service providers may advertise their compliance with regulations towards (potential) users using privacy trust seals [3, 1] issued by external certifying organizations. This is encouraged by the EU's GDPR and may have the potential to gain trust by users.

1.1 Scope and Approach

The research in this thesis considers privacy in complex electronic services comprising a user front-end which interacts with underlying sub-services provided by multiple parties. Each service component can collect personal information revealed by individuals and can store it, process it, or forward it to collaborating third-parties to profile individuals. The aforementioned scheme is exploited by providers of many popular web services (i.e. web mashups) that build profiles containing personal attributes of individuals that are traded with advertisers. The data handling practices must comply with data protection regulations such as the upcoming EU General Data Protection Regulation.

The first part in the thesis addresses three objectives of the EU General Data Protection Regulation: (a) enforcing data protection by *integrating privacy* throughout the complete service *design process*, (b) *informing individuals* about the data handling practices of the service they use, and (c) *educating end-users* about the privacy risks of revealing personal information when using services. With regard to these objectives, the first part focuses on an automated privacy analysis approach providing qualitative feedback about the distribution of an individual's personal information across all involved parties. This is modeled by means of user profiles – built by companies – which contain an individual's personal information. The feedback is meaningful for both *end-users* and *system designers* as it provides information on the distribution of an individual's personal information and his anonymity level towards service providers. The approach can inform end-users about which personal information can be collected by which party. For example, the approach could be used in a browser plug-in that informs an end-user if conflicts are found between his personal privacy preference configuration and the disclosure of his personal information towards a particular type of company, such as advertisers that are not permitted to collect his name or address. In the context of service design, the approach can be used in an automated support tool assisting designers with their decisions. The tool could notify them in case design decisions conflict with predefined privacy requirements, or could be used for comparing privacy of different design alternatives.

The privacy analysis approach only considers qualitative aspects of services at application level and abstracts away protocol specific details of lower communication layers. For example, the IP or MAC address specific to the TCP/IP protocol stack are not modeled by default. Nevertheless, they can be manually added to the model. This enables to model state-of-the-art Internet services as well as privacy-preserving applications using anonymous network technologies such as onion routers, e.g. Tor. Furthermore, the privacy analysis approach takes the user's trust perceptions into account as those may influence his behavior when using services.

The contributions in the first part of the dissertation are:

- *First contribution.* The key concepts that are necessary for capturing privacy in electronic services are identified. The concepts are part of the blueprint that is defined for modeling composite services.
- *Second contribution.* A logic-based framework is defined that supports automated reasoning about privacy in composite services.
- *Third contribution.* Multiple query categories are defined that provide meaningful feedback (i.e. qualitative feedback providing information on

how the individual's personal information is distributed across service providers and the individual's anonymity level towards service providers) for both end-users and service designers.

- *Fourth contribution.* The privacy analysis approach is realized using a knowledge-base system, i.e. the IDP system.

The second part of the thesis validates the privacy analysis approach presented in the first thesis part. The contributions are:

- *Fifth contribution.* The approach is validated by scenarios from different domains, namely: a travel booking service, a web shop, loyalty program systems, and a public transport ticketing system. The wide range of application domains demonstrates the flexibility of the approach.

The third part of the thesis focuses on *service provider accountability*, which is a privacy enforcement principle part of the EU's General Data Protection Regulation that is strongly linked with privacy by design [47]. The systems considered comprise multiple components which all have their own privacy policies. Because systems may have many of them, individuals are left oblivious about the actual data handling practices. The approach in the third part aims to provide *individuals* and *trusted auditors* who conduct privacy assessments on behalf of individuals with a system-wide panoramic view on guarantees of declared data handling practices.

The contributions in the third part of the thesis are:

- *Sixth contribution.* An inference model is realized providing individuals with global accountability guarantees of multi-component systems. It maps different user expectations to log evidence of declared data handling practices.

- *Seventh contribution.* The approach is validated on a rail-way camera surveillance infrastructure.

1.2 Outline

The remainder of the present thesis is structured as follows:

Chapter 2: Background introduces privacy in electronic services. The chapter gives an overview of technologies that are commonly used in electronic services, and discusses the integration of privacy during the design of services. Different analysis approaches related to privacy are discussed as well as reasoning systems that can be used to automate privacy analysis. The chapter also motivates the selection of the reasoning system.

PART I: Inspecting Privacy in Electronic Services

Chapter 3: Modeling Concepts identifies the main concepts that must be captured in a composite service to enable reasoning on its privacy properties. The chapter focuses on the blueprint that represents electronic services from a privacy perspective. The blueprint is applied to a service for booking travels.

Chapter 4: Modeling Approach focuses on a logic-based framework that enables the automated reasoning about privacy properties of composite electronic services. It maps the concepts identified in the previous chapter to the framework components. The framework is realized using the IDP knowledge-base system and is applied to a web shop scenario.

Chapter 5: Queries and Feedback shows the potential of the logic-based framework for extracting relevant privacy feedback. The chapter defines different categories of queries that provide feedback useful (i.e. qualitative feedback providing information on how the individual's personal information is distributed across service providers and the individual's anonymity level towards service providers) for both end-users and designers. The queries are applied to analyze the privacy in two alternative loyalty program schemes.

PART II: Extensions and Validation

Chapter 6: Modeling Ticketing Systems in Public Transport validates the modeling approach of the previous chapters on a novel privacy-preserving public transport ticketing system. The chapter identifies additional concepts that are necessary for modeling the considered ticketing systems.

PART III: Inferring Service Provider Accountability

Chapter 7: Service Provider Accountability realizes an inference model providing individuals with global accountability guarantees of multi-component systems. The feedback provided by this approach is useful for both end-users and auditors who conduct privacy assessments on behalf of the end-user. The approach is applied to a camera surveillance scenario.

Chapter 8: General Conclusions reflect on the logic based approach for inspecting privacy in complex electronic services and give directions for future research.

Chapter 2

Background

The present chapter's objective is to situate the subject of this dissertation in the domain of privacy. The research of this thesis is explicitly positioned to the related work. The chapter starts by defining basic terminology that is used throughout the remainder of this text. Section 2.2 gives an overview of different privacy related technologies used in electronic services. Section 2.3 discusses the difficulties of the integration of privacy in the design process of services. Next, different privacy analysis methodologies are categorized and discussed (see Section 2.4). Section 2.5 gives an overview of different reasoning tools and motivates the selection of the logic programming system used for the privacy analysis approach presented in the thesis. The chapter ends with conclusions (see Section 2.6).

2.1 Preliminaries

This section defines basic terminology that is used throughout the remainder of this dissertation.

- *Authentication* [126] is a process whereby a prover and verifier are involved. Two types of authentication can be distinguished. *Entity authentication* is a protocol whereby the prover assures the verifier of his identity and that he actually participated (i.e. liveliness property) in the protocol. On the contrary, *data authentication* includes proving the origin of the information (i.e. *data origin authentication*) and proving that the information is unmodified (i.e. *data integrity*).

- *Personal information* is a set of attributes related to an individual. The individual's surname, address, eye color are examples of personal information.

- *Unique identifier* is a set of attributes – part of the personal information – that can be uniquely linked to an individual, but that not necessarily identifies him. For example, an individual's user name of his chat account can uniquely be linked to him, but is not sufficient to identify him.

- *Personal identitifable information* (PII) is a unique identifier that can be uniquely linked to an individual and that can be used to identify him. For example, the individual's address and name uniquely identify him.

- *Personal data* is defined in the EU directive 95/46/EC as any information relating to an identified or identifiable natural person (i.e. data subject). It can be considered as a synonym for PII, which is used in the US privacy law.

- The *data subject* is the individual to whom personal information is linked to.

- The *data controller* is the party who determines the purpose and means of the processing of personal information.

- The *data processor* is the party that processes personal information on behalf of the data controller.

- A *service provider* is a party that offers a service towards individuals. The service provider can also be data controller or processor.

- *Privacy* has no unique definition. Gürses [92] describes privacy from the perspective of three different *privacy research paradigms.* In the first paradigm *privacy as confidentiality*, privacy is considered as avoiding that an individual's personal information leaks to the public. The next paradigm *privacy as control* refers to privacy as a data subject's interest to control his personal information revealed to a service. Finally, the *privacy by practice* paradigm defines privacy as *the freedom from unreasonable constraints on the construction of one's own identity* [14]. The solutions related to this paradigm make the way how data is collected and processed transparent. An example are privacy-awareness tools that provide feedback to individuals to let them understand the privacy consequences when dealing with services. The privacy analysis approach presented in this thesis focuses on the *privacy as control* and *privacy by practice* paradigms.

- *Data protection* applies to the processing of an individual's personal information. Data protection provides an individual rights by imposing obligations to data controllers for processing an individual's personal information. De Hert et al. [62] show that privacy and data protection are different. They see *privacy* as a tool for opacity based on human rights that protects individuals against interference by state and by private actors, while they see *data protection* as a tool for transparency built on pubic law foundations (e.g. the European General Data Protection Regulations) that is useful between private actors.

- A *privacy policy* is a declaration of intent of a service provider about his data handling practices on revealed personal information.

- A *framework* in this thesis refers to a well organized structure of generic components that can be used for modeling systems.

- A *model* is a framework instance that represents a concrete system.

2.2 Technologies

2.2.1 Authentication Technologies

A widely used authentication method is *password based authentication.* Users must enter the username of their account and prove they have knowledge of the associated secret password. However, passwords provide no strong authentication. More particularly, the user must remember many passwords as for each service separate accounts are used. As a consequence, users choose weak passwords that are easy to remember and reuse them for all their service accounts. The downside of those passwords is that they are easy to guess by computers and it requires little effort from hackers to capture the accounts of users.

A more user-friendly authentication scheme is realized by *Federated Identity Management Systems.* They delegate authentication to centralized authentication services provided by identity providers. The scheme reduces the amount of accounts that must be maintained by users and delegates authentication to third-parties that are more specialized. OpenID [147], Shibboleth [45] and OAuth [95] are some well-known examples of federated management systems. The service providers and identity providers between which mutual trust relationships exist form a federation. In the federation, a user who wants to access a service is redirected to the authentication service of the identity provider. The latter manages his electronic identity that consists of his personal attributes. After the user is authenticated, the identity provider forwards the requested attributes to the service provider after which the user is granted access to the service. For example, a music web service can redirect the user to Facebook to which he must authenticate. Subsequently, Facebook can forward the user's name, age, location and a list containing his social network friends to the music service provider. Centralizing authentication implies privacy risks as a single party, i.e. identity provider, can collect information of all user activities and build highly accurate profiles of users. As a consequence, an identity provider must be highly trusted by a user. Furthermore, users have to trust identity providers that they only forward attributes to service providers that are strictly necessary.

Public-key credentials are electronic attribute containers linked to the owner via the public key included in his credential. The credentials are signed, i.e. certified, with the private key of the issuer. During service authentication the service verifies: (a) if the user is the *credential owner* by checking he has knowledge of the private key associated with the credential's public key, and (b) if the credential is *authentic* by checking the credential's signature using the pubic key of the signer. Only those credentials issued by an issuer trusted by the service provider are accepted by a service. Public-key credentials are managed by an underlying *public-key infrastructure* (PKI). A commonly used PKI is the ITU-T standard X.509, in which X.509 certificates are issued by a *certificate authority* (CA). It is a hierarchical structured system based on the property that trust is transitive. In particular, a CA at a higher level trusts all CAs below in the hierarchy. A service provider only accepts those certificates issued by a CA that is a descendant of a trustworthy root CA and if certificates are not part of a *certificate revocation list* (CRL). Because CRLs can grow rapidly, a more feasible way of verifying the status of certificates is using the *online certificate status protocol* (OCSP). X.509 certificates are used in the HTTPS protocol to secure communication between an end-user and a web service. The protocol realizes two objectives. Firstly, web services authenticate to end-users by showing their certificates and prove they have knowledge of the private key associated with the certificate. The user is only assured that he is communicating with the correct party if he can trust the root CA of the certificate presented, and if the certificate chain is not revoked. Secondly, the credential is used to setup a secure communication channel between the end-user and the web service so that exchanged messages cannot be intercepted by eavesdroppers. For example, a Belgian citizen can authenticate to a Belgian governmental web service using his X.509 certificate based electronic identity card containing his personal information. The governmental service authenticates to the citizen by presenting its X.509 credential. A secured session is established between both parties in case their credentials are successfully validated. Using public-key credentials to authenticate to services can lead to privacy implications. All attributes, part of the credential, are revealed to the service to which the user authenticates. Moreover, user service transactions are linkable via the credential's public key, which is a unique identifier.

Authentication technologies are essential for analyzing privacy in electronic services as they determine which personal information is revealed towards services. The privacy analysis approach in the present thesis considers – besides the personal information that must be revealed for the functionality of services – also the personal information revealed due to the used authentication technology, such as the serial number of an X.509 certificate.

2.2.2 Privacy Enhancing Technologies

Privacy policies of service providers merely contain declarations on data handling practices. They are no guarantee that service providers act according to what is declared. More personal information may be collected than necessary due to the technologies used. For example, an individual who must prove his age using an X.509 certificate based electronic identity card will also reveal his name and address. Another example are service providers who own two databases, splitting personal identifiable information from collected service transaction data. However, for the functionality of the services both databases must link the data to a unique identifier (i.e. pseudonym) that refers to an individual. Only, strict access policies to both databases can prevent service providers from linking transaction data to an individual's identity. The aforementioned examples illustrate that a privacy-by-policy approach is only effective if service providers are well disciplined.

The above-mentioned privacy risks can be mitigated by minimizing the amount of disclosed personal information of individuals to what strictly is necessary for the functionality of services. Privacy-enhancing technologies (PETs) [176] are cryptographic schemes enabling minimizing data disclosure while still fulfilling the functional requirements of services in order to prevent unnecessary processing of personal information of individuals. Different types of PETs exist:

- *Anonymity networks* are networks technologies that hide the linkable information typical in commonly deployed protocols such as TCP/IP network protocols, which exchange IP addresses that often can be linked to individuals. *Onion routers* [90], such as TOR [74], are a well known anonymity network technology. Messages sent in onion networks are recursively enclosed in multiple encrypted communication layers and are sent to a recipient via a chain of a priori random chosen intermediate network nodes. Other examples of anonymity network technologies are: *mix networks* [48, 29, 58], *peer-to-peer mix networks* [124, 153, 52], and *Crowds* [148].

- *A commitment* [36] is a cryptographic protocol between a prover and verifier. The prover selects a value and commits to it without revealing the value to the verifier. Commitments comply with the hiding property, which implies that the verifier can only learn the committed value after the prover opens the commitment (i.e. revealing value). The binding property assures the verifier that the revealed value corresponds to the originally committed value.

- *A Blinded signature* [49] is a cryptographic protocol by which a sender sends a blinded message m to a signer. A drawback is that the signer cannot be assured whether m exactly contains the desired information as he cannot learn m. To prevent abuse, Abe et al. [12] introduce *partially blinded signatures*, a variation on blinded signatures. In their scheme the signer includes an unencrypted common validity term info that preserves the sender's privacy, and signs message $(r, m, info)$.

- *A zero-knowledge proof* is a cryptographic protocol [82] by which a prover convinces a verifier that he has knowledge of a secret x, without revealing it or any other additional information to the verifier. The proof comprises a phase in which a proof is built and a verification phase.

- *Anonymous credentials* are *Attribute-Based Credentials* (ABC) enable individuals to authenticate in a privacy-preserving way, while still providing strong authentication towards service providers. Examples of anonymous credential technologies are Idemix [44] and U-Prove [139] credential technologies. Both technologies support individuals to selectively disclose attributes or proving properties without disclosing the actual attribute values on which the properties are based. For example, an individual can prove, using zero-knowledge proofs, he is between twelve and eighteen years old without revealing his birth date. The issuance of Idemix and U-Prove credentials cannot be linked to the spending unless unique attributes are revealed. Furthermore, Idemix credentials enable unlinkable multiple spending. On the contrary, multiple spending a single U-Prove credential is always linkable. These credential technologies can be used in advanced attribute based identity management schemes [16].

ABC4Trust [5, 151] is a EU-funded research project that addresses the federation and interchangeability of privacy-preserving *Attribute-Based Credentials*. An objective of the project is to define a technology agnostic architecture for attribute-based credential systems making abstraction of the differences between different privacy-preserving attribute based credential systems, such as Idemix and U-Prove. The architecture includes: (a) an *issuer* which issues the credentials, (b) a *user*, (c) a *verifier* who checks if a user satisfies the access conditions for the service, (d) a *revocation authority* who can revoke issued credentials so that they are disabled, and (e) a *trusted inspector* (i.e. trusted authority) who can de-anonymize users in the case misbehavior is detected. The second objective is to provide an open reference of particular ABC systems, such as Microsoft's open source U-prove SDK that is made inter-operable with the ABC4Trust architecture.

Similarly, IRMA [104] is a project focusing on privacy-preserving attribute-based credentials inter-operable with the ABC4Trust architecture. The project's goal is to develop an experimental framework for attribute-based credentials using smart cards [181]. The framework is made open source and is freely available for using it in other applications.

The services under study in this thesis can be mapped to the ABC4Trust architecture. Issuers trusted by the service provider, i.e. verifier, issue credentials that a user can use to authenticate to a service. The credentials can be revoked by a revocation authority or by the service provider himself. The modeling approach supports the modeling of electronic services using privacy-enhancing technologies such as the ones considered in ABC4Trust and IRMA.

2.2.3 Privacy Policy Specifications

Privacy policies are often very extensive and for individuals difficult to interpret. As a result, individuals give their explicit consent to service providers for processing their personal information without being properly informed about the data handling practices. The upcoming EU General Data Protection Regulation addresses this issue by enforcing transparent data handling practices declared in easily interpretable and unambiguous privacy policies readily available to individuals. Machine interpretable languages can enable the automated interpretation of extensive privacy policies. For example, Jafari et al. [105, 106] present a framework for expressing and enforcing purpose-based policies. They use a modal logic based language for formally expressing purpose restrictions. The formal specifications enable machines to enforce service providers only to collect personal information for the specified purposes.

P3P [182] is a standard format for explicitly expressing high-level privacy practices of websites that are machine interpretable. This allows browsers – via an installed plugin [193] – to inform individuals about the practices on the website and automatically take decisions based on pre-configured privacy preferences. Like P3P, EPAL [23] is a formal privacy policy specification language. It can specify fine-grained enterprise privacy policies, while making abstraction of the technological details such as the user authentication. Privacy policies specified with EPAL and P3P can automatically be enforced. EPAL enables to specify the disclosure of personal information, but unlike P3P it is not possible to specify its recipients. Overall, both P3P and EPAL lack expressiveness for high-level company obligations [161].

In contrast to P3P and EPAL, XACML [120] is a declarative XML based language focusing on the specification of access control specifications of distributed systems. It is a machine interpretable language enabling the enforcement of access control specifications. Ardagna et al. [22] propose a framework for a privacy-preserving attribute based access control system based on the industry standards XACML and SAML. Access control specifications of service providers are expressed in an XACML based policy languages and selected attributes are asserted and transported via SAML to the service provider. The individual can select the disclosed attributes from a wallet of credentials he owns. E.g., he can disclose his name from an X.509 based identity card and his university from his Idemix based student card. Compared to EPAL, XACML is more comprehensive than EPAL since it is a functional superset of EPAL [19]. Another example of an access control specification language is CARL [43], which can be used to enforce privacy-preserving access control.

In the present thesis, the privacy analysis is performed on models representing electronic services. The models are extracted from the privacy policy specifications of the services under study. They describe the personal information collected and stored by services, and the personal information that services forward to collaborating entities. Machine interpretable policy languages can improve the overall quality of the models and privacy analysis as they enable to automatically generate models from available privacy policies of services.

2.3 Privacy by design

Currently, personal information is a main asset in many business models of companies. The more data companies collect, the more accurately they can profile individuals and the more money they can earn by trading it with third-parties. Data controllers based in a EU member state and data processors acting on behalf of them will be bound by the upcoming EU General Data Protection Regulation. Data controllers and processors are enforced to publish privacy policies compliant with regulations and act accordingly. A key data protection principle in the EU regulation is *privacy by design*. The principle must ensure that data controllers and processors provide the strongest privacy protection possible by imposing them to build privacy safeguards into their services or business operations from the earliest design stage. It should provide a solution to find a balance between the concerns of individuals and data controllers and processors. However, privacy by design is barely specified by the EU's privacy regulation, and there are challenges that must be addressed before it can be an effective privacy protection measure [160]. Particularly, building privacy into systems is not straightforward and cannot be realized by solely using privacy-enhancing technologies (see Section 2.2.2), but by a combination of those technologies with a systematic privacy engineering methodology and governance controls.

From the academic community, a universal privacy by design framework [46] based on the Fair Information Practice Principles (FIPP) [175] strives to establish the strongest privacy protection possible. According to the principles of which the framework is comprised, systems must have clearly declared privacy policies available to individuals, and appropriate mechanisms must be implemented to demonstrate compliance of data handling practices with declared ones. An individual's personal information can only be processed after he gives his explicit consent. The personal information must be securely collected and maintained throughout its entire life cycle to ensure confidentiality, and it must be made available to the data subject so that he can update his collected personal information. The FIPP also aim to have a systematic privacy methodology in the different stages of the design process. The design process must respect privacy requirements by including a conflict resolution strategy for achieving an optimized solution that meets all requirement types. For example, PETs can realize systems satisfying both privacy and functional requirements.

The aforementioned framework consists of vague high-level privacy criteria applying to technologies used, services, architectures of systems and business operations. The framework neither details concrete technologies and architectural patterns, nor systematic privacy engineering methodologies. This is acknowledged in the analysis of Gürses et al. [93]. Moreover, they motivate the need for the explicit mentioning of *data minimization* in privacy regulations. They point out that the loosely defined restrictions on data collection found in several privacy frameworks lead to different interpretations by data controllers and processors, and discuss its consequences. For example, data controllers may define a very small scope of what they consider as personal information to enlarge the scope of information they are allowed to process. The authors argue the need for generalizable privacy by design engineering methodologies based on data minimization. However, limited expertise about privacy by design methodologies exists. They propose a methodology comprising different activities. Each design of a system starts with an analysis of the *functional requirements*, in which functionality must be precisely defined to constrain the tendency of engineers to collect more personal information than required. The remaining activities – which can be performed in any order – are:

- *Data minimization.* Designers must check if the set of collected data can further be reduced and must evaluate if appropriate technologies, such as privacy-enhancing technologies, can help to realize this.
- *Modeling attackers, threats and risks.* Possible attacker types (e.g. a curious third-party), types of attacks ()e.g. linking sensitive health information to individuals), and the impact of a successful attack.
- *Multilateral Security Requirements Analysis.* Designers must analyze and resolve possible conflicts between security and privacy measures. The conflict resolution strategy may imply the re-iteration of the functional and multilateral security requirements analysis.
- *Implementation and testing.* The systems must be implemented and evaluated whether all requirements are satisfied.

The aforementioned methodology clearly shows that privacy by design engineering is not trivial as it requires designers have knowledge of different domains. For example, designers must have knowledge of complex privacy-enhancing technologies, privacy engineering and privacy law. The above methodology is not complete, it only presents the activities that are essential in any privacy by design approach. However, the method is too high-level for designers since it does not inform them about the means for conducting the actual design activities. Methodologies hiding lower-level details can assist designers who have no experience with privacy related aspects throughout the privacy by design process. For example, PriMan [145] is a technology agnostic development framework that facilitates the integration of privacy-preserving technologies into applications. Technology-specific configurations are specified in configuration policies separated from the application's source code. In this way, domain experts can specify technology-specific details independently from developers who only must focus on the functionality. Another example is a pattern language for privacy-enhancing technologies [94] that can assist non-experienced system designers with selecting the privacy-enhancing technology that satisfies privacy and functional requirements. Hoepman [98] argues that design patterns are too specific and not useful when the first concepts of a system are designed. He derives eight privacy design strategies and relates them to existing design patterns – covering aspects fundamental to privacy protection – supporting privacy by design throughout the full software design life cycle even at the beginning when concepts must be developed. The strategies are: (a) *minimize* the amount of collected data to what is strictly necessary, (b) *hide* data and the relationships with individuals from plain view, (c) *separate* the processed data into different unlinkable compartments, (d) *aggregate* data to the highest aggregation level possible such that it is still useful when processing it, (e) *inform* individuals when processing their data, (f) give individuals *control* over the personal information that can be processed, (g) data handling practices respecting an individual's privacy should be *enforced* by a privacy policy compliant with regulations, (f) *demonstrate* compliance of data handling practices with privacy policies and regulation by implementing mechanisms such as using logs and performing audits.

To ensure that data controllers and processors correctly implement privacy safeguards into their services or business operations, the European Union Agency for Network in Information Security [59] (ENISA) recommends policy makers to support privacy engineering research and that tools should be provided which assist system designers. Furthermore, policy makers should promote privacy and data protection in their norms and standardization bodies should include privacy in their standardization process and provide interoperability standards of privacy features. For example, ISO/IEC 29100 [7] is an international high-level privacy standard framework that supports data controllers and processors with defining their privacy requirements. The framework refers to known privacy protection principles, such as *consent and choice*, *data collection limitations*, and *data minimization*, and also defines a common privacy terminology and the actors and roles in processing personal information. The standard is complementary to the data protection regulations, such as the EU General Data Protection Regulation. ENDORSE [6] is a EU funded research project creating a legal technical framework to guarantee privacy and protection during data processing. The project defines a privacy rule specification language for data access and management in digital systems. A tool set is created guaranteeing data controllers/processors and individuals that the processing of personal information complies with law. ENDORSE also defines a certification methodology for data controllers and processors to increase the trustworthiness of services towards individuals. Typically, data controllers or processors can obtain a privacy seal, i.e. a certificate, asserting they act according their privacy policies and the imposed data protection regulation. Seals are issued by an external company after having conducted a successful privacy assessment. According to this scheme, the end-user is guaranteed by the certification company that his privacy is well protected by the data controller and processor. However, the privacy seal scheme is only useful if the certification company is trusted by the end-user. In reality, a wide variety of privacy seals exist and are often commercially driven. According to the EU Privacy Seals Project [150] this is disadvantageous for the legitimacy of such schemes. For instance, TRUSTe [3] and Comodo Secure [1] are clearly commercially driven. Also, certification companies are not always transparent about the criteria they evaluate for issuing privacy seals. Moreover, there is no uniformity between the criteria of different brands of privacy seals. Hence, a European-wide transparent certification scheme is essential to be effective.

Some companies, such as HP, already included privacy in their data handling processes. Their semi-automated decision support tool [141] assists their employees, who process an individual's personal information, step-by-step through a privacy related template questionnaire based on an underlying flowchart. The tool focuses on business operations rather than on the design of applications. Microsoft's Security Development Lifecycle [127] (SDL) considers privacy as a security objective, i.e. data confidentiality throughout the complete development process of applications. SDL does not only focus on the technical development aspects, it also addresses project management aspects. A maturity model assigns companies a maturity level – ranging from basic to expert – according to their expertise. The maturity model lets companies gradually adopt the complete SDL approach. SDL also defines roles to members of the development team. For example, the privacy advisor and privacy champion are responsible for privacy related aspects of the application under development. To guarantee privacy quality in development systems, the methodology includes privacy quality thresholds, which are upper-boundaries for the number of privacy related bugs. Finally, a privacy risk assessment assigns a risk level to detected privacy vulnerabilities to prioritize them.

The privacy analysis modeling approach proposed in the present thesis can provide computer-aided assistance to service designers who lack expertise of privacy-enhancing technologies and privacy regulations. The modeling approach models high-level privacy properties of different authentication technologies (e.g. selective attribute disclosure), and collaborations between different entities that are involved in the modeled service. Moreover, generic models representing privacy policies can be made available to service designers. The modeling approach enables designers to automatically check if design decisions conflict with privacy requirements without requiring expert knowledge from the designer.

2.4 Methodologies

Multiple approaches have been proposed to analyze privacy related properties. The present section gives an overview of existing methodologies and positions the approach of this thesis to them. First a global overview of all the approaches is given and their purposes are briefly discussed. An in-depth discussion follows in the remainder of this section.

Figure 2.1 gives an overview of the global methodologies for analyzing privacy. They are categorized into *privacy requirement engineering*, *quantitative* and *qualitative methodologies.* Requirements engineering approaches are well-structured methods focusing on extracting privacy requirements at system level. They can be categorized into security oriented approaches in which privacy is one of the security objectives, and in purely privacy oriented approaches. Quantitative privacy analysis methods are based on metrics enabling to quantify privacy. They result in values indicating the level of privacy in the analyzed system. For example, they measure the degree of anonymity of individuals sending messages via an anonymity network. The resulting measurements can be used to compare privacy in alternative systems and to order systems according to their level of privacy they provide. Qualitative approaches provide feedback about qualitative privacy properties of systems. E.g., they provide feedback about the anonymity level of a user in terms of anonymity, pseudonymity, and identifiability. The objectives of each qualitative approach vary. For example, approaches may analyze privacy at system, application or protocol level; or they focus on detecting conflicts in privacy policies. Formal methods are popular methods used in qualitative approaches. For example, the approach presented in this thesis extracts qualitative privacy properties from logic based models representing the system under inspection. The remainder of this section discusses each of the privacy analysis methodologies in more detail.

Privacy requirements engineering. The following approaches assist system or software engineers with defining and maintaining privacy requirements of systems. The approaches are well-structured methodologies to extract privacy related requirements from systems. Different methods/frameworks exist for deriving privacy requirements.

STRIDE [100] is a Microsoft security threat modeling approach, in which privacy in terms of data confidentiality is one of the security objectives. A *data flow diagram* of the considered system is built, representing how information enters the system and how it propagates. Security threats are manually derived from the data flow diagram. The identified threats are then assigned a severity and priority level using DREAD, Microsoft risk rating model. Appropriate security and privacy measures are implemented to mitigate the threats.

The CORAS method [121] is another popular approach. It is an intuitive graphical security oriented threat and risk modeling approach based on UML. Similar to STRIDE, it focuses on information confidentiality as one of the

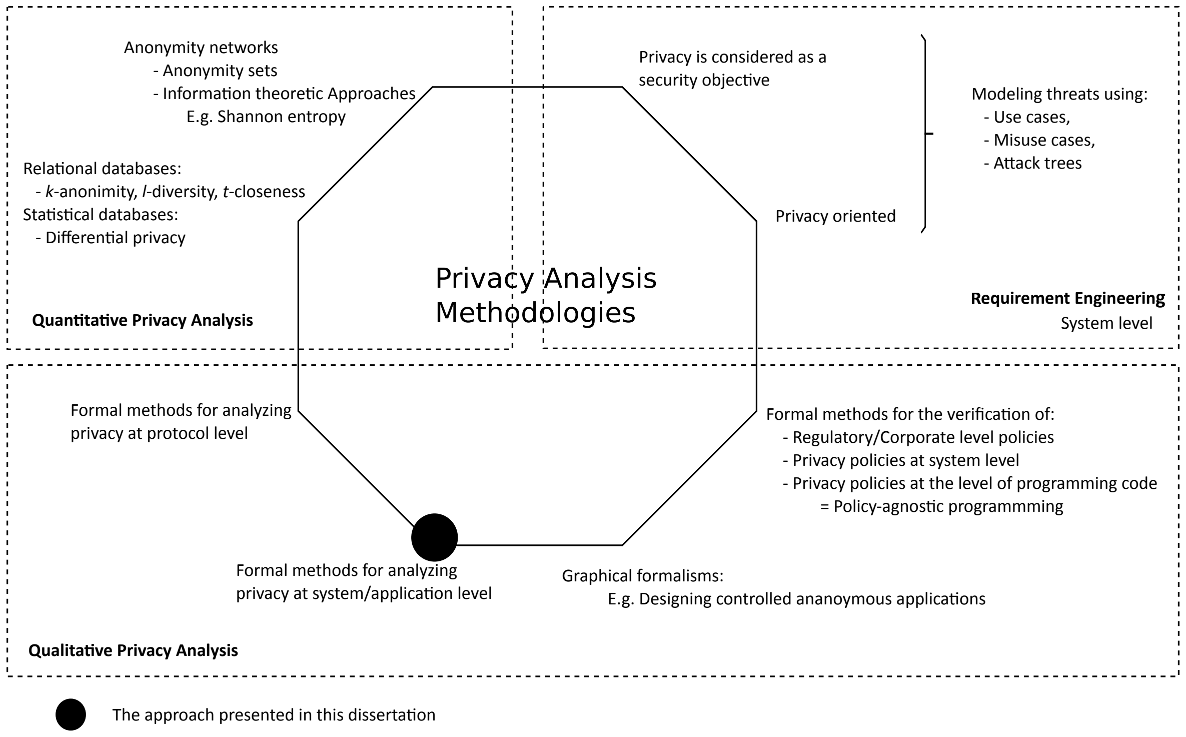
Anonymity networks
- Anonymity sets
- Information theoretic Approaches
E.g. Shannon entropy
Relational databases:
- *k*-anonimity, *l*-diversity, *t*-closeness
Statistical databases:
- Differential privacy
Quantitative Privacy Analysis
Privacy is considered as a security objective
Privacy oriented
Modeling threats using:
- Use cases,
- Misuse cases,
- Attack trees
Requirement Engineering
System level
Privacy Analysis Methodologies
Formal methods for analyzing privacy at protocol level
Formal methods for the verification of:
- Regulatory/Corporate level policies
- Privacy policies at system level
- Privacy policies at the level of programming code
= Policy-agnostic programmming
Formal methods for analyzing privacy at system/application level
Graphical formalisms:
E.g. Designing controlled ananoymous applications
Qualitative Privacy Analysis
● The approach presented in this dissertation

Figure 2.1: Overview of the privacy analysis methodologies.

security objectives. A tool supports documenting the results in UML based CORAS diagrams. Threat scenarios are identified and documented in threat diagrams used during the threat risk assessment.

Sindre et al. [159] introduce *misuse cases* – i.e. use cases describing misuse – to derive security requirements of systems. Their requirements process is a cyclic process. First, they identify the critical assets in a system. Security goals are then defined for each asset. Threats associated with the goals are identified and described using misuse cases. A risk level is assigned to the threats, and security requirements are defined taking into account risk and cost for each threat. Deng et al. [71] present a privacy threat analysis framework for the elicitation and fulfillment of privacy requirements. They also use misuse cases but with a larger focus on privacy requirements, while in the former, privacy is less prominent and considered as one of the security objectives. Similar to STRIDE, they use a *data flow oriented model* to identify the privacy threats in the system under consideration. Misuse cases are then identified for the threats, followed by a risk assessment to prioritize threats. Based on the assigned priority, privacy requirements are elicited. Finally, the analyst can select the appropriate PET via a taxonomy mapping privacy objectives to PETs. Wuyts et al. [189, 190] conducted an empirical study of the approach and conclude that the threat analysis framework can be useful for requirements engineering and architectural design.

Attack trees [154] are another strategy for modeling privacy and security threats on system. The root node of the tree is the goal of the attacks. The latter are represented by the leaf nodes. Different values can be assigned to the nodes of attack trees and facilitate estimating the likelihood an attack occurs or the cost to perform a particular attack or prevent it. Misuse cases, instead, are better suited to represent threats at application level and can easily be combined with use cases showing the interactions between a user and the system. Attack trees only show the attack paths on systems and are user-agnostic. An experimental comparison [137] shows that attack trees are considerably more effective than misuse cases for identifying possible threats in systems. The results are confirmed by Karpati et al. [110] who compared attack trees and misuse cases in an industrial setting, but they additionally conclude that both approaches are complementary because both approaches detect different types of threats. Hence, both can be combined. The idea of combining attack trees and misuse cases in a single approach is exploited by [165, 168].

The requirements analysis methods are mainly manually conducted processes. Beckers et al. [27] propose computer aided privacy threat elicitation which supports privacy analysts during the requirements analysis of software systems. The approach uses *problem frames* [103] to capture stakeholders, technology, and the personal information in the designed system. Problem frames are a requirements engineering technique defining problem classes. They help to analyze system requirements by decomposing systems in easier to handle subproblems. Faßbender et al. [80] provide tool-support – based on problem frames as well – to transform functional requirements into legal identification pattern instances [28] that can be used to derive software requirements and implementation specifications using a structured method [79].

The privacy analysis approach in the present thesis does not provide a structured way to elicit privacy requirements in systems. It merely provides an automated reasoning tool capable of verifying if a modeled electronic system complies with pre-identified privacy requirements. Hence, the privacy analysis approach in the thesis is complementary to privacy engineering approaches. In particular, privacy requirements elicited by privacy requirements engineering approaches are the input for the privacy requirement compliance verification.

Quantifying privacy. Quantitative privacy analysis methods consider privacy metrics measuring an individual's *degree of anonymity* [148] that provides a continuum between complete anonymity and identifiability towards an attacker. If the measurement of the modeled system exceeds a threshold, individuals can be considered sufficiently identifiable.

Díaz et al. [73] propose an information theoretic approach considering a set of possible (honest) senders of a message, i.e. the anonymity set [143]. They determine a sender's degree of anonymity towards an attacker. In their approach, the attacker is an observer who assigns a probability of being the sender of a message to each individual in the anonymity set. The degree of anonymity is a metric based on the Shannon entropy [157], an information theoretic concept representing the amount of information bits contained in a message (i.e. a message can be represented by n bits). The above-mentioned approach addresses the unrealistic assumption made in the approaches of Reiter et al. [148] and Berthold et al. [30]. The former measures the degree of anonymity of *Crowd* members submitting requests to a web server. Attackers assign a probability p_f to a *Crowd* member being the originator of a message. However, the degree of anonymity $1 - p_f$ does not distinguish one member from other members. The latter approach, defines the degree of anonymity as $d = log_2(N)$. The metric corresponds to the maximum Shannon entropy that only depends on the size N of the anonymity set. It assumes that all users have equal probabilities of being a message's sender. The approach is inaccurate because most realistic systems have users sending messages more frequently than others. The inequalities between probabilities is addressed by Díaz et al. defining a normalized comparison between Shannon entropy and the maximum entropy. Serjantov et al. [156] show the problems related to anonymity sets. They argue that the size of the anonymity set is not an adequate indicator because of the unequal probability distribution of sending a message over the users in the anonymity set. Their approach is similar to [73], but they consider both senders and recipients of a message. Instead of considering an anonymity set, they use a probability distribution over the set of users being the sender or recipient of a message. Similar to [73], their metric is based on the Shannon entropy expressing how many additional information bits an attacker needs to identify a user as being a sender or a recipient. More recently, Almasizadeh et al. [15] define *mean privacy*, a security metric quantifying the amount of private information leaking to an attacker expressed in units of information bits, i.e. Shannon entropy. They claim their metric is a good indicator for the security of a computer system. The lower the value of the indicator the safer a computer system is. They assign a discrete mass probability distribution to an attack tree representing all feasible attacks on a computer system. Although it is a security metric, it can be applied to determine privacy of systems based on requirement engineering approaches use attack trees. The approach assumes that a probability can be assigned to each node of the attack tree. However, it is unclear how in practice probabilities must be assigned. Tóth et al. [169] show the shortcomings of metrics based on Shannon entropy. They illustrate this by means of two different probability distributions – considering a Shannon entropy based metric – having the same degree of anonymity. However, both probability distributions provide different anonymity. They propose a *local anonymity metric* based on min-entropy and max-entropy.

The following metrics focus on privacy in databases. Sets of attributes exhibiting k-anonymity [162] map to at least k entities. The autors of [122] define l-diversity. A dataset has l-diversity if all groups of attribute sets have at least l well-represented values for the sensitive attribute. The novel privacy criterion is resistant to two attacks causing k-anonymized datasets leaking sensitive information about individuals. In case of the homogeneity attack, datasets can leak an individual's sensitive attributes when they lack diversity. An attacker can have *background knowledge* about a particular individual, which helps him to reduce the set of possible data records. Li et al. [119] claim that l-diversity is difficult and unnecessary to achieve. Moreover, it is insufficient to prevent attribute disclosure. They define t-closeness, a stronger privacy criterion. A group of attribute sets have t-closeness if the distance between the distribution of a sensitive attribute and the distribution of the attribute in the whole dataset is no more than a threshold t. The dataset has t-closeness if all groups of attribute sets have t-closeness. Differential privacy [76] is used in the context of statistical databases. Statistical operations on data sets that differ only by one element are considered ϵ-differential private if their outcome shows no significant difference . The objective is to have accurate operations, while leaking minimal personal information of individuals.

The aforementioned metrics are not generic as they are designed for specific PETs. A more generic information theoretic framework for privacy-preserving systems [146] defines *attacker's estimation error*, a privacy metric representing the estimation error of an attacker who aims to disclose privacy information that should remain concealed in the system under analysis. The metric relates the metrics used in the above-mentioned quantitative approaches, i.e. they can be considered as particular instances of the *attacker's estimation error*.

Danezis [57] acknowledges that qualitative approaches, in which metrics are a summary of the inference of an attacker, have shortcomings [163]. He states that analysis must take into account that in many cases it is hard to compute the likelihood function representing the adversary's observation due to noise in the system. Hence, the analysis must assume that adversaries can trivially retrieve all unprotected data, such as timing correlation information [131] of Tor networks.

Unlike quantitative approaches, the approach in the present thesis does not summarize privacy in a system by means of a single value. Nevertheless, the approach is complementary to the above presented approaches, and both types of approaches can perfectly be combined. More specifically, the outcome of quantitative approaches can provide statistical information representing the uniqueness of attribute sets. The approach in this thesis can mark these sets as identifiable sets that can sufficiently identify an individual. E.g. the identifiable attribute set sufficiently identifies individuals if it exhibits k-anonymity, whereby k falls below a threshold where individuals are considered (sufficiently) identifiable.

Qualitative privacy analysis. Quantitative privacy analysis approaches result in numeric values expressing the individual's privacy level in a system. The quantitative approaches considered above only provide feedback on one specific privacy aspect, such as an individual's degree of anonymity. An order can be defined between different systems w.r.t. the privacy friendliness based on the resulting numeric values. In contrast to quantitative approaches, qualitative privacy analyses provide feedback that cannot be expressed by a numeric value. The feedback from qualitative approaches is broader than from quantitative approaches. Depending on the goals of the qualitative approaches, different types of feedback can be provided. For example, approaches may provide qualitative feedback on the compliance of a service's data handling practices with regulations. Other approaches, may focus on feedback expressing an individual's privacy in terms of linkabilities to unique identifiers. A brief overview of different qualitative privacy analysis approaches is given below.

The research community identifies and recognizes the complexity of reasoning about privacy policies and correctly implementing them. *Policy-agnostic programming* [24] is a method enabling programmers to separate functional program code from privacy policy specifications. The programming model shifts the responsibility for ensuring compliance with privacy policies from the programmer to the programming system. The principle is used in the Jeeves language [191], a functional constraint language. Costanzo et al. [56] present an approach based on the same principle. They use a separation logic for enforcing declarative information flow control policies based on Hoare logic [97] enabling reasoning on programs specified in imperative languages like C. In case of composite web services, compliance with privacy policies is even more complex. Breaux et al. [38] illustrate this with a real-world Facebook-app that forwards data to a third-party advertiser who provides personalized in-app advertisements. They show the complex privacy conflicts that arise due to the privacy policies of each individual organization. They have mapped the privacy policies in human language to formal specifications in description logic [37]. The formal model, which describes privacy requirements and data flows in the system – allows to automatically detect conflicts between privacy policies claimed by individual organizations. Moreover, privacy law frameworks, such as the EU General Data Protection Regulations, may constrain data practices applied by organizations. Conflicts may arise between corporate level service policies and the ones imposed by legislation. To detect conflicts, formal methods [25, 72] can be used in which legislative rules are expressed using first-order logic supporting temporal modalities, enabling the automated compliance verification between data handling practices and legislative privacy frameworks, such as HIPAA [4]. Tschantz et al. [172] present a formal approach to reason about the acceptability of data collection. More specifically, they inspect if personal information that is collected, stored and/or (eventually) distributed really serves certain goals. E.g. they verify whether data that is collected by a doctor really serves the patient's treatment. A generic formal framework for specifying privacy policies for social networks is presented by Pardo et al. [140]. They aim to provide individuals a means to define their own privacy policies, and guarantee that their privacy preferences are respected.

Formal protocol models, such as the ones based on process algebras as the applied π-calculus, can deliver communication level data that is released by communicating entities. For example, IP addresses, session identifiers and possibly cookies are exchanged in many communication protocols. Formal verification tools as ProVerif [31], supporting the applied π-calculus, automate the formal verification of privacy-preserving protocols. Recent work applies formal methods for the verification of a privacy-preserving e-voting system [55, 54], e-auction system [75] or a protocol for electric vehicle charging [81]. Veningen et al. [178] present a deductive system to analyze privacy at the protocol level. They represent data in a three-layer model that includes the messages exchanged, their content, and their context. The approach allows for a very detailed analysis.

The design of systems is complex as designers have to consider requirements from multiple disciplines. They must not only design systems providing the functionality required, but also must ensure that systems comply with privacy requirements. As a consequence, they must have expert-knowledge of aspects from different disciplines. Computer-aided support tools assisting designers with their design choices are necessary to improve the overall quality of systems. The logic based framework [114] focuses, similar to PETs, on the data minimization principle, and reasons about the impact of architectural design choices on the privacy in advanced decentralized services comprising non-trusted stakeholders. The approach presented by Antignac et al. [21, 20] also focus on the architectural level and motivates the need for reasoning tools relying on formal models for the adoption of PETs by industry. They suggest the use of epistemic logic based models addressing conflicts between privacy requirements and other types of requirements. Naessens et al. [132] present a multi-paradigm methodology for designing controlled anonymous applications. The methodology not only takes the privacy concerns of the user into account, but also the concerns of other stakeholders, such as accountability and the personalization of services. The methodology allows to express anonymity and control requirements at a very high abstraction level. Anonymity properties can be derived from a conceptual model. Their approach also supports the semi-automatic selection of PETs to realize the objectives. Finally, strategies for resolving requirement conflicts are proposed as well. Generally, modeling approaches solely analyze privacy at protocol level or at system level, but not on both levels. Thong et al [164] bridges this by presenting a process algebra based modeling approach capable of verifying the compliance of privacy architecture objectives expressed at high-level and privacy properties of the architecture's underlying protocols and PETs.

The current thesis presents a first-order logic based modeling approach providing qualitative privacy feedback of composite electronic services involving different stakeholders. More specifically, the approach analyzes the privacy at system level by computing profiles containing personal information that can be built by service providers. The approach builds profiles of personal information that is collected via different types of authentication technologies, such as Idemix credentials or X.509 certificates. The approach is capable of providing automated feedback about the privacy consequences of design decisions, the consistence of individual privacy policies of service providers involved in the service under consideration, and the impact on the privacy of individuals who use a particular service. Since models are expressed in a first-order based logic, *the approach is only partially capable of expressing rules of privacy regulation frameworks, such as HIPAA. Expressing such rules requires logics that fully support temporal modalities, e.g. computation tree logic (CTL) where time is branched.*

2.5 Logic Programming Tools

The present section discusses different types of tools automating the reasoning on *logic program* (i.e. logic programming tools). A tool is selected suited for the realization of the privacy analysis approach presented in this thesis. This section starts by enumerating the requirements for the reasoning tool, followed by a global comparison between different types of reasoning tools. Next, the selected reasoning tool is introduces and a motivation for it is given. The section ends with a more detailed discussion of different types of reasoning tools.

A key objective of the modeling approach is to assist non-experts with the modeling of complex electronic services. The selected reasoning tool must meet the following requirements:

- *(P1.)* A *formal* representation language is preferable. Particularly, the language concepts and semantics should be derived from mathematics. First-order logic is an examples of a formal language.
- *(P2.)* It must support a representation language that is fully *declarative* and *intuitive*. Most programmers are familiar with procedural languages like Java and C++. Programs describe how a solution is computed. In contrast to procedural languages, declarative languages specify the (partial) knowledge about a solution, i.e. rules and known facts related to a solution, rather than how the solution is calculated. Declarative systems use particular inference techniques to automatically search for the solution(s) satisfying the modeled knowledge.
- *(P3.)* The modeling language must be *expressive* to specify data flows and relations in multi-component services. For example, a relation is required to express that a service is provided by a particular service provider, and that data is forwarded by one service to another.
- *(P4.)* The modeling language must be capable of expressing the *transitive closure* since it is a property of service invocations. A service invoked by a user causes the automated invocation of a sequence of sub-services. As a consequence, the user indirectly invokes a chain of sub-services.

- *(P5.)* The modeling system must be capable of handling *incomplete knowledge*. For example, a designer must analyze if collaborations between service providers make individuals identifiable. Two configurations can be considered for collaborations. Collaborations may be given and the designer of electronic services must just verify if an individual is identifiable. In the other case, the designer must search for avoidable collaborations that make an individual identifiable. The modeling system must be capable of flexible knowledge configurations.
- *(P6.)* The modeling approach can be applied on different types of services from different domains. Hence, the language must support modular modeling. For example, different modules are used for different inferences.
- *(P7.)* The selected modeling system must compute solutions within reasonable time. It is not required to have the most efficient solver.

Table 2.1 gives a global overview of different types of reasoning tools and shows their properties. It can be concluded from this table that IDP meets the aforementioned requirements. Hence, IDP is selected for the realization of the privacy analysis approach presented in this thesis.

Table 2.1: Overview of reasoning tools

	P1	P2	P3	P4	P5	P6	P7
IDP	✓	✓	✓	✓	✓	✓	±
Prolog			✓	✓		✓	+
Answer set programming			✓	✓	✓	✓	±
Operations research	✓		✓				+
Constraint programming	✓	✓			✓		±
Mathematical modeling	✓	✓	✓			✓	+

IDP. [11, 185] is a state-of-the-art knowledge base system [64]. It is a declarative logic programming system focusing on the intuitive representation of knowledge. It uses the FO(.) language. FO(.) is a formal representation language extending first-order logic (FO) with types, partial functions, aggregates, and inductive definitions [69] (see Appendix A) under the well-founded semantics [177]. Under this semantics, incomplete knowledge is supported. More specific, literals about which no information is available are assigned a value *unknown*. A program in the IDP language comprises three parts: (a) a *vocabulary* containing the symbols to model systems, (b) a *theory* expressed over the vocabulary containing the set of constraint rules and inductive definitions and (c) a three-valued structure (true, false, uncertain) over the vocabulary which is the partial assignment, i.e. interpretation, of the elements and relations of the vocabulary. The IDP system provides different inferences to find models that satisfy the theory. The IDP inference used for the modeling approach of the present thesis is *(optimal) model expansion*. Given a partial assignment (i.e. an input model containing given facts), find a complete (optimal) assignment (i.e. the resulting output model) so that all expressed constraints and definitions hold. In the remainder of the thesis the *input model* and *output model* will refer to the structure and the resulting complete assignment of the inference respectively.

The remainder of this sections gives a more detailed discussion about different types of logic programming tools.

Prolog. Prolog is a partially declarative logic programming system. Only programs containing solely a set of Horn clauses – a disjunction of at most one positive literal – are interpreted purely declarative. The property that the resolvent of two Horn clauses is a Horn clause as well is part of Prolog's inference for efficiently finding a model satisfying the program. However, Horn clauses have limited expressiveness and are inadequate to express all properties of electronic services considered in the thesis. For example, they cannot express the transitivity rule of an order relation required to model the invocation of a service as a consequence of previously invoked services. As a consequence, the non-declarative language part of Prolog is necessary, such as recursive definitions required to express the transitivity the property of an order relation. Recursive definitions comprise a set of rules that possibly refer to themselves. However, the order in which they appear in the definition influences the computation in Prolog. Consequently, the modeler must not only focus on the semantics, but also on the computational aspects. Prolog also provides negations and evaluates them using *negation as failure* (NAF) [51], inferring the negation $\neg p$ from the failure to prove p, but this is semantically different from the concept negation as known in classic logic. Overall, Prolog is less suitable for a modeling approach in which the declarative modeling paradigm is essential.

Answer set programming. Examples of ASP systems are Clingo [86] and DLV [117]. Unlike Prolog, answer set programming (ASP) is a fully declarative logic programming paradigm under the stable-model semantics [87]. Stable-model semantics provide a declarative semantics for negation as failure and addresses the difference between the truth tables of the negations from classic logic and Prolog. However, the aforementioned negation as failure interpretation is insufficient in case of incomplete knowledge. In particular, a negative literal $\neg p$ is assigned *true* if no information about p is available. To support incomplete knowledge, the standard ASP language [41] is enriched with *strong negation* [88], i.e. $\neg p$ is *true* if p is *false*. The ASP language is expressive and like the IDP language it supports both constraints and inductive definitions [70]. However, IDP supports complex nested first-order formulas, which are not supported in many ASP systems. Furthermore, ASP systems are not intuitive as they expect that modelers bear in mind how the ASP solver handles programs, which is less the case when modeling with IDP. ASP systems may support modular logic programs. For example, *Clingo* supports reusable parameterized subprograms . Benchmark results [61] from the ASP competition 2013 [17] show that IDP's efficiency is competitive compared to the best ASP systems. The choice to use the IDP system rather than ASP systems is motivated by the fact that IDP has mathematical language aspects and semantics, such as set-theory, classic logic and inductive definitions. ASP systems on the contrary, have language concepts – such as negation as failure and strong negation – originating from the field of commonsense knowledge representation and non-monotonic reasoning.

Operations research. The field of operations research (OR) includes different domains of problems. For example, improving the fleet utilization of carriers [116] to reduce the fuel costs in times of rising fuel charges is important. Nurse rostering problems [50] are another well known example in which hospitals must produce duty rosters for nurses. Different powerful analytical techniques derived from numerical mathematics can be applied to OR problems, such as linear programming models [60], integer programming models [186] and discrete time Markov chain models [135]. The performance of OR systems, such as CPLEX [9], in which linear arithmetics are used are generally superior to the IDP system, but is worse in case of boolean related problems. Modeling using OR systems is time-consuming and complex. Moreover, they have no support for inductive definitions. Consequently, each inductive definition must be manually constructed. Considering the complexity of using OR systems, they are not preferable.

Constraint programming. An example of a system for constraint programming (CP) is Minizinc [134], which supports a very expressive language and has a powerful solver. CP systems are based on mathematical concepts and overlap with FO(.) used by IDP. However, there are no CP systems that support inductive definitions, making CP systems unsuitable for the modeling approach of the present thesis. An additional disadvantage is that CP languages include concepts which imply that a theory of a problem in a particular context cannot be reused in another context. More specifically, a theory containing an array of known inputs cannot be reused in another context where the input is unknown. Similar techniques as in the IDP solver are used by the CP solvers. Compared to other MiniZinc solvers, IDP performed the best on different constraint satisfaction problems [18, 61].

Mathematical modeling. Principles from mathematics, such as set theory, form the base of Z [187], B [113], Event-B [13], Alloy [101], and TLA+ [112, 111]. The aforementioned languages are used for formally modeling dynamic systems. Some languages, such as B and Z support higher-order structures and quantifications, whereas Alloy only supports first-order structures. Furthermore, the B language enables the automated generation of program code, while Z is useful in case of large system specifications [102]. Related modeling systems are ProB [118], Alloy analyzer [101], and TLA+. They focus on the formal verification of dynamic systems rather than solving problems. The technologies used by mathematical modeling systems varies. For example, ProB is based on constraint logic programming, while Alloy uses SAT solver technology. Although some languages support higher order logics (e.g. ProB), they have no support for inductive definitions which is required for the modeling approach of the present thesis. IDP is conceptually closely related with mathematical modeling systems as it uses a logic based language as well.

2.6 Conclusions

The present chapter identified issues related to including privacy safeguards in the design process. Privacy regulations imply to integrate privacy safeguards into the complete design process from the earliest stage. However, they do not describe how this can be realized. Methodologies that can be used for the analysis and integration of privacy in the design process are discussed. The privacy analysis approach presented in this thesis is compared to them. The chapter also studied different reasoning tools that can be used for automating the privacy analysis, and motivates the selection of the IDP reasoning tool for the realization of the privacy analysis approach presented in this thesis.

Part I

Inspecting Privacy in Electronic Services

Chapter 3

Modeling Concepts

Electronic services evolved from straightforward interactions between a user and a service provider, to complex electronic services in which multiple stakeholders collaborate. Individuals typically interact with these services via a front-end. Underlying services are often activated in order to perform specific tasks. For instance, multiple services possibly administered by different service providers are activated when a user purchases an item in a web shop. The shop handles the purchase request, whereas an external payment provider is involved to complete the financial transaction. A postal delivery operator can deliver the items at the customer's home. Service providers can collect a lot of personal information during a transaction, and consequently release that data to other – possibly untrusted – entities. The latter can use these data to profile individuals, which are often unaware of the data handling practices after service consumption. Although only the personal data really required to handle the initial service request should be released, often much more information is collected. First, usually much more information than strictly necessary is disclosed during authentication. For instance, certain services may only be accessed by adults. The service provider can specify in its access control policy that the user's birth date must be proven before he can consume the particular service. This can be enforced by an authentication procedure using an eID certificate which contains the user's birth date. However, in case of an X.509 certificate, all attributes that are included in the certificate are also released, and provably linked to the user. Moreover, due to uniquely linkable information kept in the certificate, such as the serial number, all transactions that are performed with the same certificate can be linked as well. It should be noted that the user has no impact on the data processing practices once those data are released. This implies that service providers can collaborate to build huge profiles, and use these profiles to possibly discriminate some of them. For instance, insurance companies can impose a higher fee to rich people or refuse an insurance based on collected health information.

Contributions. The present chapter identifies key information of a composite web service that is necessary to allow for reasoning about its privacy properties. It focuses on a blueprint for representing composite web services from a privacy perspective. This includes the identification of different types of stakeholders and interactions between them, as well as the underlying reasons that lead to personal data release, spreading and linking in composite web services. It is noteworthy to signal that the blueprint allows to capture coarse-grained trust relations between entities in the system. Moreover, the blueprint focuses on three categories of relevant privacy feedback that can be given to designers and end-users when that key information is available. The blueprint is applied to a composite travel agency to demonstrate this. The case study also shows that solving complex queries manually is no sinecure. An automated approach is undoubtedly less error-prone and less time-consuming compared to an ad hoc analysis. The framework that supports automatic reasoning on the collected inputs will be discussed in chapter 4 and 5, and will be validated in chapter 6. The work in this chapter is peer-reviewed and published in [67].

The remainder of the present chapter is structured as follows. Section 3.1 gives a general overview of key concepts that must be captured in order to enable privacy inspection. Useful privacy feedback is classified according to three categories in Section 3.2. Basic concepts and terminology are defined in Section 3.3. Section 3.4, 3.5 and 3.6 focus in more detail on certain aspects. First, the impact of authentication technologies and service provider policies on the user's privacy is handled. Second, the impact of trust perceptions on the privacy analysis is described. Section 3.7 applies the concepts to a travel agency. Finally, we reflect about the conceptual blueprint and draw conclusions in Section 3.8 and 3.9 respectively.

3.1 General Overview of the Conceptual Service Model

Figure 3.1 depicts the key concepts that models composite services. The models include two types of stakeholders. In particular, the *user* who interacts with the system and a set of *organizations*. An organization can be a credential issuer and/or a service provider. A service provider offers one or more services that can be consumed by individuals, possibly after disclosing personal information.

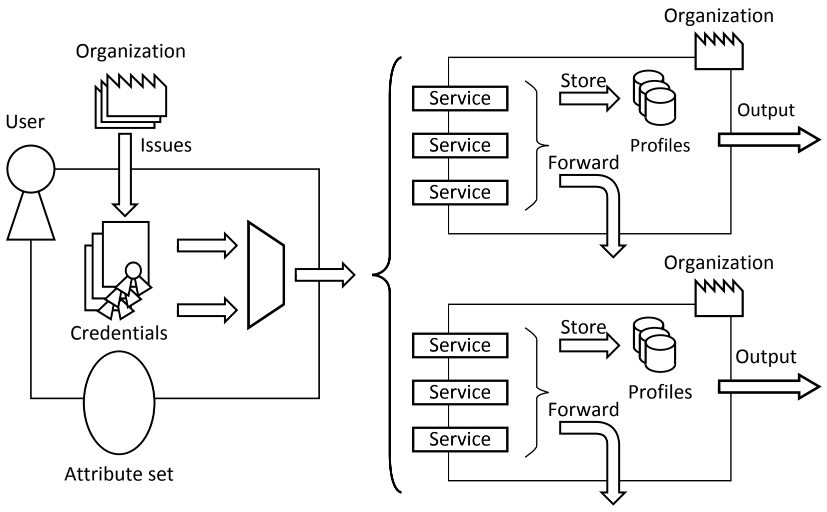

Figure 3.1: Modeling electronic services.

Such information is represented by *attributes* to which two assertion levels apply, including *non-asserted* and *asserted* attributes. Non-asserted attributes reside at the lowest level of assertion. They are disclosed by the user himself and have not been verified or guaranteed by a trusted entity. For instance, the user's name that is manually filled in by himself on a registration form is considered non-asserted. Any organization can improve the qualities of the attributes collected by verifying them. Attributes are then asserted by that organization, by e.g. sending a unique hyperlink after having verified the user's email address. The e-mail address is thus asserted after the user has browsed to the aforementioned hyperlink. In the case attributes are part of a credential, they are considered asserted/certified by an organization if it trusts the credential's issuing organization. Based on the attributes that are collected and linked, pseudonymous or identifiable profiles can be compiled.

A *policy* is assigned to each service. This can typically be extracted from the privacy policies that are currently used by many service providers. It comprises the following parts:

- *Access policy.* This part specifies the attributes that must be released/proved before the service can be accessed.
- *Storage policy.* This specifies the set of data that is stored by the service provider.

- *Distribution policy.* This part mentions the set of data that is forwarded to other organizations.
- *Output policy.* This specifies for instance, the credentials the service issues.

Besides the application itself, authentication technologies and trust relations are also formally modeled. A formal representation of different *authentication technologies* is developed. The model focuses on the attributes that are released and linkabilities that are introduced when using a specific credential technology for fulfilling the access policy. For instance, if a user must only prove to be older than 18, only that fact is disclosed when using Idemix credential technology. However, much more info is released when using an X.509 certificate containing additional personal attributes. Not only the exact birth date, but also all the other attributes are visible for the service provider. Finally, each user can assign a *level of trust* to each organization involved in the system. Contrary to highly trusted organizations, who are believed to obey their (own specified) privacy policies, untrusted ones can exhibit, from the user's point of view, data handling practices that violate the declared ones.

3.2 Feedback

The approach supports different types of feedback that is relevant for an in-depth inspection of the user's privacy. The feedback can be classified according to three classes:

- *Information spreading.* This type of feedback reflects the amount and type of information that is released towards each organization. Organizations gather information during electronic service consumption and store it in profiles. A profile keeps information that is linked to the same individual. Profiles can possibly be merged.
- *Organization behavior.* This shows the impact of collaborating organizations on the user's privacy. Collaboration can lead to discrimination. For instance, an insurance company can discriminate users who are believed to suffer from a certain illness based on information obtained from a commercial e-health provider. However, collaborations may also have benefits for the user. For instance, data from a social network profile can be forwarded to ease registration procedures, which improves the user-experience. Changes in the business landscape, such as two organizations that merge, also influence the user's privacy. For instance, they may be able to merge the profiles of their users.

- *User behavior.* The feedback depicts the impact of the decisions/strategies applied by the user on his privacy and trust. For instance, two e-shops can offer the same e-book at the same price but can apply different privacy policies. Similarly, using an anonymous e-cash system for electronic payments is often more privacy-friendly than using a debit card. Sometimes, multiple alternatives for authentication are possible. For instance, proving to be a resident of Brussels using an anonymous credential is more privacy-friendly than using an identity card that is realized with X.509 certificate technology.

Furthermore, the approach allows to detect violations against ruling policies, or conflicts between policies at different levels or between different stakeholders. For instance, corporate level policies that conflict with governmental policies, or user preferences that conflict with service policies. Changes in policies at different levels, such as governmental, corporate, service specific policies, are covered by this approach as well. The model's formal policy rules can be adapted in accordance to the changes in policies.

3.3 Terminology and Basic Concepts

E represents the set of stakeholders. It includes the *user* $u \in U \subset E$ and the system's *organizations* $o \in O \subset E$. Each organization can offer a set of services, and different organizations can offer the same service. For this reason, $\Sigma \subseteq S \times O$ represents the set of all service instances, with S the set of services. The set of attributes A includes all the attributes related to the user. It should be noted that abstraction is made of the actual attribute values. Hence, attributes only reflect the types of information they contain. For instance, the attribute *eye color* represents data of type *eye color.* Some attributes, such as the user's *eye color* or his *name* and *address*, refer to characteristics of the user. Others can be related to technological identifiers or system parameters, such as the *IP address*, or environmental context, for instance the user's *location.*

Organizations can store the attributes that are collected during service consumption in a profile, which groups related attributes. For instance, a profile can contain attributes revealed during transactions under the same pseudonym. $P \subseteq A \times E$ defines a profile, with E the stakeholder asserting the attributes. An attribute in a profile is considered *asserted* if an organization vouches for its correctness. Attributes that are considered *non-asserted* are supposed to originate from the user and their correctness is not vouched for by a trusted organization. For instance, a user's e-mail address is asserted if the service provider can verify that it belongs to the user. For that purpose, the user is obliged to click on a unique link he received in a special e-mail message. Formally, $(a_{e\text{-}mail}, o) \in P$ depicts an e-mail address that is asserted by organization o while $(a_{e\text{-}mail}, u) \in P$ represents the non-asserted variant. Profiles can be identifiable or pseudonymous:

- A profile P is *identifiable* if $\exists I \in \mathfrak{I} : \forall a \in I, \exists e \in E : (a, e) \in P$, with *identifiable set* $I \subseteq A$ and $\mathfrak{I}$ the set of all identifiable sets. These sets represent combinations of attributes that *sufficiently* identify an individual [143]. The term *sufficiently* is less restrictive than *uniquely* in a sense that it allows for identifying an individual with a certain probability. For instance, someone's first name and surname can be considered to form an identifiable set, although, it is not unlikely to find two people with the same first name and surname. Another example of this is the identification of individuals based on datasets containing meta-data of credit card transactions [63].
- A profile P is *pseudonymous*, if $\exists N \in \mathfrak{N} : \forall a \in N, \exists e \in E : (a, e) \in P$, with *pseudonymous set* $N \subseteq A$ and $\mathfrak{N}$ the set of all pseudonymous sets. These sets contain attributes that form a unique combination. All data associated with it, can be linked. For instance, the set of attributes that represents the fingerprint of a browser [78] can be considered as a pseudonymous set.

A profile P where $\forall a \in I, \exists o \in O : (a, o) \in P$ is *asserted identifiable*. Similarly, a profile can be *asserted pseudonymous*. Multiple profiles can share a pseudonym or identifiable set. In this case all these profiles can be linked and merged together. A profile is *anonymous* if it is not identifiable, nor pseudonymous.

3.4 Authentication Technologies

Users mostly must authenticate before they get access to a service. Authentication technologies T are classified into *claim-based technologies* and *network-based technologies*. In the former case, users authenticate using *credentials* that contain a set of attributes, such as anonymous credentials [139, 44], certificates (e.g. X.509), or a username/password combination. Compared to the other claim-based technologies, anonymous credentials enable the user to remain anonymous during authentication unless identifiable attributes are requested. To support accountability, authentication technologies allow to verify the authenticity of the credential's content. For instance, a digital signature guarantees that the content is authentic. $C \in 2^A \times O \times T$ defines a credential. The definition comprises a set of attributes $\{a, \dots\} \in 2^A$, the issuer $o \in O$, and the technology $t \in T$. Credentials are kept in the user's *credential wallet* W, which can grow in the case a service issues a credential.

On the other hand, network-based authentication technologies are based on user profiles P that are managed by organizations. These can be accessed by the user and possibly external organizations. Usually, this requires authentication by one or more stakeholders E and possibly also the user's consent when he must accept a profile access request. Hence, these technologies are often combined with claim-based authentication mechanisms. For instance, a user accesses a music stream service via his favorite social network account. Therefore, he must authenticate to the social network using his account name and password. Consequently, the music stream service obtains read access to the public part of the user's social network profile and also the visible part of data of his friends in his social network. Furthermore, the application can update the music listening history in his profile that is shared among his social network friends.

3.5 Service Policies

Organizations enforce policies to access their services. These contain rules that define the conditions under which a service can be used. The policy rule-types that are considered in this ontology are *pre-conditions*, *post-conditions*, *data storage*, and *data distribution*. Pre-conditions specify the attributes that must be revealed to an organization and the properties that must be fulfilled. It also includes whether data must be asserted or not. For instance, the user needs to reveal his name and birth date to the music stream provider and prove to be older than 18 before he is able to stream music to his computer. His age must be asserted and, hence, this requires the consumer to use his identity card. Post-conditions are causal rules that describe the output of services. For instance, the music stream subscription service issues a credential to a user. This credential provides access to the complete library of rock music. Data storage rules define the data that is stored in one or more profiles owned by the organization. Data distribution rules specify to which parties data is forwarded.

Each policy applies to a service $(s, o) \in \Sigma$ and is represented as $\pi_{(s,o)} \in \Pi_o$, with Π_o the set of all policies of organization o. Possibly, alternative conditions for service access may exist. For instance, during subscription the user can choose between two different social networks from which personal information is obtained. Different attributes are revealed to the music stream provider depending on the selected social network. Each policy can be a disjunction of alternative sub-policies $\pi_{(s,o)} \equiv \pi^1_{(s,o)} \vee \dots \vee \pi^n_{(s,o)}$, with $n \in \mathbb{N}_0$.

3.6 Trust Relationships

The data that organizations exchange depend on their trust relations. Users and organizations often have conflicting requirements. The former want minimal data disclosure. This view is based on the data minimization principle (i.e. to reveal the least amount of personal information). The more data a user reveals, the higher the risk to harm his privacy. The latter are profit driven and are therefore often interested in building large profiles. This enables them to improve the level of personalization and make their applications more attractive to the user. Different trust relations are applied in this ontology:

- *Users need to trust organizations.* This is represented by *storage trust* relation $T_S \subseteq O$ and *distribution trust* relation $T_D \subseteq O$. Typically, organizations specify this behavior in a privacy policy that is publicly accessible to their users. Users have to accept it before they are granted access to the organization's services. Consequently, it is convenient to specify the user's trust towards the policy set Π_o of organization $o \in O$. We assume that users do not trust other organizations by default ($o \notin T_S$ and $o \notin T_D$). This means that users really opt-in new organizations $o \in O$ in T_S and T_D. Expressing that a user trusts one of these policies presumes that only the specified data is affected by the services. We assume that untrusted organizations will store or forward all data that can be collected. For instance, an organization's policy can stipulate that it only obtains the user's name from his electronic identity card that is based on the X.509 technology. If $o \in T_S$, then only the name is stored by the organization. Otherwise, all the attributes of the user's certificate are collected and stored, although the policy might specify it differently.

- *Interactions require also the trust from organizations towards users.* If data is not asserted, users can provide false information to organizations. For instance, they can enter an incorrect name and address during registration. To exclude inaccurate data, organizations can oblige users to provide asserted data. These trust relations are specified implicitly in the organization's access policy. The trust relation is defined as $T_\Sigma \subseteq A \times O \times \Sigma = \{(a, o_a, (s, o_s)) \mid$ a precondition rule of $\pi_{(s,o_s)}$ specifies that o_a asserts attribute $a\}$.

- *Finally, trust is often required between interacting organizations.* Two types of interactions are defined, namely (1) an organization can accept credentials issued by another organization or (2) an organization can forward data to another organization. Both are implicit in the organization's policies. The former are expressed in the access policy while the latter are expressed in the data distribution rules of the policy.

The definition of this trust relation is $T_O \subseteq O \times O = \{(o_1, o_2) \mid \Pi_{o_1}$ specifies a rule where o_1 accepts credentials or profiles from o_2, or forwards data to $o_2\}$. The model implicitly assumes trust between interacting organizations.

3.7 A Travel Reservation System

The defined concepts are validated through the modeling of a travel reservation system. First, we give an overview of the system. Next, the system is modeled. Finally, we focus on valuable feedback that can be extracted from the model.

3.7.1 Scenario and Setup

A student wants to use an online travel reservation system to book a trip. The travel agency offers triple-packs including the flight, hotel, and a theme park visit. Each item can also be booked separately. This requires users to access the web services of the airline company, the hotel chain and the theme park directly. Special reductions are offered at the theme park's website to students and loyal hotel guests. The airline company and the theme park belong to the same holding. They collaborate to increase efficiency. The student believes that the travel reservation system stores and forwards all data released to it, even though the privacy policy specifies it differently. He also has limited trust in the hotel chain. He assumes they store all collected data. Before he books his trip, he wants to have an insight on the profiles that can be built by the organizations when using a specific booking strategy (*fb1*). Furthermore, he wants to find out the most privacy-preserving strategy to book the theme park tickets (*fb2*). Finally, he wants to verify whether he remains unidentifiable by the theme park when taking a student offer (*fb3*).

Services and organizations. The stakeholders in the system are the user $u \in U$ and the organizations $O = \{o_{travel}, o_{air}, o_{hotel}, o_{park}, o_{gov}, o_{univ}\}$. Note that the holding only supports collaboration between the airline company and the theme park. The government o_{gov} issues governmental electronic identity cards and electronic driving licenses, while an educational institution o_{univ} issues electronic student cards. Table 3.1 gives an overview of the authentication technologies together with the attributes that are included. All credentials are X.509 certificates except the hotel's rewards membership credential c_{rew}, which is a username/password combination. The latter gives access to a membership account P_{rew}. The services Σ are split in two groups Σ' and Σ'',

Table 3.1: Travel reservation system authentication

Credentials		**Issuer**	**Attributes**
eID	c_{eID}	o_{gov}	*name*, *address*, *citizenship*, *DoB*, *profession*, *SSN*
Driving license	c_{driv}	o_{gov}	*name*, *address*, *citizenship*, *DoB*
Student card	c_{stud}	o_{univ}	*name*, *address*, *DoB*, *institute*, *study*
Hotel rewards	c_{rew}	o_{hotel}	*reward id*
Profiles		**Owner**	$A \times E$
Hotel rewards	P_{rew}	o_{hotel}	$(reward\ id, o_{hotel})$, $(name, u)$, $(e\text{-}mail, o_{hotel})$, (DoB, u), $(rewards\ status, o_{hotel})$

with $\Sigma = \Sigma' \cup \Sigma''$. The former are directly accessible by consumers, namely $\Sigma' = \{(book, o_{travel}), (book, o_{air}), (book, o_{hotel}), (book, o_{park}), (reduction, o_{park})\})$. The latter can only be accessed by the travel agency, namely $\Sigma'' = \{(book_{ext}, o_{air}), (book_{ext}, o_{hotel}), (book_{ext}, o_{park})\}$.

Policies and trust relations. The travel agency $(book, o_{travel})$ requires that a consumer at least proves his name and address before he can book a triple-pack using the $(book_{ext}, o)$ service. Those attributes must be asserted by the government using an electronic identity card or driving license. Other attributes, such as his e-mail address and flight destination do not need to be asserted. Only personal data that is strictly necessary to obtain a plane voucher c_{plane}, a hotel voucher c_{hotel}, and a theme park voucher c_{park} are forwarded to the respective organizations. The airline company and hotel chain require at least the user's name and his address when tickets are booked directly using $(book, o_{air})$ and $(book, o_{hotel})$. These must be asserted by the government via the identity card. Other attributes, such as the user's e-mail address, hotel location, and flight destination are non-asserted. All attributes are stored by both organizations. Optionally, tourists can also define diet preferences. Both organizations issue a voucher to the consumer after a successful reservation. A theme park visit $(book, o_{park})$ only requires the user's e-mail address and the date of visit (non-asserted). The visitor gets an entry voucher. All his attributes are stored. Tourists get a reduction when they can prove to be a student or to be silver members in the hotel's rewards program. When the student uses his student card to fulfill the preconditions, the theme park obtains his name, address, and birth date (DoB), which are asserted by the student's school. The consumer can also opt to prove to be a silver member. If so, he is redirected to the hotel chain where he needs to login with his password. After a successful login, the user's *rewards status* is forwarded to the theme park. Attributes that

Table 3.2: Model of the travel reservation system policy to book a travel

$\pi^1_{(book,o_{travel})}$, $\pi^2_{(book,o_{travel})}$ – *alternatives 1 and 2*	
Reveal from c^1_{eID} or c^2_{driv}:	*name*, *address*, *citizenship*, *DoB*
Reveal from u:	*e-mail*, *flight destination*, *date of travel*, *diet*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Reveal from c_{rew}:	*reward id*
Forward to $(book_{ext}, o_{air})$:	*name*, *address*, *citizenship*, *DoB*, *e-mail*, *diet*, *flight destination*, *date of travel*
Forward to $(book_{ext}, o_{hotel})$:	*name*, *address*, *citizenship*, *DoB*, *e-mail*, *room type*, *hotel location*, *reward id*, *date of arrival*, *date of departure*
Forward to $(book_{ext}, o_{park})$:	*e-mail*, *date of visit*
$\pi_{(book,o_{hotel})}$	
Reveal from c_{eID}:	*name*, *address*, *citizenship*, *DoB*
Reveal from c_{rew}:	*reward id*
Reveal from u:	*e-mail*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Store:	*name*, *address*, *citizenship*, *DoB*, *e-mail*, *room type*, *hotel location*, *reward id*, *date of arrival* *date of departure*
Output:	c_{hotel}
$\pi_{(book,o_{air})}$	
Reveal from c_{eID}:	*name*, *address*, *citizenship*, *DoB*
Reveal from u:	*e-mail*, *flight destination*, *date of travel*, *diet*
Store:	*name*, *address*, *citizenship*, *DoB*, *e-mail*, *date of travel*, *flight destination*, *diet*
Output:	c_{air}
$\pi_{(book,o_{park})}$	
Reveal from u:	*e-mail*, *date of visit*
Store:	*e-mail*, *date of visit*
Output:	c_{park}

Table 3.3: Model of the travel reservation system policy to get a reduction

$\pi^1_{(reduction, o_{park})}$ – *first alternative*	
Reveal from u:	*e-mail*, *date of visit*
Reveal from c_{stud}:	*name*, *address*, *DoB*
Store:	*name*, *address*, *DoB*, *e-mail*, *date of visit*
Output:	c_{park}
$\pi^2_{(reduction, o_{park})}$ – *second alternative*	
Reveal from u:	*e-mail*, *date of visit*
Reveal from c_{rew}:	*reward id*
Reveal from P_{rew}:	*rewards status*
Store:	*reward id*, *e-mail*, *date of visit*
Forward to $(get\ status, o_{hotel})$:	*reward id*
Output:	c_{park}

Table 3.4: Model of the travel reservation system policy of the services that are accessible for external organizations

$\pi_{(book_{ext}, o_{hotel})}$	
Reveal from o_{travel}:	*name*, *address*, *citizenship*, *DoB*, *rewards id*, *e-mail*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Store:	*name*, *address*, *citizenship*, *DoB*, *e-mail*, *hotel location*, *room type*, *reward id*, *date of arrival*, *date of departure*
Output:	c_{hotel}
$\pi_{(book_{ext}, o_{air})}$	
Reveal from o_{travel}:	*name*, *address*, *citizenship*, *DoB*, *e-mail*, *flight destination*, *date of travel*, *diet*
Store:	*name*, *address*, *citizenship*, *DoB*, *e-mail*, *flight destination*, *date of travel*
Output:	c_{air}
$\pi_{(book_{ext}, o_{park})}$	
Reveal from o_{travel}:	*e-mail*, *date of visit*
Store:	*e-mail*, *date of visit*
Output:	c_{park}

are released to get a reduction are stored by the theme park. Tables 3.2, 3.3 and 3.4 give a formal overview of the policies that are applied in the travel reservation system.

Multiple trust assumptions can be applied. $T_S = \{o_{air}, o_{park}\}$ means that the user trusts the storage policies of the airline company and theme park. $T_D = \{o_{air}, o_{hotel}, o_{park}\}$ means that the user trusts the distribution policies of the organizations included in T_D. Note that it is quite trivial to modify these sets.

3.7.2 Feedback

To demonstrate the expressive power of our models, different types of feedback are extracted, namely *fb1*, *fb2* and *fb3*.

fb1: User profiles. The end-user and/or designer can get an overview of the information spreading in the system. Details are provided in Tables 3.5 and 3.6. Each service consumption (s, o) leads to a profile $P_{(s,o)}$ containing the attributes gathered by the service. The specific booking strategy has an impact on the type and amount of attributes that are part of the profile. $P^{eID}_{(book,o_{travel})}$ and $P^{driv}_{(book,o_{travel})}$ show the attributes in the profile when the identity card or the driving license are used, respectively. The student only has very limited trust in the travel agency's storage policy and therefore the profile contains all attributes that are released to that organization. Among others, a social security number (SSN) and profession are added to $P^{eID}_{(book,o_{travel})}$ as these are the attributes of the X.509 certificate in the eID card.

The travel agency needs to forward attributes to the airline company, hotel chain, and theme park. In return, vouchers are issued. The student assumes that the travel agency forwards all data that is gathered. However, the profiles $P_{(book_{ext},o_{air})}$ and $P_{(book_{ext},o_{park})}$ only contain the attributes that are specified in the respective storage policies, as the tourist trusts their storage policies. The profile kept by the hotel chain contains all data that is released by the user. Using the driver's license for authentication leaks less information than using the eID card. More specifically, the user's SSN and profession are stored in the profile when the eID is used. The profiles $P_{(book,o_{air})}$, $P_{(book,o_{hotel})}$, and $P_{(book,o_{park})}$ list the set of attributes that are collected if the travel agency does not mediate in the bookings. All profiles comply with the specified storage policies, except the one owned by the hotel. The hotel keeps the user's SSN and profession. It should also be noted that the strategy that is selected to

Table 3.5: Travel reservation system user profiles of the services that are accessible for users

$P^{eID}_{(book,o_{travel})}$ owned by o_{travel}	
Non-asserted:	*e-mail*, *flight destination*, *date of travel*, *diet*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Asserted by o_{gov}:	*name*, *address*, *citizenship*, *DoB*, *profession*, *SSN*
Asserted by o_{hotel}:	*reward id*
$P^{driv}_{(book,o_{travel})}$ owned by o_{travel}	
Non-asserted:	*e-mail*, *flight destination*, *date of travel*, *diet*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Asserted by o_{gov}:	*name*, *address*, *citizenship*, *DoB*
Asserted by o_{hotel}:	*reward id*
$P_{(book,o_{hotel})}$ owned by o_{hotel}	
Non-asserted:	*e-mail*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Asserted by o_{gov}:	*name*, *address*, *citizenship*, *DoB*
$P_{(book,o_{air})}$ owned by o_{air}	
Non-asserted:	*e-mail*, *flight destination*, *date of travel*, *diet*
Asserted by o_{gov}:	*name*, *address*, *citizenship*, *DoB*
$P_{(book,o_{park})}$ owned by o_{park}	
Non-asserted:	*e-mail*, *date of visit*
$P^{stud}_{(reduction,o_{park})}$ owned by o_{park}	
Non-asserted:	*e-mail*, *date of visit*
Asserted by o_{univ}:	*name*, *address*, *DoB*
$P^{rewards}_{(reduction,o_{park})}$ owned by o_{park}	
Non-asserted:	*e-mail*, *date of visit*
Asserted by o_{univ}:	*reward id*, *rewards status*

Table 3.6: Travel reservation system user profiles compiled by the services that are accessible for external organizations

$P^{eID}_{(book_{ext},o_{hotel})}$ owned by o_{hotel}	
Non-asserted:	*e-mail*, *flight destination*, *date of travel*, *diet*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Asserted by o_{gov}:	*name*, *address*, *citizenship*, *DoB*, *profession*, *SSN*
Asserted by o_{hotel}:	*reward id*
$P^{driv}_{(book_{ext},o_{hotel})}$ owned by o_{hotel}	
Non-asserted:	*e-mail*, *flight destination*, *date of travel*, *diet*, *hotel location*, *room type*, *date of arrival*, *date of departure*
Asserted by o_{gov}:	*name*, *address*, *citizenship*, *DoB*
Asserted by o_{hotel}:	*reward id*
$P_{(book_{ext},o_{air})}$ owned by o_{air}	
Non-asserted:	*e-mail*, *flight destination*, *date of travel*, *diet*
Asserted by o_{gov}:	*name*, *address*, *citizenship*, *DoB*
$P_{(book_{ext},o_{park})}$ owned by o_{park}	
Non-asserted:	*e-mail*, *date of travel*

get a reduction at the theme park also has an impact on the attributes in the profile.

fb2: Impact of reduction on privacy. Selecting a specific booking strategy can have an impact on the user's privacy. If the travel agency is used, authenticating with the drive license protects the user's privacy better then when using an eID card. However, the travel agency still collects a lot of valuable personal data. No data is released to the travel agency in case of direct bookings, but this compels the consumer to use his identity card for the hotel and airplane booking. Hence, the user's SSN is revealed to both organizations. The latter is a unique identifier to which a lot of information can be linked. If a student wants a reduction, he must book directly. Apparently, using the rewards membership seems the most privacy-friendly option as only the user's e-mail address and reward id are revealed. In contrast, the user's e-mail address, name and address are revealed if the student card is used. However, the theme park and the airline company belong to the same holding and exchange data. The profiles from the airline company and theme park can be linked by means of the user's e-mail address. Thus, the theme park also knows the user's name and address. Hence, using the student card is slightly more privacy-friendly.

fb3: Reductions are identifiable. Verifying if the student can remain unidentifiable towards the theme park when booking a voucher at reduced price is done by a query based on identifiable sets. The identifiable sets in our system are $\mathfrak{I} = \{I_1, I_2\}$, with $I_1 = \{name\}$ and $I_2 = \{address\}$ where *name* comprises the user's name and surname. Note that the user or designer can define what is *identifiable*. Note that the union $I_3 = I_1 \cup I_2$ is also an identifiable set. A user is unidentifiable to an organization if the following two conditions are met. First, the organization owns no identifiable profile. Second, the organization contains no pseudonymous set that is shared with an identifiable profile of a collaborating organization. According to these conditions, the first condition is only met when the student uses his reward id. However, the second condition is not fulfilled. To explain this, the pseudonymous sets $N_1 = \{e\text{-}mail\}$ is required. $N_1 \subseteq P_{(book, o_{air})} \cap P^{rewards}_{(reduction, o_{park})}$ and $I_3 \subset P_{(book, o_{air})}$ (profile is identifiable). Hence, booking at a reduced price is always identifiable.

3.8 Evaluation

The modeling concepts presented are holistic. They model a wide range of authentication technologies, different components of a service policy, and trust relationships between different stakeholders. Moreover, advanced interactions between the user and service providers can be expressed. The modeling concepts reside at the application layer and assume that applications run on top of anonymous channels such as onion routers [74]. As a consequence, privacy-related properties intrinsically associated to non-anonymous lower layer protocols are abstracted away by the concepts defined. An example of linkable attributes are the IP address or MAC address, that are revealed when protocols run. Nevertheless, this could easily be modeled with pseudonyms.

Different types of feedback is possible when analyzing the user's privacy in modeled systems. The feedback is derived from the model's user profiles that are compiled from service policies and the user's trust perceptions. These profiles represent the set of linkable personal attributes that possibly can be collected by organizations. These profiles are examined for linkage through pseudonyms and identities. The feedback is from the user's viewpoint because it is based on his trust perceptions about the service providers. Other aspects that are modeled, such as collaborations between organizations or used authentication technologies, have consequences for the information spreading as well. The models only consider the information spreading in the system and not the purpose for which personal data is collected. Other complementary approaches [172] can be used to analyze the purpose of collecting personal data. Improvements on the readability of the feedback are necessary. This is clearly illustrated in the case study of the travel reservation system in Section 3.7.2. It shows a large amount of data, which makes it borderline comprehensible.

A static view on the privacy is provided by the models. Consequently, sequences of events over time that influence the user's privacy cannot be expressed. The static view significantly reduces the search space and makes the feedback generation more efficient. Moreover, it makes conclusions stronger because they are not restricted to a finite time-domain as is the case if time is modeled. Models optimize the search space even more by making abstraction of users. In particular, the models only consider one user, which enables to abstract away the actual attribute values. For instance, the attributes *name* and *city* models the real-world proposition *John lives in New York* instead of $name = John$ and $city = New\ York$. However, tuples (*attribute type*, *attribute value*) can easily be added to model attribute values in case it is not feasible to abstract them away. This abstraction may also have consequences in modeling structures such as the list of friends in the user's social network. Tuples (*Friendlist*, *Friend*) can express this.

Real-world data handling practices of organizations are specified in their service policies. These often are merely vague declarations that can be interpreted in multiple ways and are insufficient to correctly extract the actual behavior. Hence, this severely complicates the modeling. Existing policy specification languages, such as P3P [182], CARL [43], and XACML/SAML based languages [22] help service providers to exactly express their policies in a standard format that is machine readable. This enables to automate the extraction of policy rules so that these can easily be included in the model.

The models are flexible enough to support system modifications. Changes in service policy can be handled automatically in case they are specified in machine readable languages. Otherwise, modifications in the service policies require manual interventions. Other types of changes, such as collaborations between organizations or newly added organizations are also added manually to the model. For instance, adding a new organization o to the system involves adding the new organization to the set of stakeholders E and the set of organizations O. New services are added to the sets Σ and S and new service policies of organization o are added to Π_o.

Future work can focus on how this approach can be integrated in tools for automated privacy analysis of systems or the design of privacy-friendly services. Such tools can verify the compliance of service policies with policies at corporate or legislative level. The tools can also be added to end-user applications, such as a browser plug-in to verify whether the data that will be revealed complies with the user defined policy. The browser plug-in consults an online available model of the application to notify a user if, e.g. his SSN will be revealed while his privacy preferences state to keep it hidden from commercial organizations.

3.9 Conclusions

This chapter identified a set of key concepts for modeling advanced electronic services in which multiple organizations collaborate. Once a system is modeled, the designer and end-user can analyze privacy properties based on varying trust assumptions. The concepts are applied to the modeling of a composite web service, namely a travel agency in which multiple stakeholders are involved. Finally, it is shown that feedback can be extracted that is meaningful for both end-users and service designers.

Chapter 4

Modeling Approach

Chapter 3 identifies key information that is necessary in order to inspect privacy properties in advanced electronic services. The information is classified according to multiple categories. Chapter 3 also shows that relevant privacy feedback can be given to both end-users and system designers. For instance, the end-user can inspect the impact of collaborations between highly untrusted entities in the system. Similarly, system designers can detect conflicts between corporate level policies and legislative privacy policies. Moreover, extracting useful privacy feedback manually proved to be not trivial in more complex settings. It is clear that semi-automating privacy inspection is useful in many settings. To support this, additional rules must be defined that express additional knowledge. For instance, generic rules can be defined that express the impact of using certain credentials technologies on the resulting privacy properties of the system. Although a security expert knows that anonymous credentials are more privacy-friendly than X.509 certificates, it is not always clear what impact design and runtime choices have on the system privacy. The impact of selecting a particular credential technology on the privacy also depends on the storage and processing policies, the collaboration between service providers, etc.

Contributions. This chapter focuses on a logic-based framework that supports automatic reasoning about privacy properties in advanced electronic services. The concepts that are identified in chapter 3 are mapped to framework components. The framework also represents the interrelationships between these concepts. Furthermore, an instantiation of the framework is realized in a knowledge base system, namely IDP. It is selected for two major reasons, namely (a) the potential of IDP to represent knowledge and reasoning and

(b) the relevance of IDP from a research perspective. The framework assists modelers in representing advanced composite web services by offering reusable building blocks. This means that some components in the framework can be reused when a modeler wants to model a new system and reason about it. The present chapter also presents an approach to *automatically* extract relevant privacy properties from the models that were built. To validate the framework, it is applied to the modeling and privacy analysis of a web shop.

The contributions in this chapter are peer-reviewed in [66].

The remainder of this chapter is structured as follows. First, a web shop and its conceptual model are described in Section 4.1. The system mentioned is used throughout the remainder of this chapter. Section 4.2 gives an overview of the modeling framework for logic-based reasoning, and introduces the major components. Section 4.3 focuses on the input model after which the inference strategy is described in more detail. This chapter ends with an evaluation of the framework (see 4.5) and conclusions (see 4.6).

4.1 Web Shop Scenario

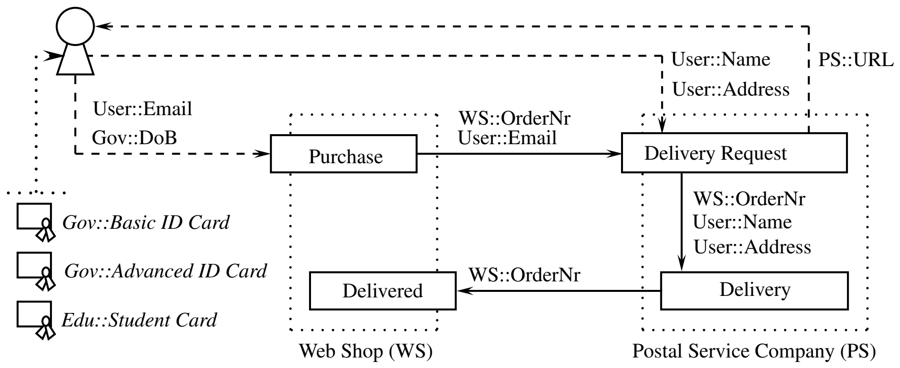

Figure 4.1: Conceptual Model of a Web Shop and Delivery Service.

The present section presents a composite web service that is used as an example to clarify the concepts throughout the rest of this chapter. The major components are depicted in Figure 4.1. The web service consists of two service providers, namely a web shop (WS) and a postal service company (PS). A consumer can purchase products at an online website managed by WS. As the shop wants to exclude minors, a consumer needs to prove to be older than 18. The web shop supports two credential technologies to fulfill that prerequisite. Either the consumer uses a basic electronic identity card or a more privacy-friendly one. It is assumed that the former consists of an X.509 certificate which embeds the customer's *name*, *address*, *birth date* (DoB), and *social security number* (SSN), and the latter can be an anonymous credential which contains the same attributes. Anonymous credentials allow to selectively disclose attributes. More specifically, a user can prove to be older than 18 without revealing his birth date. Students can purchase goods at a reduced price if they show their electronic student card. It consists of an X.509 certificate with the student's *name*, *address*, and *institute* as attributes. Consumers also must disclose their email address when they want to buy a product. When a purchase transaction is completed, WS sends transaction details together with personalized recommendations to the consumer's email address. WS also shares the order number (OrderNr) and email address (Email) with PS, and redirects the user to PS where he must fill in his *name* and *address* in the delivery information form. PS collects the data and sends a unique hyperlink to the user's mailbox, which can be used to track the delivery status. After delivery, PS informs WS that the delivery corresponding to *OrderNr* was successful. The data exchanged at lower communication layers is for simplicity omitted.

4.2 Modeling Framework

When invoking a service, information is disclosed to the organization providing the service. The information includes a user's personal information, such as his birth date or email address, or information generated by the service or the user during service consumption. For example, an order number that is generated by the service. The information can both be stored by the organization providing the service or be forwarded to other services. These services can, in turn, also store or forward the collected information. The organizations involved can use this information to compile user profiles, which collect the attributes that belong to the same user. The user profiles that organizations may possibly build, form the basis from which conclusions about the user's privacy are drawn. This privacy analysis considers that actual data handling practices of service providers are hidden to users and are possibly not compliant with declared practices. E.g., a highly untrusted service provider is suspected to store – and possibly distribute – more attributes than specified in its service policy. Hence, the user's trust perceptions of a service provider's data handling practices are considered as well.

Figure 4.2 gives an overview of the major components in the framework of the modeling approach. The *System Independent Modeling Component* consists

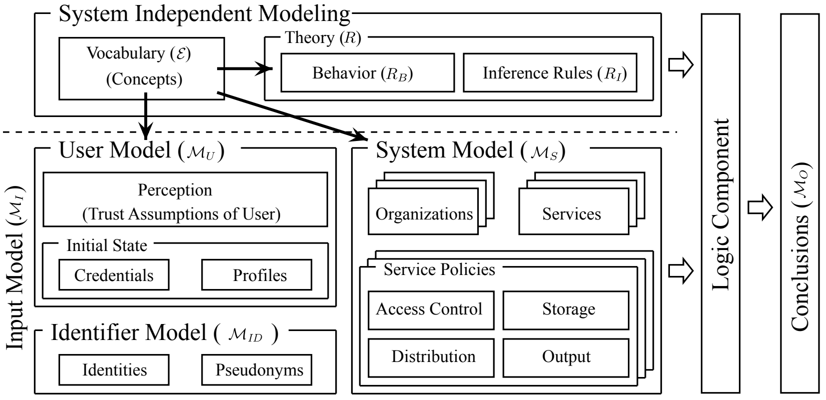
System Independent Modeling
Vocabulary ($\mathcal{E}$)
(Concepts)
Theory (R)
Behavior (R_B)
Inference Rules (R_I)
Input Model ($\mathcal{M}_I$)
User Model ($\mathcal{M}_U$)
Perception
(Trust Assumptions of User)
Initial State
Credentials
Profiles
Identifier Model ($\mathcal{M}_{ID}$)
Identities
Pseudonyms
System Model ($\mathcal{M}_S$)
Organizations
Services
Service Policies
Access Control
Storage
Distribution
Output
Logic Component
Conclusions ($\mathcal{M}_O$)

Figure 4.2: Framework for logic reasoning about privacy properties in complex electronic services.

of a vocabulary ($\mathcal{E}$) and a set of formal sentences (R) that denote generic propositions on systems and can be reused in whatever system (or application) is modeled. For instance, sentences that describe the properties of X.509 certificates. The *Input Model* ($\mathcal{M}_I$) is system independent and consists of a *User Model* ($\mathcal{M}_U$), a *System Model* ($\mathcal{M}_S$), and an *Identifier Model* ($\mathcal{M}_{ID}$). $\mathcal{M}_U$ defines the user's initial state and trust perceptions. Therefore, this component needs to be replaced if a new type of user is modeled. Similarly, $\mathcal{M}_S$ is application specific and needs to be replaced if a new application is modeled. $\mathcal{M}_{ID}$ defines sets of attributes that are each sufficient to distinguish a single user. Some sets are identifiable, others are pseudonymous. For instance, the user's name and birth date can be an identifiable set, while an email address can be a pseudonymous singleton. A modeler can then decide how strict the notion of identifiability and pseudonymity should be. Finally, the *Logic Component* represents a knowledge base tool that automatically infers *privacy conclusions* ($\mathcal{M}_O$) based on the System Independent Model and the Input Model. The rest of this section discusses the major components in more detail.

Vocabulary. The Vocabulary $\mathcal{E}$ consists of three parts, namely $\mathcal{E}_I$, $\mathcal{E}_R$ and $\mathcal{E}_O$. The *input vocabulary* $\mathcal{E}_I$ includes the symbols required for the specification of new systems (or services). $\mathcal{E}_R$ defines theory dependent symbols for reasoning, which are meant for internal use only. The *output vocabulary* $\mathcal{E}_O$ includes the symbols necessary to express conclusions. Consequently, new symbols must be defined when new types of conclusions need to be supported.

Theory. The Theory R comprises a set of logic sentences that are expressed over the vocabulary $\mathcal{E}$. It is divided into a fixed *behavior* theory R_B and an interchangeable *inference* theory R_I. R_B defines the concepts that are fixed for the same kind of inferences. For instance, the non-functional properties of authentication technologies are expressed here. More specifically, formal rules express that multiple transactions with the same X.509 certificate can be linked and that all attributes within an X.509 certificate are released to the service provider, whereas anonymous credentials support unlinkability and selective disclosure of attributes. R_I contains the theory that is required to draw conclusions from a given input model. Depending on the type of conclusions, another (pluggable) inference theory component is used. Section 4.4 focuses on an instance of R_I that is used to draw conclusions from user profiles kept at different organizations.

System Model. The System Model $\mathcal{M}_S$ defines (a) the set of organizations involved in the composite web service, (b) the set of services they offer, and (c) a set of service policies assigned to those services. Four types of policies are assigned to each service: the access control policy defines the attributes that must be disclosed (or proven) before a service can be consumed; the storage policy defines the set of attributes that will be stored by the service provider (i.e. typically a subset of the attributes released by the user); the distribution policy defines the set of attributes forwarded to other service providers (i.e. also a subset of attributes released by the user); and the output policy defines data generated during service consumption. For instance, a new account that is created or the list of purchased goods.

User Model. The User Model $\mathcal{M}_U$ defines the set of credentials of a single user (only one user of the services is considered in this appraoch). A credential, such as an electronic identity card, is a collection of attributes, certified by an issuer and managed by the user. Credentials and profiles can be used to fulfill access control policies. $\mathcal{M}_U$ also defines the trust perception of the user. More specifically, the framework allows for specifying whether a user trusts that the expressed storage and distribution practices are observed by an organization. Different users may have different trust assumptions. For instance, privacy concerned users may have less trust in the specified data practices than other users. The framework allows to model multiple types of users.

Logic Component. The Logic Component consists of a logic tool that automates the reasoning on formal models. IDP [185] has been selected for the realization of this logic reasoning system. IDP is a knowledge base system that reasons on models specified in the IDP language, which is a formal language based on typed first-order logic. This language has successfully been used for the formal specification of services. IDP reasons on an input model $\mathcal{M}_I$ using a theory R. The results are presented in the output model $\mathcal{M}_O$.

4.3 Input Model Specification

This section shows what elements and relations a modeler needs to express when modeling a particular system. The elements and relations are instantiations of the symbols in the input vocabulary $\mathcal{E}_I \subset \mathcal{E}$, and together form the Input Model $\mathcal{M}_I = \mathcal{M}_S \cup \mathcal{M}_U \cup \mathcal{M}_{ID}$. The presented input specification is based on the web shop scenario described in section 4.1 and can be expressed with the IDP language.

Organizations and Attributes. The organizations that are involved in the system being modeled are part of the *Organization* domain. They participate in services provided to the user or they can issue credentials required by services. The data that is exchanged when invoking services are modeled by attributes that are part of the *Attribute* domain. These represent data types instead of the actual data values. This abstraction is possible due to the single-user approach.

IDP Listing 4.1: Partial input model representing organizations and attributes.

$$\mathcal{M}_S :: \texttt{type}\ \text{Organization} = \{\mathit{Government}; \mathit{University}; \mathit{WS}; \mathit{PS}\}$$

$$\mathcal{M}_S :: \texttt{type}\ \text{Attribute} = \{ \\ \mathit{Name}; \mathit{Address}; \mathit{DoB}; \mathit{SSN}; \mathit{Institute}; \mathit{OrderNr}; \mathit{URL}; \mathit{Email}\}$$

Identities and Pseudonyms. The identifiers in the model, which can link profiles, are elements of the *Identifier* domain. Identifiers can belong to the *Identity* or the *Pseudonym* domain. The former sufficiently identify an individual and can link a profile to that individual, while the latter only can associate user profiles that share the same pseudonym. $\mathit{IdentifierAttr}(\mathit{Identifier}, \mathit{Attribute})$ assigns a set of attributes to each identity and pseudonym. In our system, one identity $\mathit{Identity}_1 \in \mathfrak{I}$ and four pseudonyms (i.e. singletons) $\mathit{Nym}_i \in \mathfrak{N}$ are defined. The modeler specifies that the user's

Name and *Address* belong to $Identity_1$ (i.e. collecting both attributes will make someone identifiable), and that the user's *SSN*, *OrderNr*, the unique *URL* provided by the delivery service provider, and *Email* address are unique pseudonyms. Note that another modeler could specify the user's *SSN* as an identity instead of a pseudonym, or specify that the user's *Name* is sufficient to identify the user.

IDP Listing 4.2: Partial input model representing identities and pseudonyms.

$\mathcal{M}_{ID}$ :: `type` Identifier $= \{Identity_1; Nym_1; Nym_2; Nym_3; Nym_4\}$

$\mathcal{M}_{ID}$ :: `type` Identity `isa` Identifier $= \{Identity_1\}$

$\mathcal{M}_{ID}$ :: `type` Pseudonym `isa` Identifier $= \{Nym_1; Nym_2; Nym_3; Nym_4\}$

$\mathcal{M}_{ID}$ :: $\mathsf{IdentifierAttr}(\text{Identifier}, \text{Attribute}) = \{$
$(Identity_1, Name); (Identity_1, Address);$
$(Nym_1, SSN); (Nym_2, OrderNr); (Nym_3, URL); (Nym_4, Email)\}$

User Trust Perception. These relations (part of $\mathcal{M}_U$) allow to specify trust perceptions of the user with respect to the organizations in the model. The relations below specify that the user only believes that the postal service company abides his storage policy, and that the web shop observes his distribution policy.

IDP Listing 4.3: Partial input model representing the user trust perception.

$\mathcal{M}_U$ :: $\mathsf{StorageTrust}(\text{Organization}) = \{PS\}$

$\mathcal{M}_U$ :: $\mathsf{DistributionTrust}(\text{Organization}) = \{WS\}$

Credential Specification. All credentials owned by the user are listed in the *Credential* domain. In the web shop scenario, the user may have three credentials, namely, a basic identity card, a privacy-friendly identity card, and optionally a student card. For each credential, multiple properties need to be defined, namely the set of attributes included in the credential, the underlying credential technology that is used, and the credential issuer. In the example, three credential elements are specified in *Credentials*. For instance, the user's name and address are embedded in the *BasicIDCard*, which is an X.509 certificate issued by the *Government*.

IDP Listing 4.4: Partial input model representing credentials.

$$\mathcal{M}_U :: \texttt{type}\ \text{Credential} = \{\mathit{BasicIDCard}; \mathit{PrivIDCard}; \mathit{StudentCard}\}$$

$$\begin{aligned}\mathcal{M}_U :: \mathsf{CredAttr}(\text{Credential}, \text{Attribute}) = \{& \\ &(\mathit{BasicIDCard}, \mathit{Name}); (\mathit{BasicIDCard}, \mathit{Address}); \ldots, \\ &(\mathit{PrivIDCard}, \mathit{Name}); \ldots\}\end{aligned}$$

$$\begin{aligned}\mathcal{M}_U :: \mathsf{CredTech}(\text{Credential}) : \text{ClaimbasedTech} = \{& \\ &\mathit{BasicIDCard} \rightarrow \mathit{X509}; \mathit{PrivIDCard} \rightarrow \mathit{Idemix}; \ldots\}\end{aligned}$$

$$\begin{aligned}\mathcal{M}_U :: \mathsf{CredIssuer}(\text{Credential}) : \text{Organization} = \{& \\ &\mathit{BasicIDCard} \rightarrow \mathit{Government}; \mathit{PrivIDCard} \rightarrow \mathit{Government}; \ldots\}\end{aligned}$$

Services and Service Policies. Services are uniquely referred in the model by identifiers from the *ServiceIdentifier* domain. The model assigns a type from the *ServiceType* domain to each service and associates the services with the organizations that provide them. The relations below focus on the specification of a service identified by *BasicPurchaseServ*. In addition, for each service the applicable policies are specified. The access control policy, for instance PurchaseAccessPol, specifies the attributes that need to be disclosed and their source in order to grant access (e.g. reveal *DoB* from the *BasicIDCard*). The storage policy specifies the attributes that are stored and the distribution policy defines which attributes are sent to another service. As shown below, *BasicPurchaseServ* forwards *OrderNr* to the postal company when invoking *DeliveryRequestServ*. Optionally, an output policy specifies the data returned to the user when the service was invoked.

IDP Listing 4.5: Partial input model representing services and service policies.

$\mathcal{M}_S :: \texttt{type}\ \text{ServiceType} = \{\textit{Purchase}; \textit{Delivery}; \textit{DeliveryRequest}; \textit{Delivered}\}$

$\mathcal{M}_S :: \texttt{type}\ \text{ServiceIdentifier} = \{\textit{BasicPurchaseServ}; \textit{PrivPurchaseServ}; \textit{DeliveryRequestServ}; \ldots\}$

$$\begin{aligned}\mathcal{M}_S :: \mathsf{Service}&(\text{ServiceType}, \text{Organization}) : \text{ServiceIdentifier} = \{\\ &(\textit{Purchase}, \textit{WS}) \to \textit{BasicPurchaseServ};\\ &(\textit{Purchase}, \textit{WS}) \to \textit{PrivPurchaseServ};\\ &(\textit{DeliveryRequest}, \textit{PS}) \to \textit{DeliveryRequestServ}; \ldots\}\end{aligned}$$

$$\begin{aligned}\mathcal{M}_S :: \mathsf{ServicePolicies}&(\text{ServiceIdentifier}, \text{AccessPolicy}, \text{StoragePolicy},\\ &\text{DistributionPolicy}, \text{OutputPolicy}) = \{\\ &(\textit{BasicPurchaseServ}, \textit{PurchaseAccessPol}, \textit{PurchaseStorePol},\\ &\quad \textit{PurchaseDistrPol}, \textit{PurchaseOutPol}); \ldots\}\end{aligned}$$

$$\begin{aligned}\mathcal{M}_S :: \mathsf{RevealAttr}&(\text{AccessPolicy}, \text{AttrSrc}, \text{Attribute}) = \{\\ &(\textit{PurchaseAccessPol}, \textit{BasicIDCard}, \textit{DoB}); \ldots\}\end{aligned}$$

$$\begin{aligned}\mathcal{M}_S :: \mathsf{StoreAttr}&(\text{StoragePolicy}, \text{Attribute}) = \{\\ &(\textit{PurchaseStorePol}, \textit{OrderNr}); \ldots\}\end{aligned}$$

$$\begin{aligned}\mathcal{M}_S :: \mathsf{DistrAttrTo}&(\text{DistributionPolicy}, \text{ServiceIdentifier}, \text{Attribute}) = \{\\ &(\textit{PurchaseDistrPol}, \textit{DeliveryRequestServ}, \textit{OrderNr}); \ldots\}\end{aligned}$$

$\mathcal{M}_S :: \mathsf{OutputAttr}(\text{OutputPolicy}, \text{Attribute}) = \{(\textit{PurchaseOutPol}, \textit{URL})\}$

4.4 Inferring Privacy Based on Trust

This section presents the general idea behind the inference rules R_I that are used for the automated reasoning on the input model $\mathcal{M}_I$. The inference strategy starts from a *service invocation graph*, which is a representation of the possible invocations of services which may either be directly or indirectly invoked by a user. It is used to compile user profiles that belong to organizations, based on the user's trust in the involved organizations. These user profiles show for each organization the personal information that can be linked to the same user and possibly make him identifiable. In the rest of this section, this inference strategy is further discussed and demonstrated using the web shop scenario.

Service Invocation Graph. This is an acyclic directed graph $G = \langle \Pi, \mathcal{F} \rangle$, where each node $\pi_o \in \Pi_o$ represents a service provided by organization $o \in O$, with $\Pi_o \subseteq \Pi$ the set of services provided by o. Each edge $f \in \mathcal{F} \subseteq \Pi \times \Pi$ defines the invocation of a service by another service. These edges are derived from the data distribution specifications of the service policies. Nodes without incoming edges, denoted by $\pi_o^* \in \Pi^* \subseteq \Pi$, are only invokable by users. We assume that services can neither directly nor indirectly invoke themselves. As a result, this graph is acyclic and a *strict order relation* $<_G$ on Π that expresses the relative invocation order between services can be defined. For instance, $\pi_o^i <_G \pi_{o'}^j$ expresses that $\pi_{o'}^j$ is invoked by π_o^i (directly or indirectly), with π_o^i, $\pi_{o'}^j \in \Pi$. Note that a service policy is assigned to each service.

The service invocation graph for the web shop scenario is illustrated in Figure 4.3. As an example, when a user makes a purchase at the web shop, the web shop sends a delivery request to the postal service. There are four alternative invocations that depend on the selected purchase option. One of them is presented by an edge in the service invocation graph from the *WS:BasicPurchase* service (π_{WS}^{a*}) to the *PS:DeliveryRequest* service (π_{PS}^1). The other incoming edges of π_{PS}^1 represent the other alternatives.

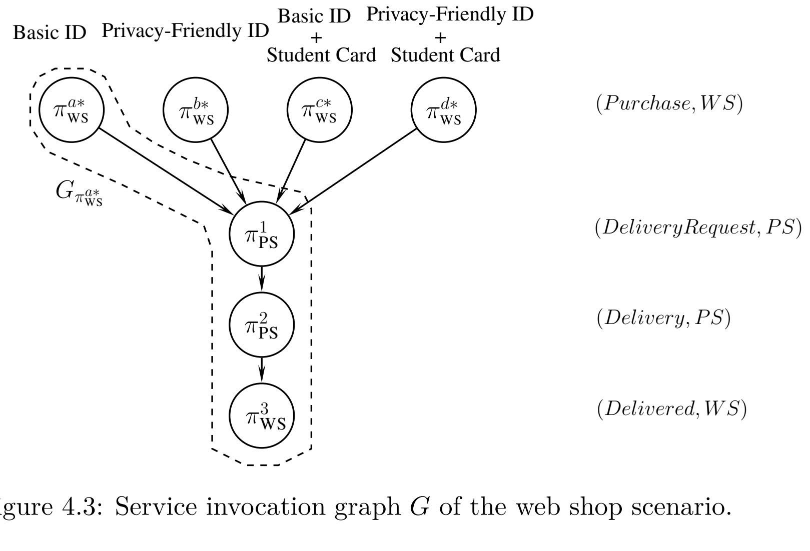

Figure 4.3: Service invocation graph G of the web shop scenario.

Figure 4.3 also shows the sub-graph $G_{\pi_{\mathrm{WS}}^{a*}}$ of G. This sub-graph includes all services that are invoked when invoking π_{WS}^{a*} . Semantically, it corresponds to the services in the web shop scenario that are invoked when a product is purchased with a basic ID card. More formally, $G_{\pi_o^*} = \langle \Pi_{\pi_o^*}, \mathcal{F}_{\pi_o^*} \rangle$ is the sub-graph for a user invocable service π_o^* that is defined by $\Pi_{\pi_o^*} = \{\pi_{o'} \in \Pi \mid \pi_o^* <_G \pi_{o'}\} \cup \{\pi_o^*\}$ and $\mathcal{F}_{\pi_o^*} = \{(\pi_{o'}^i, \pi_{o''}^j) \in \mathcal{F} \mid \pi_{o'}^i \in \Pi_{\pi_o^*} \wedge \pi_{o''}^j \in \Pi_{\pi_o^*}\}$, with $o', o'' \in O$. We assume that the sub-graph contains no alternative service invocations (i.e. nodes have at most one incoming edge). Therefore, it forms a tree with root node π_o^*. Starting from these sub-graphs $G_{\pi_o^*}$, user profiles are compiled.

User Profiles. In this approach, we analyze the user's privacy even if organizations are not trustworthy. Therefore, the user profiles that are built, are not only based on what is specified in the service policies, but also on the user's trust perception. In other words, profiles satisfy the user's expectations. The user's trust is specified in relation to the data storage and data distribution policies of organizations:

- *Storage trust* ($o \in T_S$), specifies that for each service $\pi_o \in \Pi_o$, the user believes that only the attributes Ω_{π_o} specified in the storage policy of π_o, are actually stored. Without storage trust, all data possibly collected by these services, is expected to be stored.
- *Distribution trust* ($o \in T_D$), specifies that for each service $\pi_o \in \Pi_o$, the user believes that only the attributes Θ_{δ_o} specified in the distribution policy of π_o, are forwarded to the specified service. Otherwise, all data possibly collected by these services, is expected to be forwarded.

The compilation of user profiles, that organizations build by executing a *user invokable service* π_o^*, consists of two steps. In the first step, the sub-profile $P(\pi_{o'}|\pi_o^*)$ is compiled for each service $\pi_{o'} \in \Pi_{\pi_o^*}$. It defines the attributes that are persistently kept by o' when executing $\pi_{o'}$. Next, the sub-profiles of different services that are kept by an organization, are combined to compile its user profiles.

STEP 1: compiling sub-profiles for π_o^*. A sub-profile $P(\pi_{o'}|\pi_o^*)$ is compiled for each of the nodes $\pi_{o'}$ of the sub-graph $G_{\pi_o^*}$. These store the attributes that are collected by $\pi_{o'}$ with respect to the user's expectations. Equation (4.1) gives the attributes that are part of sub-profile $P(\pi_{o'}|\pi_o^*)$. It consists of two conditional rules. The first gives the attributes stored in the case the user trusts the organization's storage policy. This is trivial because only the attributes are stored that are explicitly specified in the storage policy of that service. The other rule lists the attributes stored in case that there is no storage trust. This involves different attribute sources, namely:

- *disclosed attributes* $\Upsilon_{\pi_{o'}}$ contain the attributes revealed due to the access policy of $\pi_{o'}$. It not only consists of the attributes required by the access policy, but also of the additional attributes disclosed due to the used authentication technology. For instance, in the case of X.509 certificates, although the user is only required to reveal his *birth date* to access a service, he reveals all other attributes included in the certificate as well.
- *generated data* $\Gamma_{\pi_{o'}}$ contains the data generated during service consumption. For instance, when a user purchases a product in the web shop an order number will be generated.
- *forwarded attributes* $\Phi(\pi_{o'}|\pi_o^*)$ consist of the data that is forwarded by services $\pi_{o''} \in \Pi_{\pi_o^*}$ to $\pi_{o'}$, with $\pi_{o''} <_G \pi_{o'}$.

The attributes from $\Upsilon_{\pi_{o'}}$ and $\Phi(\pi_{o'}|\pi_o^*)$ are derived from the access policy of the service $\pi_{o'}$, which is trivial. Finding the attributes that are forwarded to $\pi_{o'}$ is more complex and depends on the distribution trust with respect to the services that forward data to this service. These attributes are given by Equation (4.2). This equation gives the attributes that are forwarded to service $\pi_{o'}^i$ by service $\pi_{o''}^j$, which directly invokes it (i.e. $\pi_{o''}^i = \mathit{pred}(\pi_{o'}^j)$). It consists of two rules that cover both the case in which there is distribution trust, and the case in which there is no distribution trust. In the former case, the attributes are specified in the distribution policy of $\pi_{o''}^j$. In the latter case, $\pi_{o''}^j$ forwards all the attributes that it generates itself ($\Gamma_{\pi_{o''}^j}$) and that are disclosed to it due to its own access policy $\Upsilon_{\pi_{o''}^j}$ or that have been forwarded by its predecessor $\Phi(\pi_{o''}^j|\pi_o^*)$.

This is applied to the web shop scenario, which is depicted in Figure 4.1. The data that is received by service *WS:Delivered* from the perspective of a customer depends on his trust perception of the parties involved. The service only receives the order number *OrderNr* from service *PS:Delivery*, if the user has distribution trust in the postal service company PS ($PS \in T_D$). Whereas if $PS \notin T_D$, *WS:Delivered* may not only receive *OrderNr* from *PS:Delivery*, but also the *Name* and *Address* of the user and the *URL* to track the delivery status of the purchase.

$$P(\pi_{o'}|\pi_o^*) = \begin{cases} \Omega_{\pi_{o'}} & , o' \in T_S \\ \\ \Upsilon_{\pi_{o'}} \cup \Gamma_{\pi_{o'}} \cup \Phi(\pi_{o'}|\pi_o^*) & , o' \notin T_S \end{cases} \tag{4.1}$$

$$\Phi(\pi^i_{o'}|\pi^*_o) = \begin{cases} \Theta_{\pi^j_{o''}} & , o'' \in T_D \\ \Upsilon_{\pi^j_{o''}} \cup \Gamma_{\pi^j_{o''}} \cup \Phi(\pi^j_{o''}|\pi^*_o) & , o'' \notin T_D \end{cases} \tag{4.2}$$

$$\text{with } \Phi(\pi_{o'}|\pi^*_o) = \emptyset \text{ for } \pi_{o'} = \pi^*_o, \text{ and } \pi^j_{o''} = \mathit{pred}(\pi^i_{o'})$$

STEP 2: Compiling user profiles. The user-profiles of the organizations are compiled using their sub-profiles that were generated in the previous step. This requires the presence of one or more shared identifiers $id \in \mathfrak{I} \cup \mathfrak{N}$. Initially, all sub-profiles that include an identifier, are considered as user profiles. User profiles sharing a same identifier are merged into a single user profile that contains all the attributes from these user profiles. Since identifiers may be defined as a set of attributes (see Section 4.3), and these attributes may be spread over different user profiles, merging these profiles may have as side-effect that a new identifier appears in the resulting profile. Therefore, this step is repeated until no new identifiers are formed in the merged user profiles. As a result, a number of user profiles $P^{id}_{o'}(G|\pi^*_o)$ are compiled with attributes that can be linked by organization o' to a unique identifier id after the user invoked service π^*_o. Hence, the organization may keep multiple user profiles of the same user who cannot be linked to each other because they have no shared identifier. Note that the same user profile may contain multiple identifiers.

Impact of Multiple Service Invocations on Profiles. Until now, we have focused on the profiles that can be compiled when a user invokes a single service π^*_o. However, in practice, a user typically invokes multiple services over time. Hence, user profiles often grow gradually. For instance, in the web shop scenario, the user purchases a product with a *BasicIDCard* and *StudentCard* (i.e. service invocation $\pi^{b*}_{\mathrm{W}s}$). Later on, he receives an *PrivIDCard* from the government and quits high school. If he then purchases an item, service invocation $\pi^{b*}_{\mathrm{W}s}$ is applied. Similarly, consider a web shop that offers three services to users, namely *Register*, *Browse*, and *Purchase*. When browsing the site, a persistent cookie is installed that is requested each time to personalize offers. Before a user can *Purchase* a product, he must *Register*. At that phase, additional attributes are possibly disclosed. In that case, it does not make sense to only evaluate the impact of the service invocation *Purchase* on the profiles kept by organizations. On the contrary, the impact of multiple types of service invocations on the profiles is relevant. Our framework easily compiles profiles that result from multiple invocations. $P^{id}_{o'}(G|[\pi^{a*}_{o_a}, \pi^{b*}_{o_b}, ..., \pi^{x*}_{o_x}])$ defines the expected profile kept by o' with attributes linked to id after a sequence of

Table 4.1: User's additional attributes in the web shop's user profile related to the case where both organizations are trusted.

	$T_S = \{PS\}$, $T_D = \{WS, PS\}$	$T_S = \{PS\}$, $T_D = \{WS\}$
$P^{id}_{\text{WS}}(G\|\pi^{a*}_{\text{WS}})$	*Name*, *Address*, *DoB*, *SSN*	*Name*, *Address*, *URL*, *DoB*, *SSN*
$P^{id}_{\text{WS}}(G\|\pi^{b*}_{\text{WS}})$	—	*Name*, *Address*, *URL*
$P^{id}_{\text{WS}}(G\|\pi^{c*}_{\text{WS}})$	*Name*, *Address*, *DoB*, *SSN*, *Institute*	*Name*, *Address*, *URL*, *DoB*, *SSN*, *Institute*
$P^{id}_{\text{WS}}(G\|\pi^{d*}_{\text{WS}})$	*Name*, *Address*, *Institute*	*Name*, *Address*, *URL*, *Institute*

services $[\pi^{a*}_{o_a}, \pi^{b*}_{o_b}, ..., \pi^{x*}_{o_x}]$ are invoked by the user. $P^{id}_{o'}(G|[\pi^{a*}_{o_a}, \pi^{b*}_{o_b}, ..., \pi^{x*}_{o_x}])$ is constructed by merging all the profiles $P^{id}_{o'}(G|\pi^{i*}_{o_i})$ with $\pi^{i*}_{o_i} \in [\pi^{a*}_{o_a}, \pi^{b*}_{o_b}, ..., \pi^{x*}_{o_x}]$.

4.5 Evaluation

To validate the framework, both the theory R – including the behavior R_B and inference rules R_I – and the input model $\mathcal{M}_I$ for the web shop scenario were realized[1] in IDP. This is used to compare the alternative services of *WS::Purchase* (i.e. π^{a}_{WS}, π^{b*}_{WS}, π^{c*}_{WS}, and π^{d*}_{WS}) under different trust perceptions of the user. When considering the case in which both the web shop and postal service company, comply with their service policies (i.e. this corresponds with a user perception where services are fully trusted), the resulting user profiles are independent of the used credentials. WS stores only the *OrderNr* and *Email*. PS instead, stores the user's *Name* and *Address*, the *OrderNr* and the tracking *URL*. Both organizations have user profiles that are pseudonymous. The postal service can also identify the user by his *Name* and *Address* from its user profile. When no storage trust is present in both WS and PS, extra attributes are part of the user profiles and it depends on the distribution trust of the user. Table 4.1 presents the extra attributes collected in the web shop's user profile for the alternative services of *WS::Purchase* under different trust perceptions. In the second column, the user has distribution trust in both organizations, while in the third column he only trusts WS. The table clearly shows that, except for service π^{b*}_{WS} that uses the privacy-friendly ID card in the case that $T_D = \{WS, PS\}$, the user is always identifiable by the web shop. These two examples illustrate some of the conclusions generated by the framework. For instance, to remain unidentifiable towards WS, the user needs to trust the storage and distribution policies of both WS and PS.

[1]see https://github.com/decroik/inspect-privacy-and-trust/, 2013

In addition, user profiles are bounded by a minimum and maximum. For the user profile $P_{o'}^{id}(G|\pi_o^*)$ we get that $P_{o'}^{id}(G|\pi_o^*)^{\mathfrak{T}_{max}} \subseteq P_{o'}^{id}(G|\pi_o^*) \subseteq P_{o'}^{id}(G|\pi_o^*)^{\mathfrak{T}_{min}}$. The minimum user profile correspond to a full user trust perception $\mathfrak{T}_{max} = \{T_S^{max}, T_D^{max}\}$, where $\forall o' \in O : o' \in T_S^{max} \wedge o' \in T_D^{max}$. The maximum user profile instead, corresponds to a user who has no trust $\mathfrak{T}_{min} = \{T_S^{min}, T_D^{min}\}$, with $T_S^{min} = \emptyset$ and $T_D^{min} = \emptyset$. It is trivial to prove this order between user profiles using Equations (4.1) and (4.2). This requires to define the order relation between two trust perceptions $\mathfrak{T}_u = \{T_S^u, T_D^u\}$ and $\mathfrak{T}_v = \{T_S^v, T_D^v\}$, namely $\mathfrak{T}_u \geq \mathfrak{T}_v$, with $\mathfrak{T}_u \geq \mathfrak{T}_v \Leftrightarrow T_S^v \subseteq T_S^u \wedge T_D^v \subseteq T_D^u$.

Flexibility. Different *types of users* are easily modeled based on the user's initial state (e.g. the credentials the user owns) and his trust perception. Modelers can use this to estimate the acceptance of a system for different types of real-world users, since their perceptions influence their attitude towards the attributes they disclose [192, 77]. Furthermore, our framework supports a less strict definition of *identifiability* and *pseudonymity* represented in the identifier model $\mathcal{M}_{ID}$, which may be closer to their real-world counterparts.

Basic Framework. The approach presented here, is a basic framework for the analysis of the privacy of users across different organizations. This framework already allows to analyze many real-world scenarios and offers interesting conclusions. Nevertheless, the framework can easily be extended to include many more features, and coarse-grained concepts proposed here can be further refined. Extra inference rules R_I can be added to the theory R to extract more conclusions. For instance, inference rules can be defined to show the gradual growth of the user's profiles over time for each additional invocation of a service. Other inference rules may be used to verify the correctness (consistency) of the input model or show the resulting user profiles when organizations collude. Extensions to the vocabulary $\mathcal{E}$ may be necessary when defining new conclusions and adding new inference rules to the framework. However, both inference rules and vocabularies can be part of a library that is available to modelers. Currently, the user's trust perception is rather coarse-grained based on storage and distribution trust. Although it may express many real-world settings, the framework can easily be extended with more fine-grained trust perceptions.

Usability. Current service policies often vaguely describe their data practices. It is, therefore, not trivial to use them as an input to automatically generate the system model. Specification languages [22, 43, 182] present proposals that enforce organizations to act according to the service policies that accurately describe their data practices. These formal representations can be used to automatically derive the system model in our framework. To make the system modeling more comprehensive for non-experts, an input and output component can be added to the framework. For instance, the input component can automatically generate a user model $\mathcal{M}_U$ and identifier model $\mathcal{M}_{ID}$

from a graphical representation made by the modeler. Similarly, a graphical representation of the conclusions can be created to ease the interpretation of the generated output.

4.6 Conclusions

The present chapter presented a formal approach to analyze privacy properties in advanced electronic services. A framework realized in IDP is designed, comprising an input model and an inference strategy. The latter supports the compilation of user profiles from policies that are defined by service providers. Relevant privacy properties can be extracted from these profiles. Among others, it allows to check whether the actual system behavior corresponds to the user's expectations about the provider's data practices. To show the framework's flexibility, it is realized and applied to a web shop. This chapter also described the inference strategy in more detail, and applies it to a user with certain privacy and trust assumptions. The next chapter will model multiple users and show that the acceptability of the system under study also depends on the user's privacy preferences and trust in other stakeholders.

Chapter 5

Queries and Feedback

An approach to model electronic services with as main goal to analyze the privacy properties is presented in chapter 3 and 4. The former defines the key concepts for representing the privacy-related aspects of composite electronic services; the latter integrates these concepts in a framework that consists of components and relationships between them. Moreover, a strategy to infer privacy properties was described, and the framework was realized in a knowledge base system, namely IDP. The previous chapters also show how relevant privacy feedback can be extracted.

Contributions. It is the present chapter's goal to show the enormous potential of the framework for extracting relevant privacy feedback. Therefore, it is fully devoted to queries and feedback. Queries are classified according to multiple categories. They can be applied to inspect the privacy of advanced composite electronic services. Some queries focus on feedback related to personal information spreading in the system whereas other queries show what data can be linked. Multiple queries are applied to two alternative loyalty schemes. The present chapter shows that the framework supports reasoning about the privacy properties of different schemes. The resulting privacy properties can be compared. This is useful feedback for both end-users and system designers.

The contribution in this chapter are peer-reviewed in [68].

First, two loyalty schemes that are used in the rest of this chapter are described. Section 5.1 gives an overview of the conceptual models. Thereafter, the privacy assessment strategy is explained, and relevant privacy queries are defined (see section 5.2). Relevant privacy properties of both loyalty schemes are then evaluated and compared in Section 5.3. This chapter ends with an evaluation of the approach (see 5.4) and conclusions (see 5.5).

5.1 Loyalty Schemes

To demonstrate the power of the modeling approach, the methodology is applied to two variants of a loyalty system. This case study targets the validation of the approach and aims at demonstrating its practicability. An in-depth privacy assessment will be performed, and both variants will be compared based on key privacy properties. In a loyalty system, customers can collect *loyalty points*. If a threshold is exceeded, the customer can get a reduction after which the *loyalty point balance* is decreased with a given value. This section gives an overview of the functional behavior of both variants.

Loyalty scheme 1. In the first scheme, each customer owns a *plastic loyalty card* with a QR code printed on it. The latter embeds a *customer identifier* (i.e. a unique pseudonym) and is issued during registration. During this phase, the customer releases personal information to the loyalty provider (LP), including his *address*, *e-mail address*, *phone number*, and possibly a collection of other *personal properties*. After registration, the card can be used at a grocery store (GS). When the customer checks out, the cashier scans all *products*, and subsequently the *customer identifier* on the plastic card. All these data are sent to the loyalty provider, which calculates the number of *loyalty points* that are collected. Afterwards, LP sends back the *increment*, together with the customer's *new balance*, to the grocery store. These data are then printed on the customer's ticket. The loyalty provider forwards the list of products in each shopping cart to an advertiser (AD). The advertiser pays a monthly fee to get the data, and can build anonymous profiles based on the retrieved information. The data can be processed to increase the attractiveness of product offerings at certain grocery stores.

Loyalty scheme 2. In the second variant, the plastic card is replaced by a smartphone app that is provided by a *loyalty app provider* (LA), and can be used to manage multiple loyalty cards in a virtual card wallet. When using it the first time, users must release their *name* and *e-mail address* during the registration phase. Each virtual card contains a *customer identifier* (i.e. a unique pseudonym) stored in the context of the app. It can be displayed as a QR code by the app and subsequently presented to the grocery store (GS). When the app is installed, the consumer gets a notification of the app's permissions. It mentions that the app provider can collect the user's *e-mail address*, his *location* and a *fingerprint* of the mobile device when the app is running. The app policy further defines that these data can be forwarded to an advertiser (AD). A user typically starts the loyalty app when checking out. He then selects the right virtual card, and presents the right QR code to the scanner. The rest is similar to variant 1.

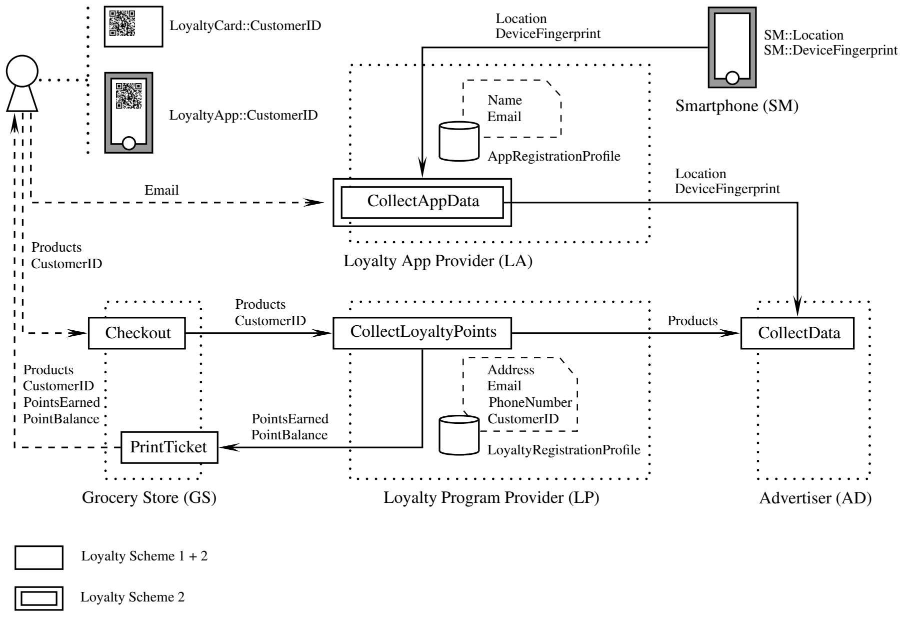

Figure 5.1: Conceptual model of loyalty schemes 1 and 2.

Conceptual model. Figure 5.1 displays a graphical overview of the data flow in both variants of the loyalty system. It shows that the user can either use a plastic loyalty card or a mobile app to identify him to the grocery store. The model also includes the set of stakeholders (i.e. User, *GS*, *LP*, *AD* and *LA*) and the set of services that are assigned to each service provider. Moreover, it defines the order in which services are invoked, and the access policies that are assigned to each service. The latter is specified in the privacy policies that are made publicly available by the service providers. This can either be in a P3P policy file for traditional web services or as a list of permissions that is displayed to the user when a mobile app is installed. Two actions (or services) are initiated by the customer (or by his mobile device), namely *CheckOut* and *CollectAppData*. The former is represented by node π^{c*}_{GS} in the service invocation graph and refers to releasing the *customer identifier* embedded in the QR code to the scanner when checking out at the grocery store. The latter is represented by node π^{s*}_{LA} in the service invocation graph and defines the data that is released by the smartphone to the loyalty app provider when the app is running. Node π^{s*}_{LA} is only relevant for variant 2. Figures 5.2 and 5.3 derive the service invocation graphs for variants 1 and 2 respectively. It should be noted that that services that are marked with an asterix are invoked by the customer (or his smartphone).

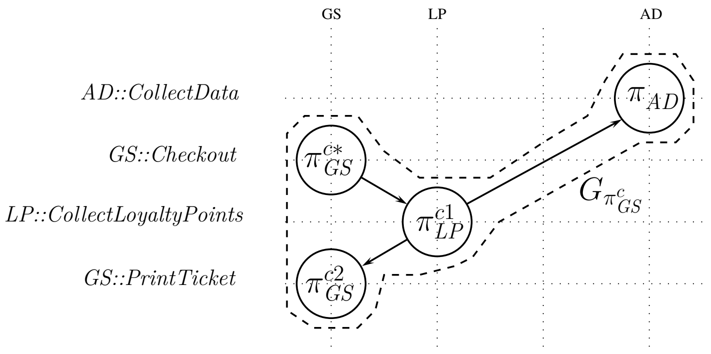

Figure 5.2: Service invocation graph of loyalty scheme 1.

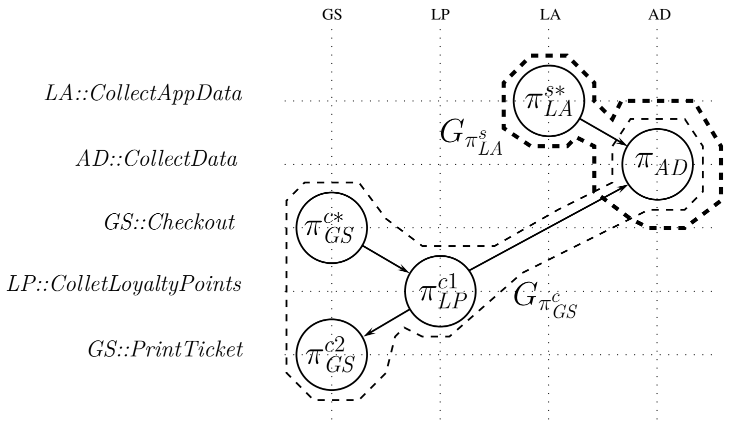

Figure 5.3: Service invocation graph of loyalty scheme 2.

5.2 Privacy Assessment Strategy

The privacy assessment inspects and queries the profiles that can be built by the stakeholders in the loyalty schemes. The profiles under study are *expected profiles*. This means that they are variable under different user trust perceptions. For instance, a user may highly trust a specific service provider. Amongst others, this means that he believes that the service provider will comply with the storage policy that is specified and made publicly available. Only the attributes that are included in the storage policy will be stored even if a user releases more personal data during a service transaction. For instance, if a user needs to release his birth date (to prove to be older than 18 years old) to access a service and the policy does not state that the age will be stored, the age will not be included in the (expected) profile. On the contrary, a user may have low trust in a service provider, and believes that all personal attributes that are released, will be stored (even though specified differently in the privacy policy). Besides the trust assumptions, the user's notion of identifiability and anonymity will also have an impact on the outcome of the queries. For instance, one user can assume that the *birth date* and *city* make him identifiable, whereas another user can assume that, besides the *birth date*, his full *address* is needed to make him uniquely identifiable. In this chapter, we model two prototypical types of users, and assess (and compare) the privacy properties by querying the expected profiles under different trust and identifiability assumptions. Both loyalty schemes are evaluated. It is worth mentioning that our approach is flexible to model many different types of users. This is exactly one of the key properties of our methodology. The rest of this section is structured as follows. First, user profiles are formally defined. Thereafter, the queries that will be performed on both loyalty schemes are classified. Finally, two prototypical users (or consumers) are instantiated. The next section returns query results on both schemes given the trust and identifiability assumptions of these consumers.

User profiles. The tuple (o, id, s) defines a profile that is kept by organization $o \in O$ and that contains the unique identifier $id \in ID$ (i.e. a unique pseudonym or a set of uniquely identifying attributes). The profile (o, id, s) is created as soon as the user invokes service $s \in S$. $UserProfile(o, id, s, a)$ defines that attribute $a \in A$ is included in the profile (o, id, s). For instance, $UserProfile(LP, CustomerID, Checkout, Address)$ defines that the profile $(LP, CustomerID, Checkout)$ also contains the customer's *Address*.

Querying privacy properties. Multiple types of queries can be performed on the (expected) user profiles. They return useful information related to the privacy properties of the overall system. The queries can be classified according to four categories. Table B.1 in Appendix B formally defines the queries:

- **Querying the anonymity level (i.e. *Q1*).** This class defines the anonymity level of a prototypical user towards each organization in the system under study after he invoked a specific service. This anonymity level is derived from the (expected) information that is included in the profiles. Three anonymity levels are defined. A user is *identifiable* towards an organization if the profile that can be compiled by that organization contains identifiable information that is sufficient to point to a physical individual. A user remains *pseudonymous* towards an organization if a persistent unique pseudonym is part of the profile. However, the pseudonym cannot be linked to an individual. A user is *anonymous* towards an organization if no unique data is revealed to that organization when a particular service is accessed.

- **Querying the attributes that are released to organizations (i.e. $Q2$).** A straightforward query can return all attributes that are kept in the profile by an organization after a user invoked a service. However, this set of attributes can be large. Therefore, two variants *Q2.1* and *Q2.2* are presented here. *Q2.1* queries all organizations in the system that store a given set of attributes *SearchSet* after a certain service invocation. For instance, the *SearchSet* can be $\{$*Address*, *Products*, *Name*, *CustomerID*$\}$. If so, the organizations are returned that store at least those four attributes about the user in a profile. *Q2.2* allows to check if certain privacy requirements are met in a given system. More specifically, the designer or audit instance can impose the requirement that certain organizations should not be able to store certain sensitive information. For instance, personal medical records may certainly not be stored and processed by a commercial advertiser in an e-health system. Similarly, secret banking information may not be stored by the web shop in an e-commerce application. In the latter case, only the bank institution may store bank account information. Query *Q2.2* allows to input a set of attributes that may not be revealed towards certain organizations. Subsequently, the query returns the service requests that may violate those requirements, together with more information about the exact requirement that is violated. In the loyalty system, it might be useful to impose the requirement that the advertiser (AD) may not collect a user's *Name*, *Address* or *Email*, and then check whether this is fulfilled when alternative services are invoked.

- **Querying the impact of underlying collaborations (i.e. $Q3$).** In some cases, information exchange between organizations is important to guarantee a quality of service in composite web services. For instance, a web shop may release the customer's address to a delivery service provider. Without that information, the latter can never deliver a packet to the customer. However, in other cases, information exchange is not functional, but rather a means to increase profit. It is noteworthy to signal that these types of collaborations are often not specified in the privacy policies. For instance, a commercial organization, such as the loyalty program provider, may make a contract with an advertiser. The contract can define that the commercial organization releases certain attributes or even full profiles to the advertiser. In return, the former receives a monthly fee. In many cases, these types of collaborations lead to unidirectional information exchanges for which an organization receives money. In our methodology, $\mathit{ShareDataWith}(o_{\mathit{from}}, o_{\mathit{to}})$ defines that organization o_{from} releases the collected user profiles to organization o_{to} (probably in return for a compensation). This predicate can be used in two types of queries *Q3.1* and *Q3.2*.

Q3.1 analyzes the new links that are introduced if a new collaboration takes place. For instance, we can query the new links and profiles if the loyalty provider (LP) shares all collected profiles with the grocery store (GS). In order to investigate that impact, the user adds a predicate $\mathit{ShareDataWith}(LP, GS)$ to the model, and thereafter applies query *Q1* (see Table B.1 in Appendix B). Next, *Q2.1* returns the attributes in the profiles.

Q3.2 returns a set of collaborations between organizations that are necessary to fulfill more advanced requirements. It should be noted that, in some cases, no valid set can be returned. Four types of advanced requirements can be imposed. We can either impose that users are *Identifiable*, *Pseudonymous* or *Anonymous* towards certain organizations after certain service invocations are completed. If a user must be *Unidentifiable*, this means that he remains either *Pseudonymous* or *Anonymous* towards a certain organization. We can further impose that certain collaborations must occur and that others are forbidden. We refer to query *Q3.2* in Table B.1 in Appendix B for a concrete instantiation of the query. Amongst others, the set of additional requirements imposes that the user must be identifiable towards the grocery store after *Checkout*, and at the same time remain unidentifiable to the loyalty app provider. Moreover, LP must release all the collected data to GS. At the same time, some collaborations are not allowed. Similar to the first example, this type of query can be solved by reusing queries *Q1*, *Q2.1* and *Q2.2*.

- **Querying trust that is required between organizations (i.e. $Q4$).** If multiple entities are involved in a composite web service, certain organizations must trust others. This is typically the case for attributes that are exchanged between organizations. An organization Y that receives user attributes from another organization X must rely on X when it comes to the correctness of the user attributes. Moreover, in composite web services, certain user attributes are passed through a chain of organizations. In that case, the final recipient must trust all entities in the chain. Query $Q4.1$ focuses on these trust relations. It searches for the organizations that must be trusted by a given set of organizations *TrustRelationshipsOf*. These trust relations are represented by *Trusts*(o_{ref}, e) that expresses that o_{ref} *trusts* stakeholder e (i.e. the user or an organization). This relation is built from the data flow of attributes *AttributeDataFlow*. This is constructed by the inductive definition *r4.2* (see Table B.1 in Appendix B), and can be derived from the service invocation graph. Although not illustrated in the presented loyalty schemes, our methodology also supports returning trust relations that are required between organizations in case attributes are certified (or signed) by an issuing organization O. In that case, the receiving organization Y only needs to trust O, and not all organizations X' in a chain that possibly forwards the data towards the final receiver. The outcome of these queries is independent of the user's trust assumptions.

Two prototypical consumers. The queries are performed on two prototypical consumers in the next section. Both consumers have different trust perceptions and identifiability assumptions. Lisa is modeled as a consumer who is rather cautious about her privacy, and Homer as a more careless character. It should be noted that the methodology should be flexible enough to model a wide range of user types. Both characters are discussed in more detail below. An overview of their identifiability and trust assumptions is also listed in Table 5.1:

- **Homer.** He trusts nearly all entities in the loyalty system. This means that he expects that most entities will comply with their privacy policies. More specifically, he expects that the grocery store, the loyalty app provider, and the loyalty program provider will only store the attributes that are specified in their policy, and that they will only forward a selection of those attributes to other entities as specified in the policy. In fact, the advertiser is the only entity that is not trusted by Homer. Homer further assumes he becomes identifiable if both this *Name* and *Address* (i.e. I_1) are included in a profile. A set of other personal attributes – like his *e-mail address* (i.e. N_2), his *customer identifier* (i.e. N_3) and *device fingerprint* (i.e. N_4) – are considered pseudonyms.

- **Lisa.** She worries more about her privacy. In contrast to Homer, she only trusts the grocery store. She assumes that the loyalty app provider, the loyalty program provider and the advertiser do not always comply with their specified privacy policy. Moreover, she assumes that either revealing her *Name* (i.e. I_2) or *Address* (i.e. I_3) makes her identifiable towards an organization. Just like Homer, she marks a set of attributes as pseudonymous.

Table 5.1: User models of a less privacy aware user *Homer* and a more privacy aware user *Lisa*, applied to the modeled services of the loyalty schemes in Section 5.1.

	Homer	*Lisa*
Trust Perception		
Storage Trust	*GS,LA,LP*	*GS*
Distribution Trust	*GS,LA,LP*	*GS*
Identifiability model *(Identifiers)*		
Identity	$I_1 = \{Name, Address\}$	$I_2 = \{Name\}$, $I_3 = \{Address\}$
Pseudonym	$N_1 = \{PhoneNumber\}$, $N_2 = \{Email\}$, $N_3 = \{CustomerID\}$, $N_4 = \{DeviceFingerprint\}$	$N_1 = \{PhoneNumber\}$, $N_2 = \{Email\}$, $N_3 = \{CustomerID\}$, $N_4 = \{DeviceFingerprint\}$

It is noteworthy to signal that not all these user assumptions should be modeled each time a new system is analyzed. In fact, prototypical and realistic identifiability models can be reused when other systems are modeled. For a specific system, the user must mark the level of trust he has in all stakeholders, although experience teaches us that this is easier by far than reasoning about realistic identifiability assumptions.

5.3 Privacy Assessment Results

The present section discusses the results of the queries applied to the model of two variants of the loyalty scheme. These queries give feedback about the privacy level and perception of two prototypical consumers, namely Homer and Lisa, and mainly extract the feedback from the service invocation graphs and the compiled profiles. The list of queries – as implemented in IDP – can be found at *https://github.com/decroik/inspect-privacy-and-trust.* The first scheme inspects the impact on the privacy of a consumer when checking out (i.e. *Checkout*) with a plastic loyalty card at the grocery store. The second variant performs a similar evaluation. The impact of running the mobile app (i.e. *CollectAppData*) and checking out (i.e. *Checkout*) with the virtual card are both considered here in the privacy analysis.

Table 5.2 and 5.3 compare the level of anonymity that Lisa and Homer have in both schemes given the trust perception and identifiability assumptions of them. Therefore, query *Q1* is performed. The results show that, when checking out, Homer is persistently pseudonymous towards the loyalty program provider and remains anonymous towards all other organizations in both variants of the ecosystem. However, in variant 2, the mobile app makes Homer pseudonymous towards the loyalty app provider and advertiser. In contrast, Lisa becomes identifiable towards the loyalty program provider and the advertiser when checking out with the plastic card, whereas she remains anonymous towards the grocery store. Running the mobile app and checking out makes her identifiable towards the loyalty program and app provider, and to the advertiser. Hence, users who define more strict privacy concerns will probably prefer the plastic card compared to the mobile app variant.

Table 5.2: Outcome of query *Q1* applied to variant 1. Three levels of anonymity are defined: (I)dentifiability, (P)seudonymity, or (A)nonymity.

	Checkout	
	Homer	*Lisa*
GS	A	A
LP	P	I
AD	A	I

Table 5.3: Outcome of query *Q1* applied to variant 2. Three levels of anonymity are defined: (I)dentifiability, (P)seudonymity, or (A)nonymity.

	CollectAppData		*Checkout*	
	Homer	*Lisa*	*Homer*	*Lisa*
GS	A	A	A	A
LP	A	A	P	I
AD	P	I	A	I
LA	P	I	A	A

Table 5.4 and 5.5 return results related to queries *Q2.1* and *Q2.2* respectively. They analyze the attributes that are kept in profiles. Table 5.4 shows that no organization except the advertiser has the set $\{Address, Products, Name, CustomerID\}$ in a profile. This is only valid for Lisa's profile caused by her belief that much more information is stored and distributed by the stakeholders in the ecosystem than actually presented in their privacy policy. Moreover, this is only valid for variant 2 of the loyalty system. Hence, more privacy aware consumers may opt for using a plastic card.

Table 5.4: Outcome of query *Q2.1*. The table shows the consumer's user profiles containing $\{Address, Products, Name, CustomerID\}$.

	Homer	*Lisa*
Variant 1		
	–	–
Variant 2		
		$(AD, N_1, Checkout)$
		$(AD, N_2, Checkout)$
		$(AD, N_3, Checkout)$
		$(AD, N_4, Checkout)$
		$(AD, I_2, Checkout)$
		$(AD, I_3, Checkout)$

Table 5.5: Outcome of query *Q2.2*. It shows whether the attributes *(1) name*, *(2) address*, and *(3) e-mail address* are stored by the advertiser when invoking a specific service.

	Homer	*Lisa*		
Rule		(1)	(2)	(3)
Variant 1				
Checkout			✓	✓
Variant 2				
CollectAppData		✓		✓
Checkout		✓	✓	✓

As advertisers are typically highly untrusted by many consumers – even Homer may have his doubt about the trustworthiness of an advertiser – it might be useful to inspect whether certain attributes are kept in their profiles about consumers. Consumers want to inspect if their *Name*, *Address* or *Email* is exposed to them. Releasing the first two attributes allows advertisers to overload the user's postal box with publicity; the latter can lead to spam. Inspecting the presence or absence of these attributes is exactly what is performed by query *Q2.2*. The results are listed in Table 5.5.

Queries *Q3.1* and *Q3.2* focus on the impact of collaborations between organizations on the user's level of anonymity. Query *Q3.2* starts from a set of identifiability/anonymity requirements Req_{Ident} and collaboration constraints Req_{Coll}, and returns what additional collaborations are at least required to

Table 5.6: Results of query *Q3.2*. Additional collaborations that are necessary to fulfill requirements $Req_{Ident} \cup Req_{Coll}$ (see Table B.1 in Appendix B) for the second variant.

	Homer				*Lisa*			
to	*GS*	*LA*	*LP*	*AD*	*GS*	*LA*	*LP*	*AD*
from								
GS								
LA				✓				
LP	✓			✓	✓			
AD	✓	✓				✓		

fulfill Req_{Ident}. Our query imposes two identifiability requirements Req_{Ident}, namely (a) the grocery store (*GS*) wants to identify consumers at checkout and (b) consumers are not anonymous towards the loyalty app provider (*LA*) during checkout when the loyalty app is running. Furthermore, we assume that *LP* must collaborate with *GS*, and a set of other collaborations are not allowed. For instance, neither *GS* nor *LP* may collaborate with the loyalty app provider (*LA*).

The query returns if all these requirements can be satisfied, and if so, evaluates which collaborations (i.e. data sharing between organizations) are necessary to fulfill all these requirements. Table 5.6 shows the results after being applied to the second variant of the loyalty scheme. From the perspective of *Homer*, at least five collaborations are required. In contrast to *Lisa*'s perspective, two collaborations may meet the requirements as returned in the table.

To analyze the impact of the new collaborations on the anonymity level of *Homer* and *Lisa*, query *Q3.1* is applied on the model (including the new additional data sharing predicates). Table 5.7 shows the anonymity level in scheme 2 of *Homer*'s and *Lisa*'s. It is trivial that the overall level of anonymity decreases compared to the situation in which less data sharing was introduced. At *Checkout*, *Homer* is now pseudonymous towards the loyalty program provider and identifiable towards the grocery store, loyalty app provider, and advertisers. *Lisa* becomes identifiable to all organizations. The fact that customers are identifiable towards advertisers at *Checkout* is unacceptable for *Lisa*. If query *Q3.2* is applied with the additional constraint that it is not permitted to be identifiable at *Checkout* towards advertisers, we see that no solution exists for *Lisa*. This result is expected because Table 5.3 shows that even without collaborations she is already identifiable.

Queries can check if these new collaborations allow organizations to link

Table 5.7: Results of query *Q3.1* that return the consequences of the collaborations of Table 5.6 on the consumer's anonymity level. Three levels of anonymity are defined: (I)dentifiability, (P)seudonymity, or (A)nonymity.

	CollectAppData		*Checkout*	
	Homer	*Lisa*	*Homer*	*Lisa*
GS	P	A	I	I
LP	A	A	P	I
AD	P	I	I	I
LA	P	I	I	I

Table 5.8: Results of query *Q2.1* to analyze the attribute based consequences in variant 2 of the collaborations of Table 5.6. User profiles *(Organization, Identifier, Service)* containing set $\{Address, Products, Name, CustomerID\}$ are marked (✓).

	Homer				*Lisa*			
	GS	*LA*	*LP*	*AD*	*GS*	*LA*	*LP*	*AD*
CollectAppData								
	–	–	–	–	–	–	–	–
Checkout								
I_1	✓	✓		✓				
I_2						✓		✓
I_3						✓		✓
N_1	✓	✓		✓		✓		✓
N_2	✓	✓		✓		✓		✓
N_3	✓	✓		✓		✓		✓
N_4	✓	✓		✓		✓		✓

Homer's and *Lisa*'s purchased *products* with their *name*, *address* and *customer id*. The results of *Q2.1*, presented in Table 5.8, show that from *Homer*'s perspective the grocery store, the loyalty app provider, and advertisers are able to link those user attributes. From *Lisa*'s perspective, only the loyalty app provider and advertiser can do this. Furthermore, query *Q2.2*, of which the results are presented in Table 5.9, shows that advertisers can retrieve the *name*, *address*, or *e-mail address* when checking out at the grocery store in the second loyalty scheme.

Finally, query *Q4.1* returns the required trust relationships between the

Table 5.9: Results of query *Q2.2* to detect violations in variant 2 in case of the collaborations of Table 5.6. Advertisers are not permitted to store the customer's *(1) name*, *(2) address*, or *(3) e-mail address.* User profiles *(AD, Identifier, Service)* that violate this, are marked (✓).

	Homer			*Lisa*		
Rule	(1)	(2)	(3)	(1)	(2)	(3)
CollectAppData						
I_1						
I_2				✓		✓
I_3						
N_1						
N_2	✓		✓	✓		✓
N_3						
N_4	✓		✓	✓		✓
Checkout						
I_1	✓	✓	✓			
I_2				✓	✓	✓
I_3				✓	✓	✓
N_1	✓	✓	✓	✓	✓	✓
N_2	✓	✓	✓	✓	✓	✓
N_3	✓	✓	✓	✓	✓	✓
N_4	✓	✓	✓	✓	✓	✓

Table 5.10: Results of query *Q4.1* to analyze trust relationships in scheme 1 for *Homer* and *Lisa*. Organizations that are trusted by another are marked (✓).

Trusted	*User*	*GS*	*LP*	*AD*
By				
GS				
LP	✓	✓		
AD				

stakeholders for both variants of the loyalty system. The results, presented in Table 5.10, show that for both variants the loyalty provider must trust the user and grocery store. This result is derived from the invocation graphs, and is independent from the user model applied.

5.4 Evaluation

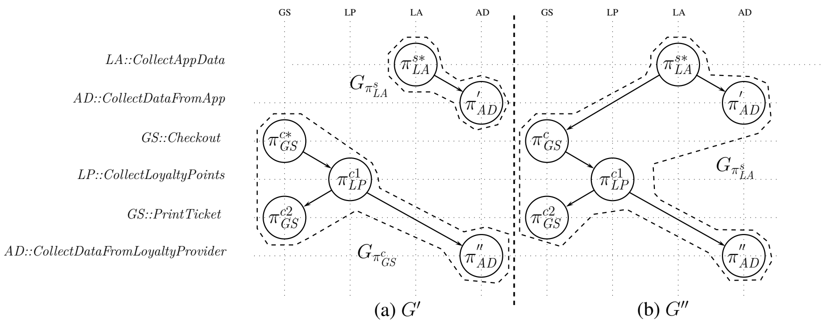

Figure 5.4: Alternative service invocation graphs of loyalty scheme 2.

This section reflects on the usability and flexibility of the modeling approach, and proposes alternative service representations. Model extensions are mentioned at the end of this section.

The privacy analysis methodology allows to extract meaningful information towards *system designers*, *audit instances* and *end-users*. These are the major three stakeholders that can benefit from the formal methodology.

System designers can adopt the approach for many reasons. First, they can extract whether the current design is compatible with corporate privacy policies and legislation, and eventually perform modifications that may lead to compliance. Second, the methodology also returns relevant information about the identifiability level and size of the profiles kept by certain service providers. This gives feedback about possible additional security measures that should be taken. If profiles are identifiable, they are probably a more interesting target for hackers. In contrast, anonymous profiles have less value and, consequently, will be attacked with a much lower probability. Third, system designers can also query the impact of alternative design decisions. For example, the impact of using alternative authentication tokens on the privacy can be returned. More specifically, although it is already clear that anonymous credentials are in many cases more privacy friendly than traditional certificate technology, this methodology can show the exact profiles in both cases. Hence, a more fine-grained comparison is possible. If the privacy gain by using anonymous credentials is meaningless or low, the designer might opt for more straightforward authentication technologies (such as X.509 certificates). However, if the privacy increases considerably when using anonymous credentials, this solution might be considered (although this technology often introduces some performance penalty). Finally, prototypical users can be modeled. From this, the designer can derive which types of characters will typically use the system, and which ones will be reluctant to use it. This can give indications about the market share of a typical system. For instance, if a prototypical user is modeled that represents the trust perception and privacy wishes of 25 percent of the population, and if serious conflicts arise between the trust required to meet the privacy wishes and the trust perception of that prototypical user, that system will probably hardly be adopted by those individuals.

Audit organizations are possibly the major adopters of this methodology. Those instances can verify whether systems under study are compatible with privacy legislation, and derive the required trust in the organizations for realizing certain privacy claims (related to data storage, processing and distribution). Audit organizations will typically also be interested in the impact of collaborations on the overall privacy.

Finally, the *end-users* can compare the privacy-friendliness of systems with similar functionality, and select one based on the privacy analysis that is possibly performed and published by an audit organization. For instance, if multiple web shops offer the same book, the user can select a particular shop not only based on the price, but also based on the privacy properties when using the service. If multiple credentials, such as a Google account, a student card or an electronic identity card, are supported for accessing/using the service, the user can now select the most appropriate token based on the privacy properties. Although the user cannot model each system it accesses, model descriptions of multiple electronic services can be published by an audit instance. Instead of moving a meaningless privacy slider that is often used in current web browsers, the user can set his trust perceptions and identifiability assumptions. This only needs to be done once. These inputs can serve as input to the models that are published by audit instances. It is worth mentioning that one could argue that the user needs to input his trust perception in each individual entity in the system. However, categories can be proposed, and the input tool can already propose many suggestions. For instance, users typically have lower trust in commercial organizations than in governmental ones. Similarly, users are willing to release bank account information to financial institutions whereas they are willing to release e-health attributes to health providers (and not to their bank institution).

It is worth mentioning that this methodology models the privacy properties of one single individual. Hence, it can provide feedback about user's privacy in terms of *(un)linkability* (i.e. anonymity, pseudonymity, and identifiability) when using electronic services as defined by Phitzmann et al. [143]. Complementary strategies are needed to evaluate *undetectability* and *unobservability* (i.e. two privacy concepts also defined by Phitzman). The user profiles and the attributes they contain, generated by our approach, can serve as input for other types of analysis, for instance approaches based on anonymity sets [156].

Representations of service invocations in scheme 2. In some cases, multiple variations are possible to model a single application. We discuss two alternatives for the service invocation graph of scheme 2. The first one, depicted in Figure 5.4a, splits node $\pi_{AD} \in \Pi$ into two nodes, namely $\pi_{AD}^{'} \in \Pi'$ and $\pi_{AD}^{''} \in \Pi'$. This is based on the property that nodes with multiple incoming edges can be split into nodes with one incoming edge that represent the same service as the original one. However, this comes at a price, namely, the required memory for inferring the service model increases with the number of nodes. Hence, unless required by semantics, nodes should remain undivided. Nodes should be split only when they meet one of the following conditions:

- A node in a sub-graph has at least two incoming edges from nodes in the same sub-graph (i.e. a sub-graph must be a tree).
- The access policies related to each incoming edge of a node are different. For instance, a web e-mail provider's service policy specifies that user information is logged (i.e. logging service) during registration (i.e. registration service) and when the e-mail service is used (i.e. send e-mail service). During registration the user's name and e-mail address are logged, while only the e-mail address and time information is collected when the user sends an e-mail.

In case of the second scheme, the service π_{AD} collects data from π_{LA}^{s*} and π_{GS}^{c3*}. However, it is typical that these service policies are vague and only mention they forward data to *an advertiser* without mentioning the exact advertiser. In the model, such services are connected with empty service policies (service policies are not available to the modeler). Furthermore, it requires to model that a user has no trust in organizations that host these services. They assume that such organizations store and forward all the data they can collect.

The second alternative, depicted in Figure 5.4b, concerns the checkout at the grocery store as a causality of selecting the loyalty card in the loyalty app, namely, $\pi_{LA}^{s*} <_G \pi_{GS}^{c}$ (note that $\pi_{GS}^{c} \notin \Pi^*$). It meets the assumption that a customer selects his loyalty card only to checkout at the grocery store. Because π_{LA}^{s*} and π_{AD}^{c3} are both part of $G_{\pi_{LA}^{s}}$, π_{AD} must be split into $\pi_{AD}^{'}$ and $\pi_{AD}^{''}$. This requires additional memory for inferring the model. Moreover, complications arise with modeling the scanning of the QR code from the loyalty app during the checkout phase at the grocery store. This is modeled as a QR code that is forwarded from π_{LA}^{s*} to π_{GS}^{c}. Nevertheless, it deviates from reality and leads to incorrect results. In reality, it is physically only possible to scan a customer's

QR code with the scanner. In the model however, forwarded data depends on the customer's trust perception. In case it is modeled that he has no trust in the data forwarding practices of an organization, then all data possibly collected by that organization is forwarded. For instance, the loyalty app provider collects and forwards not only the QR code, he also collects other personal data, such as the customer's e-mail address, his location, and linkable data specific to his mobile device (e.g. a unique mobile device ID) or mobile operator (e.g. carrier user ID). Currently, this cannot be expressed and a more fine-grained user trust perception model is required to accurately model this loyalty scheme.

Temporal privacy analysis. In the current model, privacy is analyzed based on user profiles generated from a given state. However, more useful queries are possible when temporal aspects are added to models of the current approach. In particular, each service that is invoked is attached to a point in time. User profiles can then be generated depending on the point of time. Also, dependencies between services can be expressed. For instance, collecting loyalty points is only possible after a customer registered to the loyalty program. This extension creates a wide variety of possible privacy analysis. A first example is to infer a sequence of services that is required for invoking a given service while preserving a user's privacy and meeting a set of functional requirements. Another example is to verify the reachability of a user invokable service that is given.

5.5 Conclusions

This chapter shows the potential of the framework for inspecting the privacy in composite electronic services. It focuses on relevant privacy feedback that can be extracted by querying models that are constructed with the framework. We show that multiple users can be represented. The framework can reason about the acceptability of a system with respect to the privacy preferences and trust assumptions of those users. Moreover, the framework allows to model multiple instances of the same application, and subsequently reason and compare those instances. To demonstrate the flexibility of the framework, two loyalty schemes were modeled and queried. Moreover, multiple users are modeled. We show that a user model can be reused if a new application is modeled. Vice versa, users with different privacy and trust expectations can easily be added without impact on the system modeling part.

Part II

Extensions and Validation

Chapter 6

Modeling Ticketing Systems in Public Transport

Part I of this thesis presented a modeling based approach for inspecting privacy in complex electronic services. Chapter 3 identified modeling concepts essential for capturing privacy properties. Chapter 4 defines a framework comprising the previously defined modeling concepts. The potential of the framework for deriving feedback of privacy properties is shown in Chapter 5. The flexibility of the approach is demonstrated by inspecting the privacy in services from different application domains. The framework considers users who interact with a single service front-end that automatically invokes underlying sub-services.

Contributions. This chapter extends the framework of Part I to analyze the privacy of a user who interacts with multiple service front-ends. Additional modeling concepts are defined to capture advanced credential technologies and relate credential ownership and service invocations to time. A privacy-preserving public transport ticketing system is presented and modeled in IDP for validating the framework. Queries are defined that search for a sequence of executable services that represents the order in which a user interacts with the service front-ends . Time-dependent user profiles, from which feedback about privacy properties are extracted, can be built based on these extensions.

The privacy-preserving public transport ticketing system is joint work with Milica Milutinovic and is peer-reviewed in [129]. The contributions to this paper include the research of state-of-the-art public transport ticketing system technologies, assistance with the protocol specifications, and the performance measurements of the protocols.

The remainder of this chapter is structured as follows. Section 6.1 categorizes credential technologies and extends the privacy analysis framework of Part I with newly defined concepts capturing the properties of alternative credential technologies. Next, modeling concepts capturing temporal properties related to service invocations are added to the framework (see Section 6.2). Section 6.3 defines the modeling concepts necessary to represent conditional service accesses. The inference strategy to compute the user profiles associated with a sequence of services invoked by a user is discussed in Section 6.4. Section 6.5 presents a privacy-preserving public transport ticketing system that is used for the validation of the privacy analysis approach and extensions. This chapter ends with conclusions in Section 6.6.

6.1 Supporting Alternative Credential Technologies

6.1.1 Motivation

The framework presented in Part I of this thesis supports a limited set of credential technologies. It is important to show the extendability of the framework since other credential technologies may have more advanced properties. First, privacy properties are identified that characterize more advanced credential technologies. Modeling extensions are then defined that represent these identified properties.

6.1.2 Privacy Properties of Credential Technologies

Many ticketing services in public transport are based on contactless smart cards. They are issued as personalized or anonymous passes, such as the personalized or anonymous OV-cards [170] in the Dutch public transport system. The personalized version records traveler's personal attributes, while the anonymous card provides better privacy. However, a traveler with an anonymous card has access to a limited set of services. For example, in the Dutch public transport system, a traveler with an anonymous OV-card cannot benefit from reductions based on his age since no personal attributes are stored on his card. The cards are read when a traveler enters the vehicle and in some implementations, such as in the Dutch public transport system, also when he exits. This allows to employ a pay-per-use system, i.e. the traveler pays a fixed price (e.g. the maximum ticket value) when starting a ride and is returned the change when exiting. A commonly used smart card technology for implementing these systems is

Mifare Classic [136]. They are used in the Dutch OV-chip card ticketing system and the London Oyster chip cards [171]. Transactions with this type of cards require mutual authentication between the card and the reader before the card's memory can be accessed. This involves, even in case of an anonymous pass, the exchange of the card's unique identifier, making all transactions with the card linkable. Another example of a commercially used technology is *Calypso* [42]. This is an NFC-enabled smart card technology used in Belgium (MOBIB) [130] and Paris (Navigo) [133]. Similar to Mifare Classic cards, transactions in which a Calypso smart card is used are linkable as well since the card's serial number is disclosed.

A more privacy-preserving public transport ticketing system is presented in Section 6.5. This system relies on *eTokens* [128]. The tokens, denoted by $\{C, info, pbsig\}$, represent a partially blinded signature *pbsig* on a public information part *info* and a private part *C*. It is assumed that the public information part, which is always revealed, leaks no unique attributes that can be linked to the traveler. The private part is a commitment on a secret. The secret is linked to an Idemix credential and is not revealed to public transport operators. A traveler possesses a batch of tokens that can only be spent once. The tokens allow to travel anonymously since their issuance cannot be linked with their spending and the spending of two tokens cannot be linked as well.

Table 6.1 gives an non-exhaustive overview of the privacy properties of advanced credential technologies relevant to public transport ticketing systems. The considered properties are:

- *Linkability/Unlinkability.* A service's access policy may require that individuals prove ownership of a set of credentials (i.e. spending a credential) to access the service. The credentials are issued by a trusted third-party or by the service provider. Unlinkability between credential *issuance* and *spending* (*C1*) and between *two spendings* of a same credential (*C2*) are two essential properties when considering privacy in services.
- *Attribute disclosure.* Data minimization is a main privacy protection principle aiming for minimal disclosure of personal attributes to service providers. For this reason, it is important to consider *selective attribute disclosure* (*C3*) of credential technologies. For example, a user can choose to only reveal his name even though his address and birth date are included in the credential as well. Furthermore, the selective attribute disclosure property also enables to prove attribute properties without revealing their values (i.e. attribute predicate disclosure). For example, an individual proves his age is between 18 and 25, based on his birth date included in his credential, without revealing birth date.

Table 6.1: Overview of privacy properties of credentials

	X.509	Idemix	U-Prove	Mifare Classic	Calypso	eToken
(L)inkability / (U)nlinkability[1]						
(*C1*) Issuance ↔ Spending	L	U	U	L	L	U[2]
(*C2*) Spending ↔ Spending	L	U	U[3]	L	L	U[2,3]
Attribute disclosure						
(*C3*) Selective attribute disclosure	X	✓	✓	X	X	X

[1]: It is assumed that no unique set of attributes are disclosed
[2]: It is assumed that the eToken's public information part *info*, which is always revealed, leaks no unique attributes that can be linked to an individual
[3]: Can only be obtained with different credentials used for each spending

Linkability/Unlinkability. Depending on the used credential technology, unique attributes that can be linked to an individual may be revealed towards the issuer and service provider during the issuance and spending of a credential. During the issuance process, the unique identifiers of X.509 certificates, Mifare Classic and Calypso smart cards are revealed (e.g. an X.509 certificate's serial number and public key, or the card's unique identifier in case of a Mifare Classic and Calypso card). Moreover, since the same identifiers are also disclosed when spending the above-mentioned credentials, their issuance and spending (*C1*) and multiple spendings (*C2*) are linkable.

Better privacy properties can be achieved with PETs such as eTokens, Idemix and U-Prove credentials. During the issuance of Idemix credentials, the user sends, along with the (blinded) attributes, a freshly generated *one-time-used* pseudonym (i.e. commitment on his *secret*) to the issuer. Furthermore, mutually unlinkable *one-time-used* pseudonyms are generated each time the Idemix credential is spent. Given that no unique set of attributes is disclosed during the spending, the issuance of the Idemix credential cannot be linked to the spending (*C1*) and multiple spendings of the same credential cannot be linked (*C2*). An eToken is a hybrid cryptographic scheme based on Idemix credentials, partially blinded signatures and commitments that can only be spent once. Their issuance and spending is unlinkable (*C1*) given that no unique set of attributes is disclosed during the spending. Unlinkable spendings (*C2*) can be realized if for each spending a new eToken is selected from a pool of eTokens. The issuance of U-Prove credentials instead, uses blinded signatures to realize unlinkability between the issuance and spending (*C1*). Similar to eTokens, unlinkable spendings of U-Prove credentials can be realized in case different credentials are selected fro each spending from a pool of credentials (*C2*).

Attribute disclosure. (*C3*) Selective attribute disclosure is supported by Idemix and U-Prove. Moreover, both technologies enable to prove properties based on attributes included in the credentials without revealing their values. In contrast to Idemix and U-Prove, all attributes included in eTokens, X.509 certificates, Mifare Classic and Calypso based smart cards are disclosed.

6.1.3 Modeling Credential Technologies

Credential technologies. In the framework presented in Part I, credentials are represented by elements of the *Credential* domain. Sub-domains of the *Credential* domain are defined for each supported credential technology. For instance, X.509 certificates are represented by elements of the *X.509* domain, which is a sub-domain of the *Credential* domain. Similarly, sub-domains are defined for each credential technology in Table 6.1.

Linkability/Unlinkability. In some PETs, credential spendings with the same credential cannot be linked. The current framework cannot express this since it makes only distinction between *pseudonyms* and *identities* that are represented by the domains *Pseudonym* and *Identity* respectively. Both are sub-domains of the domain *Identifier*. To support this, a new domain and predicate are defined. *TransactionPseudonym* is a sub-domain of *Identifier* representing one-time used pseudonyms (i.e. transaction pseudonyms [143]). The predicate $\mathsf{CredentialIdentifier}(\mathit{Credential}, \mathit{Identifier})$ expresses which credential technology-specific identifiers are revealed when spending a credential. Listing 6.1 illustrates how the newly defined concepts can be used to model an X.509 certificate based identity card and one that is based on Idemix credentials. The X.509 certificate based identity card reveals a set of pseudonyms (i.e. the card serial number and signature, and the owner's pubic key) in case it is spent. The Idemix based identity card only reveals a one-time-used identifier when it is spent.

IDP Listing 6.1: Modeling example of credential technology specific identifiers revealed when spending a credential.

`type` Idemix = $\{IdentityCardIdemix\}$
`type` X509 = $\{IdentityCardX509\}$
`type` TransactionPseudonym = $\{OneTimeUsedPseudonym\}$
`type` Pseudonym = $\{SerialNumber; PublicKey; Signature\}$
`type` Identity = $\{\}$

CredentialIdentifier(Credential, Identifier) = {
$(IdentityCardIdemix, OneTimeUsedPseudonym)$;
$(IdentityCardX509, SerialNumber)$; $(IdentityCard, PublicKey)$;
$(IdentityCard509, Signature)$}

Attribute Disclosure The framework is extended with concepts capable of expressing the attribute disclosure properties (*C3*) of the credential technologies considered in Table 6.1. The current framework supports expressing the non-selective and selective disclosure of attribute values, but lacks expressiveness to capture *attribute predicate disclosure*. Attributes can be considered as properties describing the user (e.g. his eye color) and its context (e.g. his location). Moreover, properties can be based on combinations of attributes. For this reason, the framework is extended with the PropertyBasedOn(*Attribute*, *Attribute*) predicate. This predicate relates a property (i.e. first argument) to the attributes from which it is derived (i.e. second argument). Listing 6.2 illustrates how the property $18 < Age < 25$ can be expressed in terms of his *birth date*. For example, an individual proves his age is between 18 and 25 using the birth date that is included in his credential. In case, for example, an Idemix credential is used, this can be proven while keeping the birth date hidden. In case an X.509 certificate is used, then besides his birth date all other attribute values in the credential are revealed as well. Additional rules are defined in the framework's theory describing the selective attribute disclosure.

IDP Listing 6.2: Modeling example of the property $18 < \text{Age} < 25$.

```
type Attribute = {AgeBetween18and25; BirthDate}

PropertyBasedOn(Attribute, Attribute) = {
        (AgeBetween18and25, BirthDate)}
```

6.1.4 Reflection

The modeling concepts of the framework in Part I (see Listing 4.4) can be reused to represent the advanced credential technologies in Table 6.1. However, they lack the expressiveness to fully capture the properties of these credential technologies. For this reason, additional modeling concepts were defined to express advanced properties such as selective attribute disclosure and the disclosure of unique sets of attributes (i.e. pseudonyms or identities) related to the used credential technology. The latter could be expressed by the predicate CredAttr(*Credential*, *Attribute*) as defined by the framework in Part I (see Listing 4.4). However, its semantics is not accurate. For example, the relation CredAttr(*IdentityCardIdemix*, *OneTimeUsedPseudonym*) expresses that *OneTimeUsedPseudonym* is an attribute of *IdentityCardIdemix*, but it does not express that it is inherently revealed due to the technology used by the credential that is spent.

6.2 Temporal Aspects of Privacy

6.2.1 Motivation

The framework in Part I focuses on analyzing the privacy of a user interacting with the front-end of a single service that, in his turn, interacts with underlying sub-services. In more realistic settings, users interact with front-ends of multiple services. For example, a user first *registers* to the public transport ticketing system to obtain a transport pass. He can then use that pass to *validate bus trips* or to *purchase transport products* (e.g. a monthly reduction plan). This section, hence, extends the framework to capture the privacy of a user who interacts with multiple services.

6.2.2 Modeling Temporal Aspects of Privacy

In the framework of Part I, let $x <_G x_1 <_G x_2 <_G \ldots <_G x_p$ and $y <_G y_1 <_G y_2 <_G \ldots <_G y_q$ denote the order in which services are invoked as a consequence of a user invoking services x and y respectively. A *causal-order relation* $<_G$ is considered between services automatically invoked by x and between services automatically invoked by y. However, the order between services x_i and y_j cannot be expressed by means of this causal-order relation. Therefore, a time-order relation is added to the framework.

Let $x <_T y$ define the *time-order relation*, denoting that the user invokes service x before service y. For example, x represents the registration to the public transport ticketing system and y the purchase of a transport product. For this reason, a finite *Time* domain is added to the framework with discrete ordered time points of a linear (i.e. unbranched) time line. The aforementioned linear time representation can be expressed using the *Linear Time Calculus* (LTC) used for modeling dynamic systems [34] and which is supported by the IDP language. Real world service invocations, occurring in continuous time, are mapped to discrete finite time. Figure 6.1 depicts the two-dimensional *time-causality service invocation graph* of user-invoked services x, y, and z. The services invoked by the user are ordered along the *Time* axis according to the order in which they are invoked by the user. Services that are invoked by other services (i.e. causality) are ordered along the *Causal* axis.

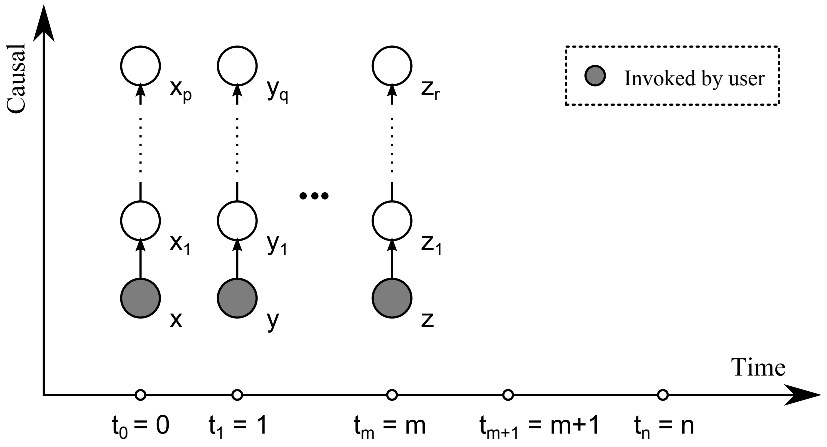

Figure 6.1: A service invocation time-causality graph

The time-order between invoked services implies that some relations must include an additional *Time* argument to express time dependencies. The number of those relations grows linearly with the size of the *Time* domain (i.e. Cartesian product). As a result, the search space to compute solutions (i.e. output models) grows as well. The following modeling decisions are made to compute solutions more efficiently and avoid an explosion of the search space.

- *Finite Time domain.* The *Time* domain is finite and cannot grow dynamically. An estimation of size of the *Time* domain must be made beforehand. It is important to limit the size of the *Time* domain since it influences the size of the search space. However, the smaller the size of the *Time* domain, the weaker the conclusions that can be drawn from the resulting output models. For example, in case no satisfying output models are found within the modeled time interval, there is no guarantee that no models exist outside the interval. To minimize the *Time* domain, services invoked by the user are mapped to consecutive time points of the interval $[t_0 \ldots t_m]$ (see Figure 6.1). No information is lost compared to invoked services that are not mapped to consecutive time points as only the relative time-order is considered in the framework.
- *Non-simultaneous service invocations by the user.* To decrease the complexity of the modeling, a user cannot simultaneously invoke services (i.e. along the *Time* axis). This corresponds with the real world in which a user cannot simultaneously interact with different services (i.e. at least an infinite small time delta can be detected between two invoked services).
- *Service invocation relative order.* Discrete time points in the model represent a relative order between services invoked by the user. Let x and y be two services consecutively invoked by the user at respective real-world continuous time points t_x and t_y, with $t_x < t_y$. The time points are mapped to the respective discrete time points $t_0 \in$ *Time* and $t_1 \in$ *Time*, with $t_0 < t_1$.
- *Orthogonality property.* *Time* and *causality* are orthogonal. This property implies that two services x and *x'* are invoked at the same time point in case $x <_G x'$. This reduces the modeling complexity since it enables specifying a time-order between sub-services of different user-invoked services (e.g. $x_i <_T y_j$, with $x <_T y \wedge x_i <_G x \wedge y_j <_G y$). Moreover, relations $(x, x), (x, x_1), \ldots, (x, x_p)$, expressing that services $x, x_1, \ldots, x_p$ are invoked by service x which is invoked by the user, can be rewritten as $(t_0, x), (t_0, x_1), (t_0, x_2), \ldots, (t_0, x_p)$, with t_0 being the time point at which service x is invoked by the user.

Listing 6.3 illustrates how the invocation of the services depicted in Figure 6.1 are modeled. A *Time* domain of n time points is defined. The services are referred to by elements of the *ServiceIdentifier* domain (see Section 4.3). The causal order between services is defined by the relations described by the predicate *DistrAttrTo* (see Section 4.3). A partial function *ServiceInvokedByUserAt* is defined representing the time at which a service is invoked by the user. The function is partial as only a subset of the *Time* domain is related to invoked services. For instance, there are no services invoked by the user at time points $m+1$ until n in Figure 6.1.

IDP Listing 6.3: Modeling the services invocations of Figure 6.1

`type` $\text{Time} = 0 \ldots n$
`type` $\text{ServiceIdentifier} = \{x; x_1; \ldots; x_p; y; y_1; \ldots; y_q; z; z_1; \ldots; z_r\}$
`type` $\text{DistributionPolicyId} = \{dpx; dpx_1; \ldots; dpx_{p-1}; dpy; dpy_1; \ldots; dpy_{q-1}\}$

$$\begin{aligned}&\mathsf{DistrAttrTo}(\text{DistributionPolicyId}, \text{ServiceIdentifier}, \text{Attribute}, \text{Stakeholder}) = \{\\ &\quad (dpx, x_1, a_x, o_x); (dpx_1, x_2, a_x, o_x); \ldots; (dpx_{p-1}, x_p, a_x, o_x);\\ &\quad (dpy, y_1, a_y, o_y); (dpy_1, y_2, a_y, o_y); \ldots; (dpy_{q-1}, y_q, a_y, o_y);\\ &\quad (dpz, z_1, a_z, o_z); (dpz_1, z_2, a_z, o_z); \ldots; (dpz_{r-1}, z_r, a_z, o_z)\}\end{aligned}$$

`partial` $\mathsf{ServiceInvokedByUserAt}(\text{Time}) : \text{ServiceIdentifier} = \{$
$\quad 0 \to x; 1 \to y; m \to z\}$

6.2.3 Reflection

The modeling of temporal aspects of privacy is computational more complex because the size of the search space increases in relation to the size of the *Time* domain. The modeling decisions in this section minimize the size of the *Time* domain which is more efficient to find solutions (i.e. reducing the size of the search space). Nevertheless, a modeler must foresee how many time points will be needed to find a solution since the *Time* domain is finite and cannot grow dynamically. If the sequence of user-invoked services is known, the modeler can easily estimate the number of time points (i.e. the size of the *Time* domain is equal to number of user-invoked services). Otherwise, the modeler must estimate the number of required time points.

6.3 Conditional Service Invocations

6.3.1 Motivation

In the privacy analysis framework, as presented in Part I, it is assumed that a user owns the credentials requested by a service's access policies. As a consequence, services are executed unconditionally. For example, the traveler must *register* to the public transport ticketing system to obtain a travel pass. To *purchase* transport products he must show his travel pass. Hence, this section extends the framework with modeling concepts required to capture time-dependent credential ownership.

6.3.2 Modeling Conditional Service Invocations

The framework should make a distinction between a service that is invoked and a service that is executed. The former is represented by the predicate $\mathsf{ServiceInvokedAt}(\mathit{ServiceIdentifier}, \mathit{Time})$ specifying the time point at which a service is invoked. The latter is represented by the predicate $\mathsf{ServiceExecuteAt}(\mathit{ServiceIdentifier}, \mathit{Time})$ specifying the time point at which a service is executed. A service can only execute if all access policy requirements are met.

A user initially owns a set of credentials (e.g. his identity card). Other credentials are issued to him via a registration phase (e.g. his travel pass). Modeling concepts are defined capturing both options. The predicate *InitiallyOwnedCredentials* describes the unary relation specifying the credentials initially owned by the user. Listing 6.4 shows an input model representing a user who initially owns an identity card. The second type of relation is described by the fluent (i.e. properties that can change over time) *OwnCredentialAt*, defined in Listing 6.5. They represent the credentials that are owned by a user at a given time point. The definition in Listing 6.5 includes the rules to derive those relations. The first rule includes the initially owned credentials (i.e. $t = 0$). The second rule, which is based on a service's output policy specifications, includes the credentials issued by a service executed at time point t. The final rule includes the credentials that are already owned by the user at time point $t - 1$ and are not revoked at time point t.

IDP Listing 6.4: Modeling an identity card initially owned by the user.

$\mathsf{InitiallyOwnedCredential}(\text{Credential}) = \{IdentityCard\}$

IDP Listing 6.5: Modeling time-dependent credential ownership.

$\mathsf{OwnCredentialAt}(\mathit{Credential}, \mathit{Time})$

$$\overset{Def}{=} \begin{cases} \forall c : \mathsf{OwnCredentialAt}(c, 0) \leftarrow \mathsf{InitiallyOwnedCredential}(c). \\ \\ \forall c \forall t : \mathsf{OwnCredentialAt}(c, t) \leftarrow \exists s \exists op : \mathsf{ServiceExcecute}(s, t) \wedge \\ \qquad\qquad \mathsf{OutputPolicy}(s) = op \wedge \mathsf{OutputCred}(op, c). \\ \\ \forall c \forall t : \mathsf{OwnCredentialAt}(c, t) \leftarrow \exists t_x : \mathsf{OwnCredentialAt}(c, t_x) \wedge \\ \qquad\qquad \neg \mathsf{RevokedCredential}(c, t) \wedge t = t_x + 1. \end{cases} \tag{6.1}$$

6.3.3 Reflection

In the framework as presented in Part I, credential ownership is modeled implicitly. The elements in the *Credential* domain are assumed to be owned by the user from the beginning. In this section, credentials can be issued at any time point in the *Time* domain. For this reason, the framework is extended to make a distinction between initially owned credentials and those that are not owned from the beginning. The former are specified by the modeler in the framework's user model (i.e. the part of the input model describing the user); the latter can be automatically derived from the service's output policy included in the system model (i.e. the part of the input model describing the system).

6.4 Dynamic User Profiles

6.4.1 Motivation

The privacy analysis framework as presented in Part I is limited to a user interacting with a single service front-end that, in turn, interacts with underlying sub-services. Privacy properties are extracted from the user profiles that are built based on what is specified in the service policies and also on the user's trust perception. This chapter extends the framework with newly defined concepts capturing the privacy of a user interacting with multiple service front-ends. As a consequence, modeled user profiles are dynamic as their content can increase with every service that is invoked by the user.

6.4.2 Modeling Dynamic User Profiles

The framework in Part I represents a user profile (see Listing 6.6) as a set of *attributes* stored by an *organization* that can link the set to an *identifier*. An attribute in the user profile is asserted by a *stakeholder* and is collected by a *service* invoked by the user. This representation lacks expressiveness to capture dynamic user profiles.

IDP Listing 6.6: The modeling representation of a user profile as presented in Part I.

$$\mathsf{UserProfile}(\text{Organization}, \text{Identifier}, \text{ServiceIdentifier}, \text{Attribute}, \text{Stakeholder})$$

The framework is, hence, extended with a time-dependent representation capturing dynamic user profiles (see Listing 6.7). It is noteworthy to mention that the predicates representing the static (see Listing 6.6) and dynamic user profiles are overloaded. Dynamic user profiles are represented as a set of *attributes* stored by an *organization* that can link the set to an *identifier*. An attribute in the user profile is asserted by a*stakeholder* and is collected by a service that is invoked by the user at a given *time point*. It is trivial to prove, using the *non-simultaneous service invocation* and *orthogonality* property (see Section 6.2.2), that *ServiceIdentifier* in Listing 6.6 can be replaced by *Time*.

IDP Listing 6.7: The modeling representation of a dynamic user profile.

$$\mathsf{UserProfile}(\text{Organization}, \text{Identifier}, \text{Time}, \text{Attribute}, \text{Stakeholder})$$

6.4.3 Strategy to Infer Dynamic User Profiles

Consider services x and y invoked by a user and organizations o_x and o_y, respectively, providing these services. The user profiles at a given time point t are computed in multiple steps.

STEP 1: build user profile along the causality axis. The first step is similar to the inference strategy as presented in Section 4.4 as it considers the causality dimension. User profiles are separately computed for services $x, x_1, x_2, \ldots, x_p$

and $y, y_1, y_2, \dots, y_q$, with $x <_G x_1 <_G x_2 <_G \dots <_G x_p$ and $y <_G y_1 <_G y_2 <_G \dots <_G y_q$. The resulting user profiles, represented by the predicate in Listing 6.6, are used in the next step.

STEP 2: build user profile along the time axis. The second step considers the time at which services are invoked by a user. The user profiles built are inductively defined in Listing 6.8. The *first definition rule* adds the user profiles associated with the service invoked by the user at time point t, resulting from *STEP 1*. The *second rule* states that existing user profiles are never destroyed. More specifically, if a user profile exists at moment $t-1$, then it exists at moment t as well. The *third rule* considers the collaboration between two organizations, which is modeled with the unidirectional relation $\mathsf{ShareDataWith}(\text{Organization}, \text{Organization})$. Given two organizations o_1 and o_2, $\mathsf{ShareDataWith}(o_1, o_2)$ expresses that o_1 shares its data with o_2. For instance, o_2 pays to get the customer database of o_1. The *final definition rule* considers a user profile of organization o linked to identifier id_x at moment t.

IDP Listing 6.8: Inductive definition of a dynamic user profiles.

$\mathsf{UserProfile}(\mathit{Organization}, \mathit{Identifier}, \mathit{Time}, \mathit{Attribute}, \mathit{Stakeholder})$

$$\overset{Def}{=} \begin{cases} \forall o \forall id \forall t \forall a \forall e : \mathsf{UserProfile}(o, id, t, a, e) \leftarrow \exists s : \\ \qquad \mathsf{UserProfile}(o, id, s, a, e) \wedge \\ \qquad\quad \mathsf{ServiceInvokedByUserAt}(t) = s. \\ \\ \forall o \forall id \forall t \forall a \forall e : \mathsf{UserProfile}(o, id, t, a, e) \leftarrow \exists t_0 : \\ \qquad \mathsf{UserProfile}(o, id, t_0, a, e) \wedge t = t_0 + 1. \\ \\ \forall o \forall id \forall t \forall a \forall e : \mathsf{UserProfile}(o, id, t, a, e) \leftarrow \exists o_x : \\ \qquad \mathsf{UserProfile}(o_x, id, t, a, e) \wedge \mathsf{ShareDataWith}(o_x, o). \\ \\ \forall o \forall id \forall t \forall a \forall e : \mathsf{UserProfile}(o, id, t, a, e) \leftarrow \exists id_x : \\ \qquad \mathsf{UserProfile}(o, id_x, t, a, e) \wedge \\ \qquad\quad \neg\Big(\exists a_x : \mathsf{IdentifierAttr}(id, a_x) \wedge \\ \qquad\qquad \neg\big(\exists e_x : \mathsf{UserProfile}(o, id_x, t, a_x, e_x)\big)\Big). \end{cases} \tag{6.2}$$

6.4.4 Reflection

The inference strategy for computing dynamic user profiles reuses the steps for compiling the user profiles as presented in Part I. This enables splitting the inference strategy in two independent computations. The first step is a pre-computation generating an output model that contains the user profiles as presented in Part I. In the following step, the resulting user profiles are included in the input model to compile the dynamic user profiles. This approach makes the computation of dynamic user profiles more efficient.

The dynamic user profiles are complementary to the profiles as presented in Part I. The former contain the personal attributes collected from a user interacting with multiple service front-ends; the latter keep the personal attributes of a user who invoked a single service. Both are combined to extract feedback regarding the user's privacy. For example, a user revealed his e-mail address and name to a service provider during previous service invocations. His name and e-mail address are part of the same dynamic user profile as the service provider can link both attributes (i.e. the user is identifiable towards the service provider). At time point t, the user invokes a service. It can be derived from the user profile associated with the invoked service that the user revealed his e-mail address. Even though he only revealed his e-mail address, the service provider can identify him since the provider can link his e-mail address to his name (i.e. derived from the dynamic user profile).

6.5 Modeling the Privacy-Preserving Ticketing System

The present section applies the modeling concepts defined in this chapter to analyze the privacy in a Privacy-preserving Public Transport Ticketing System (PPTS) [129]. The system provides a privacy-friendly alternative to state-of-the-art ticketing systems in which a traveler's transactions (e.g. purchases, trips) can be linked. First, a background of public transport ticketing systems is given, followed by a high-level overview of the PPTS system. The PPTS system is then modeled and queried for privacy properties.

6.5.1 Background

As technological advances are creating possibilities for advancing the ticketing systems, there are a number of research initiatives focusing on this topic. Some solutions try to increase the efficiency of ticket purchasing. Initial proposals focus on existing mobile features [107, 35]. SMS messages are investigated in combination with the NFC technology, resulting in higher user satisfaction and usability [89]. The work of Finzgar and Trebar [83] uses NFC communication channels and QR codes to reduce the infrastructure requirements and simplify the existing systems. However, the use of NFC requires careful privacy considerations and users tend to express privacy concerns about solutions that use this technology [99]. Similarly, multiple studies identify that privacy is a serious issue in advanced ticketing systems. They can collect information about users, such as locations and movements [84, 142]. The way some commercial systems are deployed also create grounds for concern [184]. For example, the Washington D.C. Metro was functioning for years without a clearly defined privacy policy [167, 96]. Many solutions rely on cards with unique identifiers and utilize other personal information, even credit card data [166]. To tackle this, there is a limited number of research proposals. Verslype et al. [179] design a system based on the use of anonymous credentials. While the design focuses on protecting the privacy of the user, it is not flexible for public transport systems. It does not consider pricing per traveled distance and reduction plans. On the other hand, work of Jorns et al. [109] focuses on the problem of location services. As the telecommunications service providers gradually open their interfaces for mobile applications that use travelers' location and presence information, privacy issues arise. The authors propose a pseudonymous system to keep the users identity hidden. However, the users are pseudonymous and every ticket contains a unique identifier, linking its purchase and usage. Finally, the proposal of Heydt-Benjamin et al. [96] uses an e-cash payment scheme, anonymous credentials and proxy re-encryption for concealing personal data, while ensuring correct payment. However, the system describes the functioning on a higher level of abstraction and system flexibility can be limited compared to currently offered services.

6.5.2 Public Transport Ticketing System Architecture

The ticketing service is offered through collaboration of multiple stakeholders. It usually comprises the *ticketing system operator* (TSO), who issues travelers' passes and handles the related interactions, and the *public transport organizers* (PTO), who organize the actual public transport. The TSO usually collaborates with multiple PTO entities. It manages the personal information obtained during the registration procedures and the identifiers contained in the passes. The PTO is able to record the trips taken and the disclosed data, such as the unique pass identifier [136, 42]. Existing systems collect this travel data in order to optimize the provided services. However, recording unique identifiers, such as smart card serial numbers, is a major privacy concern. It allows for creating profiles and, possibly, linking them with information obtained during registration, including a traveler's personal identifiable information. Some solutions for mitigating privacy concerns [142] rely on corporate level policies that separate travel transactions from user data and restrict access to only one of the databases. However, the privacy depends on the discipline inside the organization and may be prone to internal or external attacks which could link identifiable data to travel patterns.

6.5.3 Privacy-Preserving Public Transport Ticketing System Services

This sections gives a high-level overview of the services in the PPTS system.

Similarly to existing systems, the proposed scheme consists of interactions between a traveler, a ticketing system operator (TSO) and a public transport organizer (PTO). Before participating in the PPTS system, every traveler needs to install a smartphone application provided by the TSO (i.e. *PPTS application*), which interfaces with the ticketing system. All travelers are also issued with credentials which serve as personal passes. Before utilizing the public transport services, a traveler recharges his ePurse balance, which is stored on his smartphone, with single use tickets. The application on his smartphone can also store products such as monthly passes. Even though the products are linked to the traveler's credential in order to prevent unauthorized sharing, no identifying data is disclosed at the time of purchase. In order to use the transport services, a traveler's application contacts the TSO to be issued with a single-use ticket for the desired ride. For this, the application spends or proves possession of a previously purchased product and demonstrates that it is linked to the credential the traveler owns. The acquired temporary ticket is validated by the PTO's validation machine on the vehicle. At the end of the journey, the traveler's phone interacts with the validation machine once more to be issued with change in case the spent product is not fully used. The obtained change proof can then be exchanged with TSO for a long-term token. Although there are multiple interactions with TSO/PTO, they are mostly transparent to the traveler, as he only initiates the ticket issuance and travel start/end, while all the other operations are automatically performed by the application. The services in the PPTS system are discussed next. The detailed protocol interactions can be found in [129].

Traveler Credential Issuance. After installing the PPTS application on the smartphone, a traveler interacts with the online ticketing application of the

TSO to be issued an anonymous credential which serves as a personalized pass and is denoted as the *PPTS credential*. Credential attributes include the traveler's personal information (e.g. name and birth date), the validity information and the traveler's *ticketing system secret* (*tss*), which is different from the credential's master secret and is not disclosed to the TSO. The personal information is provided by the traveler, similar to online registration procedures in existing systems. In case additional assurances are required, the traveler's electronic identity card can be used for asserting this data. In systems where the smart card is delivered to the traveler's home address, there is additional confirmation of the provided address information. The PPTS system can also realize this, by having a code sent via post, which is used upon reception to complete the registration and credential issuance. On the whole, this approach improves the efficiency, while providing the same guarantees as in existing public transport ticketing systems, such as in the Dutch OV-card system. Idemix credential technology allows for all subsequent interactions with the TSO or PTO to remain unlinkable to the credential issuance. Additionally, for a better privacy-protection it is assumed that the network layer meta-data does not allow linking activities of the same user.

ePurse balance recharge. In order to recharge the ePurse balance, the traveler makes a request via his PPTS application to the TSO and pays the desired amount. The request only contains the requested amount to be added to the ePurse. In order to be granted the recharge, the traveler's PPTS application also proves that the traveler holds a valid PPTS credential by creating a zero-knowledge proof of knowledge. In case of a successful proof, the TSO can generate an appropriate invoice for the traveler. The traveler makes the payment via a third-party payment service provider (PSP). After a successful payment, the TSO is notified and issues a number of *eTokens*. The public details in the tokens contain the denomination information, which add up to the requested recharge amount. The denomination of each token corresponds to the maximal charge for a ride. This way, the service provider is assured that the full trip fee will be paid, because the traveler spends one token when entering the vehicle and is reissued the change if a cheaper ride was taken. In the final step, the traveler's PPTS application stores the details of each *eToken*.

Purchase of travel products. The proposed system also allows for the PTO providers to offer multiple-use products, such as monthly discounts, which is a service already present in current public transport ticketing systems (e.g. the Dutch U-OV bus or NS railway services). The traveler initially requests a multiple-use product via his PPTS application. The request only records the type of product that is requested. The traveler also proves ownership of a valid

PPTS credential. It may, additionally, be required to prove certain properties, such as the age group, using the birth date attribute of the credential. The proof is provided in the form of a signed zero-knowledge proof (SPK). It only convinces the provider that the attributes in the traveler's credential satisfy the given properties, and hide other information recorded in the credential. If the proof verifies, the PTO can generate an appropriate invoice for the traveler. The traveler makes the payment via a third-party payment service provider (PSP). Similar to the ePurse recharge, the requested product is issued by the PTO in the form of a batch of *eTokens*. The tokens contain a hidden link to the traveler's PPTS credential and the product specification, such as the product validity, product type and issuing PTO, in the public information part. These tokens are spent with the TSO before starting a trip. Until the restrictions on these tokens (such as validity date) is met, the application interacts with the PTO before the last token is used; the last token can be spent with the TSO to obtain a new bundle of *eTokens*.

Validation of the trip start. For charging the ePurse, the traveler's PPTS application interacts with the TSO to spend an *eToken* and obtain a temporary single-use ticket from the TSO. The ticket is sent to the PTO's validation machine over an NFC channel upon entering the bus, where it is verified. Alternatively, the traveler's PPTS application could interact directly with the PTO's validation machine to spend an *eToken*. However, this would take too much time to validate a trip start as the verification of *eToken* is a computational intensive task given a validation machine running on a low-end processor.

Validation of the trip end using the ePurse. In existing systems such as in the Dutch public transport ticketing system, the travelers validate their cards when exiting the vehicle, in order to receive back the difference between the guaranteed amount and the price of their journey. Similarly, in the PPTS system where the traveler's PPTS application establishes an anonymous short-range communication with the PTO's validation machine to receive the change. The validation machine calculates the applicable fare before generating the trip-end ticket. This ticket is then sent to the traveler's PPTS application which forwards it to the TSO that reissues the change.

6.5.4 The PPTS Model

Figure 6.2 depicts the conceptual model of the PPTS which is realized in IDP [1]. The model shows the system's stakeholders, the user's credentials, and the personal information collected and distributed by the services. The model decomposes the services, which are presented in Section 6.5.3, into sub-services. Table 6.2 details the service decomposition and associates the services with steps in the considered protocols. The detailed protocol steps can be found in [129]. The services in the table are denoted as *Service Provider::Service[n]*, with *n* the steps in the protocol considered. The *ePurse balance recharge* and *purchase of travel products* services include a payment phase where the traveler pays for his purchases via the online payment interface provided by the third-party payment provider (PSP). The payment makes the traveler identifiable towards the bank. Hence, additional payment details are modeled according to state-of-the-art payment services. It involves a third-party *payment services provider* (PSP) acting on behalf of the payee as a proxy service redirecting payments to the user's *bank*. The remaining stakeholders in the PPTS that are modeled are: the *user*, the *TSO*, and the *PTO*. The PPTS application is represented in the model by the TSO since it provides the application. Consequently, it is assumed that the data stored in the application can be accessed by the TSO.

[1]see `https://github.com/decroik/inspect-temporal-privacy`

Table 6.2: Decomposition of the PPTS services into sub-services (the detailed protocol steps can be found in [129]).

Travel credential Issuance
TSO::IssuePPTSCred*
ePurse Balance Recharge
TSO::Recharge*[$1-4$]) $\rightarrow$ PSP::SelectBankRecharge[5] $\rightarrow$ Bank::PayRecharge[5] $\rightarrow$ PSP::ConfirmPaymentRecharge[5] $\rightarrow$ TSO::IssueRecharge[$6-10$]
Purchase of Travel Products
PTO::BuyProduct*[$1-4$] $\rightarrow$ PSP::SelectBankProduct[5] $\rightarrow$ Bank::PayProduct[5] $\rightarrow$ PSP::ConfirmPaymentProduct[5] $\rightarrow$ PTO::IssueProduct[$6-10$]
Validation of the trip start
TSO::TripStart*[$1-13$] $\rightarrow$ TSO::GetTripProofOfPayment[14] $\rightarrow$ PTO::ValidateTripStart[$15-18$]
Validation of the trip end
PTO::TripEnd*[$1-7$] $\rightarrow$ TSO::ReceiveTripEnd[$8-9$] $\rightarrow$ TSO::IssueChange[$10-16$]

*: A user-invoked service
Service notation: *Service Provider::Service[protocol steps]*

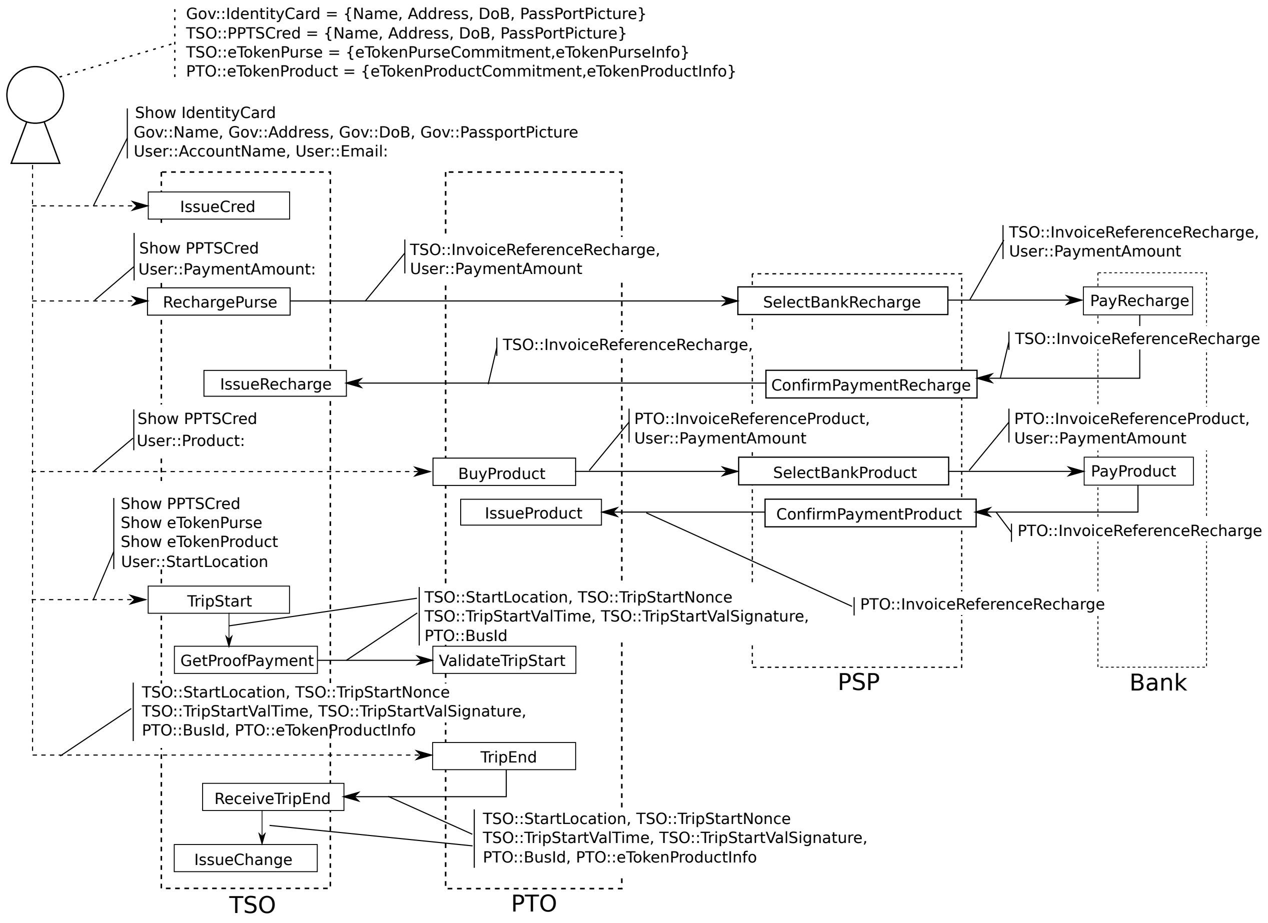
Gov::IdentityCard = {Name, Address, DoB, PassPortPicture}
TSO::PPTSCred = {Name, Address, DoB, PassPortPicture}
TSO::eTokenPurse = {eTokenPurseCommitment,eTokenPurseInfo}
PTO::eTokenProduct = {eTokenProductCommitment,eTokenProductInfo}
Show IdentityCard
Gov::Name, Gov::Address, Gov::DoB, Gov::PassportPicture
User::AccountName, User::Email:
IssueCred
Show PPTSCred
User::PaymentAmount:
RechargePurse
TSO::InvoiceReferenceRecharge,
User::PaymentAmount
SelectBankRecharge
TSO::InvoiceReferenceRecharge,
User::PaymentAmount
PayRecharge
TSO::InvoiceReferenceRecharge,
TSO::InvoiceReferenceRecharge
IssueRecharge
ConfirmPaymentRecharge
Show PPTSCred
User::Product:
BuyProduct
PTO::InvoiceReferenceProduct,
User::PaymentAmount
SelectBankProduct
PTO::InvoiceReferenceProduct,
User::PaymentAmount
PayProduct
IssueProduct
ConfirmPaymentProduct
PTO::InvoiceReferenceRecharge
Show PPTSCred
Show eTokenPurse
Show eTokenProduct
User::StartLocation
TripStart
TSO::StartLocation, TSO::TripStartNonce
TSO::TripStartValTime, TSO::TripStartValSignature,
PTO::BusId
PTO::InvoiceReferenceRecharge
GetProofPayment
ValidateTripStart
PSP
Bank
TSO::StartLocation, TSO::TripStartNonce
TSO::TripStartValTime, TSO::TripStartValSignature,
PTO::BusId, PTO::eTokenProductInfo
TripEnd
ReceiveTripEnd
TSO::StartLocation, TSO::TripStartNonce
TSO::TripStartValTime, TSO::TripStartValSignature,
PTO::BusId, PTO::eTokenProductInfo
IssueChange
TSO
PTO

Figure 6.2: Conceptual model of the PPTS.

The credentials modeled are:

- the *electronic identity card* issued by the government, represented by *Gov::IdentityCard*, containing attributes $\{Name, Address, PassportPicture, DoB, SSN, SerialNumber\}$.
- the *PPTS credential* issued by the TSO, denoted as *TSO::PPTSCred*, containing attributes $\{Name, Address, PassportPicture, DoB\}$
- the *ePurse eTokens* issued by the TSO, represented by *TSO::eTokenPurse*, containing the committed secret *eTokenPurseCommitment* linking the *eToken* to the PPTS credential and the information part *eTokenPurse-Info*.
- the *travel product eTokens* issued by the PTO, represented by *PTO::eTokenTravelProduct*, containing the committed secret *eTokenProductCommitment* linking the *eToken* to the PPTS credential and the information part *eTokenProductInfo*.

6.5.5 Privacy Assessment and Results

The present section analyzes the privacy of a user interacting with multiple service front-ends. These interactions are represented by a *path of user-invoked services*. The privacy analysis consists of two steps. The first step verifies the conditions regarding the paths of user-invoked services. The second step performs a privacy analysis based on the user profiles compiled due to the execution of the services in the considered path. The privacy feedback is extracted from these user profiles using the queries presented in Chapter 5 (i.e. *Q1*, *Q2*, *Q3*, *Q4*).

The PPTS system is modeled in IDP to illustrate how the newly defined modeling concepts and above-mentioned steps can be applied to analyze the privacy in systems in which a user interacts with multiple service front-ends. It is assumed that the user fully trusts the stakeholders in the PPTS (i.e. he trusts that organizations act according to their service policies).

STEP 1: Verifying the execution of the path of user-invoked services. This consists of two types of verifications (i.e. *V1* and *V2*) which are defined in Table 6.3.

- *Execution of the services in a given path of user-invoked services* (*V1*). It is verified that the service access conditions (i.e. credential ownership)

Table 6.3: STEP 1: Verifying the execution of the path of user-invoked services.

V1. Execution of the services in a given path of user-invoked services.

r5.1 $\mathsf{ServiceInvokedByUserAt} = \{0 \to \mathit{IssuePPTSCred}, 1 \to \mathit{Recharge}, 2 \to \mathit{BuyProduct}, 3 \to \mathit{TripStart}, 4 \to \mathit{TripEnd}\}$

compute $\forall o \forall id \forall t \forall a \forall e \big(\mathsf{UserProfile}(o, id, t, a, e)\big)$

V2. Reachability of services.

$$\forall s \big(\mathsf{Reachability}(s) \Rightarrow \exists t \big(\mathsf{ServiceExecuteCompleteAt}(t) = s\big)\big).$$
$$\forall s \forall s' : \mathsf{Reachability}(s) \wedge \mathsf{ReachabilityViaService}(s') \Rightarrow$$
$$\exists t \exists t' : \mathsf{ServiceExecuteCompleteAt}(t) = s \wedge \mathsf{ServiceExecuteCompleteAt}(t') = s' \wedge t' < t.$$
$$\forall s \forall s' : \mathsf{Reachability}(s) \wedge \mathsf{ReachabilityNotViaService}(s') \Rightarrow$$
$$\exists t \exists t' : \mathsf{ServiceExecuteCompleteAt}(t) = s \wedge \mathsf{ServiceExecuteCompleteAt}(t') = s' \wedge t < t'.$$
$$\forall s \forall c \big(\mathsf{Reachability}(s) \wedge \mathsf{ReachabilityNotCredential}(c) \Rightarrow$$
$$\exists t \big(\mathsf{ServiceExecuteCompleteAt}(t) = s \wedge \neg\big(\exists t' \big(\mathsf{OwnCredentialAt}(c, t') \wedge t < t'\big)\big)\big)\big).$$

with $\mathit{ServiceExecuteComplete}(\mathit{Time}) : \mathit{Service}$

$$\overset{Def}{=} \begin{cases} \forall s_u \forall t : \mathsf{ServiceExecuteCompleteAt}(t) = s_u \leftarrow \\ \quad \mathsf{ServiceInvokedByUser}(s_u) \wedge \mathsf{ServiceExecuteAt}(s_u, t) \wedge \\ \quad \neg\big(\exists s : \mathsf{ServiceInvokedBy}(s, su) \wedge \neg\mathsf{ServiceExecuteAt}(s, t)\big). \end{cases}$$

r6.1 $\mathsf{Reachability} = \{\mathit{TripEnd}\}$
r6.2 $\mathsf{ReachabilityViaService} = \{\mathit{Recharge}\}$
r6.3 $\mathsf{ReachabilityNotViaService} = \{\mathit{IssuePPTSCred}\}$
r6.4 $\mathsf{ReachabilityNotCredential} = \{\mathit{eTokenPurse}\}$

search $\forall t \forall s \big(\mathsf{ServiceInvokedByUserAt}(t) = s\big)$ and compute $\forall o \forall id \forall t \forall a \forall e \big(\mathsf{UserProfile}(o, id, t, a, e)\big)$

are met for each service (including the sub-services) so that they are able to execute. The access conditions, such as proving ownership of an electronic identity card, are extracted from the access policies (see Section 4.3) associated with the services. An example is given by *r5.1* in Table 6.3.

- *Reachability of services* (*V2*). The reachability of user-invoked service s (*Reachability(s)*) is defined as the ability of s to execute during the considered *Time* domain. Considering the access policies of each service, a search is performed to find a path of user-invoked services enabling the execution of s. For instance, *r6.1* queries whether *TripEnd* can execute. It is assumed that a service s invoked by a user at time point t can execute only if all sub-services on which it relies can execute as well. This is represented by the function $ServiceExecuteCompleteAt(t) = s$ (see Table 6.3). The search may result in multiple alternative paths of user-invoked services. Examples of additional conditions that can be checked during the reachability analysis are presented below.

– The modeler specifies a set of services *s'* that must execute before service *s* executes. For example, rule *r6.2* specifies that *Recharge* must execute before *TripEnd*. This may be used by the modeler to give a hint to the reasoner to reduce the time for finding a solution.
– The modeler specifies a set of services *s'* that excludes the paths of user-invoked services in which the services *s'* execute before service *s* executes. This is useful for verifying the correctness of the access conditions of service *s*. In case a path is found in which services *s'* are executed before service *s*, the designer must reconsider the design. For instance, rule *r6.3* verifies if it is possible to validate the start of a bus trip without having been issued a PPTS credential.
– The modeler specifies a set of credentials *c* that excludes the paths of user-invoked services in which services issue the credentials *c* before service *s* is executed. This is useful for verifying the correctness of the access conditions of service *s*. In case a path is found in which service *s* can execute before the user owns credentials *c*, the designer must reconsider the design. For example, rule *r6.4* verifies if it is possible to execute *s* without having eTokens in the ePurse.

STEP 2: Computation of the user profiles associated with the path of user-invoked services. User profiles are computed for each time point in the *Time* domain in case the above-mentioned verifications result in a path of user-invoked services in which all services are able to execute. Queries Q1, Q2, Q3 and Q4 (see Chapter 5) can then used to extract feedback from these profiles regarding the privacy of a user in the system under analysis.

The modeler verifies if a path of user-invoked services exists in which the traveler must recharge his ePurse (*r6.2*) before *TripEnd* executes (*r6.1*). This results in a path of user-invoked services $\{0 \rightarrow \mathit{IssuePPTSCred}, 1 \rightarrow \mathit{Recharge}, 2 \rightarrow \mathit{BuyProduct}, 3 \rightarrow \mathit{TripStart}, 4 \rightarrow \mathit{TripEnd}\}$.

User profiles are then computed based on the path of user-invoked services resulting from *r6.1* and *r6.2*. Table 6.4 shows the anonymity level (i.e. feedback from query *Q1*) of the traveler in the PPTS system considering the above-mentioned path of services. The traveler is identifiable towards the TSO due to the personal information (e.g. his name and address) disclosed during the PPTS credential request ($t = 0$). He can also be identified by the TSO and PTO if he recharges his ePurse ($t = 1$) or purchases transport products ($t = 2$). In both cases, the traveler can be identified via the TSO's or PTO's bank account, which records the invoice reference number associated with the purchase and the traveler's name in the payment transaction history.

Table 6.4: The traveler's anonymity level (*Q1*) in the PPTS system when executing $\{0 \rightarrow$ *IssuePPTSCred*$, 1 \rightarrow$ *Recharge*$, 2 \rightarrow$ *BuyProduct*$, 3 \rightarrow$ *TripStart*$, 4 \rightarrow$ *TripEnd*. A traveler can be (I)dentifiable, (P)seudonymous or (A)nonymous.

	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$
TSO	I	I	I	I	I
PTO	A	A	I	I	I
PSP	A	P	P	P	P
Bank	A	I	I	I	I

The above-mentioned user profiles contain the attributes collected from services invoked during the time interval [0..4]. However, they cannot provide feedback regarding the user's privacy in case he uses a single service front-end. Additional feedback can be extracted from user profiles as presented in Listing 6.6 that are computed. Table 6.5 shows the anonymity level (i.e. feedback from query *Q1*) of the traveler in case he validates the start (i.e. *TripStart*) and the end (i.e. *TripEnd*) of his bus trip. The table shows that in both cases the traveler is anonymous towards the organizations in the PPTS system.

Table 6.5: The traveler's anonymity level (*Q1*) in the PPTS system for *TripStart* and *TripEnd*. A traveler can be (I)dentifiable, (P)seudonymous or (A)nonymous. The results are based on the user profiles as presented in Listing 6.6.

	TripStart	*TripEnd*
TSO	A	A
PTO	A	A

Based on the results shown in Tables 6.4 and 6.5, it can be concluded that a traveler's trip start and end transactions cannot be linked to the user profiles kept by any of the PPTS stakeholders. Additionally, the modeler verifies if a traveler cannot validate the trip end (*r6.1*) in case he did not request a PPTS credential (*r6.3*) or in case he did not recharge his ePurse (*r6.2*). In both cases, no solutions are found.

6.5.6 Reflection

The verifications defined in this section(i.e. *V1* and *V2*), can be combined with the queries from Chapter 5 to analyze the privacy of a user who interacts with multiple service front-ends. Based on the privacy analysis of the PPTS system, it can be concluded that the memory consumed by the privacy analysis is too high. This may have two reasons. The first reason are the time dependencies of the relations, which cause the search space increases in relation to the size of the *Time* domain. The second reason is the degree of freedom in the path of user-invoked services (i.e. the number of unknown services that is searched for). The degree of freedom can be reduced by giving hints (see rule *r6.2* in Table 6.3). Another approach is to split the inference for the generation of output models in two steps in which the output of the first step is the input for the second step. The first step considers the causal-order between services to compute the user profiles. In the second step, the time-order between services is considered to compute the user profiles. This strategy, however, limits the flexibility of the privacy analysis. For example, it is impossible to directly search for a path of user-invoked services so that a user is anonymous towards the TSO when validating the trip start. This can be solved by first generating all possible paths of user-invoked services in which *TripStart* is reachable. Then, the anonymity level of the user towards the TSO can be extracted from the user profiles built for each path. Only those paths for which a user is anonymous towards the TSO are selected.

6.6 Conclusions

This chapter showed how to extend the capabilities of the privacy analysis framework to capture the privacy of a user who interacts with multiple service front-ends. Newly defined modeling concepts are added to the framework that represent the properties of advanced credential technologies, temporal and conditional aspects of service invocations. Furthermore, the inference strategy to compile user profiles as presented in Part I is extended with an additional step that captures time-dependent user profiles. This chapter also presented a privacy-preserving public transport ticketing system (PPTS) to validate the privacy analysis framework. The framework extensions are realized in IDP to analyze the privacy in the modeled PPTS system. This analysis showed that even for a medium-sized model, such as the model representing the PPTS system, no solution could be found because of the memory required by the privacy analysis. Hence, the analysis had to be split in two independent steps in order to extract privacy properties from the model representing the PPTS system.

Part III

Inferring Service Provider Accountability

Chapter 7

Service Provider Accountability

Chapters 3, 4 and 5 present an approach for modeling advanced electronic services and inspecting their privacy. The approach gives an overview of the profiles that can be compiled by organizations. These, in their turn, can contain sensitive personal information that can be linked to an individual. The profiles are the input for a qualitative privacy analysis that provides meaningful feedback to both end-users and service designers. The former can select a service based on the minimal data disclosure principle; the latter can select privacy enhancing technologies and define consistent privacy policies from the earliest design stage.

In many cases, privacy policies can be downplayed to merely vague declarations of intended data handling practices. Often, a guarantee of compliance between the actual data handling practices and the declared ones is absent. As a result, individuals remain uncertain about the processing of their personal information. Even if all entities publish detailed unambiguous privacy policies easily interpretable by an individual, they are still opaque towards individuals because systems may involve many collaborating entities. It is useful for individuals to have a system-wide panoramic view on data handling practices rather than a per-entity view. The protection of personal information is not solely the responsibility of consumers. Service providers too, must be enforced to see privacy as a major obligation. Therefore, the upcoming European General Data Protection Regulation explicitly cites the *accountability* principle, which puts responsibility on *data controllers* (e.g. service providers) as well. They have the obligation to demonstrate compliance with Regulations and internal policies and must implement mechanisms to ensure this. For example, log evidence containing traces of data processing can be used by auditors to verify compliance. Accountability is a privacy enforcement principle strongly linked with privacy by design [47], putting privacy as a concern that must be integrated from the earliest design stage. Even though loggings may leak privacy sensitive information, privacy-preserving logging solutions that improve the transparency of data handling practices towards individuals exist. For example, Pulls et al. [144] propose a distributed privacy-preserving transparency logging solution that makes it impossible to link multiple log entries associated to a same individual.

Contributions. The present chapter realizes an inference model providing individuals with global accountability guarantees of multi-component systems. The approach maps different user expectations to log evidence of declared data handling practices. The chapter mainly focuses on interactions between organizations. The feedback provided by this approach is useful for both end-users and auditors who conduct privacy assessments on behalf of the end-users. The approach is validated on a railway surveillance infrastructure.

The contributions in the present chapter are peer-reviewed in [65], which is a joint work with Denis Butin (INRIA Lyon). The contributions to this paper include assistance with defining the railway surveillance case, defining the inference strategy providing individuals with global accountability guarantees, and the IDP realization of the inference model.

The present chapter continues the discussion of related work on formalizations of service provider accountability and privacy in Section 7.1. The modeling approach is applied to a railroad surveillance infrastructure use case, which is presented in Section 7.2. Section 7.3 gives an overview of the accountability inference framework and its components and applies the framework to the use case. The approach and its results are evaluated in Section 7.4. The chapter ends with conclusions about limitations and possible extensions of the framework (see Section 7.5).

7.1 Related Work

The approach of using standardized privacy policies to enable accountability by clarifying obligations is widespread. In particular, the idea of performing a posteriori verification of compliance by combining privacy policies with data handling logs appears in [183]. The gap between system event logs and logs at the level of abstraction of privacy policies is addressed in [40]. The consequences of log design choices for log analysis and accountability are addressed in [39].

Adequate log design for compliance checking is tricky because of the numerous possible semantic ambiguities. Both papers presume a single data controller rather than the setting of this chapter – a constellation of interacting data processors with different, potentially incompatible privacy policies.

Logs tracing events are the source for audits which are conducted to verify compliance with privacy policies. Audits are time-consuming as privacy policies are usually expressed in a natural language. Different research approaches propose formal semantics – machine interpretable – for expressing privacy policies. As a consequence audits can be automated and become less expensive. The authors of [155] address privacy compliance in big data systems. They define a privacy policy specification language familiar to natural language, yet machine interpretable while suited for privacy-experts who have no experience with formal representations. More expressive semantics are offered by [85, 26]. In practical settings, logs trace events in systems may be incomplete. Log entries may be overwritten by more recent ones or they may be inconsistent, or it may be impractical to log particular events. Policies are expressed in first-order or first-order temporal logic, enabling the automated reasoning on incomplete logs to check compliance with privacy policies. Other work [173] defines formal semantics to express purpose restrictions on data processing. Systems are considered as partial black boxes. An automated planning based approach predicts log traces. Actual log traces which differ from predictions are flagged as potential violations. Audits are considered by Blocki et al. [32, 33] as a game between an auditor and adversary (e.g. an employee of the data controller). In this game the former must find the best audit strategy to defeat the latter.

The approach in the present chapter uses a first-order logic based language that is capable to express typical aspects of data handling practices, such as purpose restrictions, data retention limits, and data forwarding. Incomplete logs are considered as well. More specifically, the approach can represent policy statements for which log evidence is missing as mere declared statements. Moreover, the approach associates statements with particular components of which a system is comprised.

Besides the area of computer science, accountability is also a topic of privacy regulations [91]. A key question related to this chapter is how far data controllers should be required to go to demonstrate compliance. Distinctions are sometimes [53] made between different levels of accountability, ranging from public declarations of intent to full technical transparency, such as the ones advocated here. The adequacy of procedures, i.e. organizational measures, is very often under discussion. Privacy Impact Assessments are often advocated [188] and can be seen as a bridge between accountability of procedures and accountability of practice if the assessment is conducted in sufficient detail. The question of privacy-preserving surveillance infrastructures is particularly addressed in the PARIS project [2], from an interdisciplinary angle.

7.2 A Railway Station Surveillance Scenario

To illustrate the modeling approach, the scenario of video camera surveillance in a railway station is considered. The cameras in the railway station collect images (i.e. personal data) of individuals (it is assumed that cameras do not record sound). Several categories of personal data can be inferred from camera recordings, such as identification through face detection [180], gait recognition [115], behavioral tracking [125] and many others. Signs inform passersby that the *Railway Company* installed *Cameras* for video surveillance. The cameras provide the railway's *Monitors* in the control room with real-time video feeds containing *Blurred Faces* and *Gaits* of travelers. Furthermore, detailed images of individuals' *Full Body* and *Gait* are stored in the railway's *Image Database* serving as *Evidence* in legal investigations. Only authorized *Image Processors* employed by the railway company have access to it. Additionally, surveillance *Guards* employed by a third-party *Security Company* patrol in the station. They are authorized to view real-time images on the monitors, and carry a *Mobile Device* for registering *Contextual Data* (e.g. time and location) in case of incidents. These devices are connected with the *Status Database*, property of the security company. It is only accessible for the security company's *Status Processors* upon request of legal institutions for collecting *Evidence*. It is assumed that surveillance guards are honest and obey the policies imposed by the security company (e.g. guards are not permitted to film incidents with their smartphone).

The trusted auditor (acting on behalf of a filmed individual) is external to the model and the data handling statements from the entities collecting personal data, listed in Table 7.1 are used to illustrate the approach.

7.3 Components of the Accountability Inference Model

The present section describes the framework's building blocks (depicted in Figure 7.1) in detail. Entities, all related to a core organization, provide individual statements about their data handling practices. These statements can be provided together with unverified logs, verified logs, or by those

Table 7.1: Camera surveillance data handling statements. Entity statements are (D)eclarative, (L)ogged-unverified or Logged-and-(V)erified.

(R)ailway Company, (C)amera, (M)onitor, (I)mage Database Statements		
Stat R.1	(L)	Full body pictures with blurred or clear faces, gaits, heights, and behavior are recorded for incident detection.
Stat R.2	(D)	Collected pictures containing evidence of incidents can be forwarded to legal authorities upon their request.
Stat R.3	(L)	Pictures are never collected for commercial purposes.
Stat R.4	(L)	The maximal retention time for any category of collected personal data is 60 days.
Stat C.1	(L)	Cameras in the station record full body pictures with blurred or clear faces, gaits, heights, and behaviors of travelers for incident detection purposes.
Stat M.1	(L)	Guards monitor in real-time full body pictures with blurred faces, gaits, heights, and behaviors of travelers in the station for incident detection purposes.
Stat I.1	(L)	Full body pictures with clear faces are stored as evidence of possible incidents.
Stat I.2	(V)	Access to stored full body pictures with clear faces is only granted to the image processor upon request of the legal authorities.
Stat I.3	(V)	Full body pictures with clear faces, gaits, heights, and behavior are never processed for the purpose of identification.
Stat I.4	(D)	Stored images are deleted after 30 days, unless they are being used as evidence in legal cases.
(S)ecurity Company, M(O)bile Device, Status (D)atabase Statements		
Stat S.1	(D)	Time and location of incidents are collected as evidence.
Stat S.2	(L)	Time and location of incidents are only forwarded to legal authorities upon request.
Stat O.1	(V)	Surveillance guards collect time and location as evidence in case of incidents.
Stat D.1	(V)	Time and location of incidents are collected as evidence.
Stat D.2	(V)	Access to stored time and location of incidents is granted to status processors for gathering evidence.
Stat D.3	(V)	The time and location of incidents are deleted after 90 days unless they are being used as legal evidence.

which exist on their own, without companion evidence. Different categories of personal data can be modeled. As a consequence, statements are fine-grained enough to express different guarantees about various types of personal data. The data subject is represented by a trusted auditor. This auditor takes into account the subject's trust perceptions. Global accountability guarantees are automatically computed using the computation rules in the

System Independent Part of the framework. These guarantees are described in the *Global Accountability Profile* (GAP) that is automatically inferred, using a *Knowledge Base System* (IDP), from the *System Model* and the *User Model*, both part of the framework's *Input Model*. The system model includes the individual statements, the relations between the entities expressing the statements, and the level of evidence characterizing the statements. The user model includes the level of trust of the user. As a consequence, the global accountability guarantees take into account both factual evidence and subjective appreciations of privacy risks. This combination reflects the fact that different data subjects demand different levels of proof to be satisfied. The model provides data subjects with an overview of the accountability guarantees resulting from a set of interacting entities. In addition, it allows them (or the auditor, on their behalf) to check whether their personal privacy preferences are compatible with this global accountability panorama. The remainder of this section further details the framework's elements.

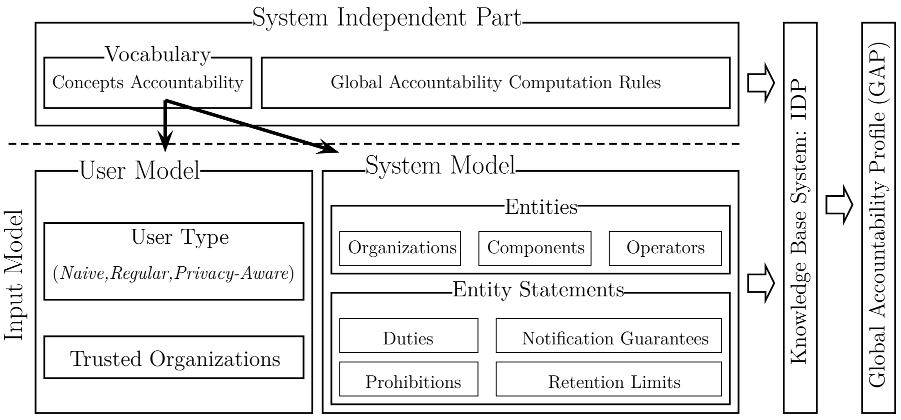

Figure 7.1: Structure of the global accountability inference model.

7.3.1 Personal Data

Organizations collect personal data of data subjects that interact with systems owned by these organizations. Being accountable to data subjects involves clarifying which types of their personal data are harvested and used. These categories are represented by the *DataCategory* type. All categories of collected personal data involved in a given scenario must be spelled out in the input model as the contents of *DataCategory*.

One can define hierarchies of personal data categories. This models the fact that categories of personal data can be subsets of other categories, e.g. the age of an individual gives strictly more information than a predicate on whether the individual is over 18. The data category hierarchy is represented using *DataCategoryOf(DataCategory,DataCategory)*, which deduces hierarchical knowledge from the initial specifications. Listing 7.1 depicts the IDP input model for the personal data and their hierarchy of the camera surveillance scenario.

IDP Listing 7.1: Partial user model representing personal data categories and hierarchies in the video surveillance scenario.

```
type DataCategory = {
        PersData; Face; BlurredFace; Gait; Height; Behavior; Location; Time;
        PictureIncident}

DataCategoryOf(DataCategory, DataCategory) = {
        (Face, PictureIncident); (BlurredFace, PictureIncident);
        (Gait, PictureIncident); (Height, PictureIncident);
        (Behavior, PictureIncident)}
```

7.3.2 Entities

Data subjects and the auditors that act on their behalf are not explicitly modeled since their point of view is external. An arbitrary number of active entities can be modeled in the framework's system model. Active entities are those that handle personal data of the subjects and provide some degree of accountability, i.e. declarations (with or without proof) about the data processing they perform. A distinction is made between *Stakeholders* and *Components*. A stakeholder is either an *Organization*, or an *Operator* acting on behalf of exactly one organization. An organization can employ more than one operator. Components are constituents of data processing systems. A component belongs to exactly one organization, but can be used by multiple operators.

Components process personal data under the responsibility of the organizations that own them. Organizations or authorities may restrict access to the data categories that a given component is capable of collecting. Authorized categories for a given component are specified using *ComponentCanCollect(Component, DataCategory)*. Listings 7.2 and 7.3 depict the IDP specification of the *entities* involved and their *relationships* in the camera surveillance scenario respectively.

IDP Listing 7.2: Partial system model representing the entities in the video surveillance scenario.

```
type Entity = {
        RailwayCompany; SecurityCompany; LegalAuthority; Camera; Monitor;
        MD; SurveilanceGuard; ImageProcessor; StatusProcessor; ImageDB;
        StatusDB}

type Stakeholder isa Entity = {
        RailwayCompany; SecurityCompany; LegalAuthority; SurveilanceGuard;
        ImageProcessor; StatusProcessor}
type Component isa Entity = {
        Camera; Monitor; MD; ImageDB; StatusDB}

type Organization isa Stakeholder = {
        RailwayCompany; SecurityCompany; LegalAuthority}

type Operator isa Stakeholder = {
        SurveilanceGuard; ImageProcessor; StatusProcessor}
```

7.3.3 Statements and Local Accountability Statements

All entities involved in data handling relevant to a given data subject are assumed to exhibit some level of accountability of practice, i.e. they publish precise declarations about their intended personal data handling practices. In general, each entity publishes a different data handling statement. A one-to-one mapping between entities and data handling statements is assumed, and is modeled using function *StatementFrom(Statement) : Entity*. Listings 7.4 and 7.5 show the part of the system model that defines a subset of the *statements* of the *railway company* listed in Table 7.1[1]. Those statements include the following aspects:

- Purposes of use, i.e. the list of finalities for which the collected personal data may be used (for instance statistics or direct marketing) — this is modeled using *StatementPurpose(Statement, Purpose)*. Multiple purposes can be defined for a statement.

[1]For the complete model of the statements, see `https://github.com/inferring-accountability/inferring-accountability/`, 2014.

IDP Listing 7.3: Partial system model representing the entity relationships in the video surveillance scenario.

$$\begin{aligned}&\mathsf{ComponentOf}(\text{Component}) : \text{Organization} = \{\mathit{Camera} \rightarrow \mathit{RailwayCompany};\\ &\quad \mathit{Monitor} \rightarrow \mathit{RailwayCompany}; \mathit{ImageDB} \rightarrow \mathit{RailwayCompany};\\ &\quad \mathit{StatusDB} \rightarrow \mathit{SecurityCompany}; \mathit{MD} \rightarrow \mathit{SecurityCompany}\}\end{aligned}$$

$$\begin{aligned}&\mathsf{EmployeeOf}(\text{Operator}) : \text{Organization} = \{\\ &\quad \mathit{SurveilanceGuard} \rightarrow \mathit{SecurityCompany};\\ &\quad \mathit{StatusProcessor} \rightarrow \mathit{SecurityCompany};\\ &\quad \mathit{ImageProcessor} \rightarrow \mathit{RailwayCompany}\}\end{aligned}$$

$$\begin{aligned}&\mathsf{OperatorOf}(\text{Operator}, \text{Component}) = \{(\mathit{SurveilanceGuard}, \mathit{Monitor});\\ &\quad (\mathit{SurveilanceGuard}, \mathit{MD}); (\mathit{ImageProcessor}, \mathit{ImageDB});\\ &\quad (\mathit{StatusProcessor}, \mathit{StatusDB})\}\end{aligned}$$

$$\begin{aligned}&\mathsf{ComponentCanCollect}(\text{Component}, \text{DataCategory}) = \{(\mathit{Camera}, \mathit{Face});\\ &\quad (\mathit{Camera}, \mathit{BlurredFace}); (\mathit{Camera}, \mathit{PictureIncident}); (\mathit{Camera}, \mathit{Gait});\\ &\quad (\mathit{Camera}, \mathit{Height}); (\mathit{Camera}, \mathit{Behavior}); (\mathit{Monitor}, \mathit{Face});\\ &\quad (\mathit{Monitor}, \mathit{BlurredFace}); (\mathit{Monitor}, \mathit{PictureIncident}); (\mathit{Monitor}, \mathit{Gait});\\ &\quad (\mathit{Monitor}, \mathit{Height}); (\mathit{Monitor}, \mathit{Behavior}); (\mathit{ImageDB}, \mathit{Face});\\ &\quad (\mathit{ImageDB}, \mathit{BlurredFace}); (\mathit{ImageDB}, \mathit{PictureIncident}); (\mathit{ImageDB}, \mathit{Gait});\\ &\quad (\mathit{ImageDB}, \mathit{Height}); (\mathit{ImageDB}, \mathit{Behavior}); (\mathit{ImageDB}, \mathit{Time});\\ &\quad (\mathit{ImageDB}, \mathit{Location}); (\mathit{StatusDB}, \mathit{Time};); (\mathit{StatusDB}, \mathit{Location});\\ &\quad (\mathit{MD}, \mathit{Time}); (\mathit{MD}, \mathit{Location})\}\end{aligned}$$

- The category of personal data that is used, i.e. the collection of personal identifiable information. Possibly, multiple subject data categories exist for a statement — this is modeled using predicate *StatementSubject(Statement, DataCategory).*
- Global retention limits, i.e. the period of time after which the personal data will be deleted by the entity (e.g. 30 days) — this limit is expressed using a partial function (i.e. not every statement expresses a retention limit) *StatementRetentionLimit(Statement) : Duration.*
- *Obligations* built from a *Condition* and an *Action.* These are modeled using partial functions *StatementCondtion(Statement) : Condition* and *StatementAction(Statement) : Action*). Both are partial functions because not all statements are linked to actions (e.g. retention limits),

and unconditional obligations are modeled by only modeling the actions of statements.

- Personal data may be forwarded to organizations. In the model, this is expressed using *StatementDestination(Statement, Organization)*. Possibly, a statement has multiple destinations.

Obligations are flexible and can be used to express a variety of constraints. Conditions are events that trigger a reaction, e.g. the personal data is accessed or the data subject has requested an update. Actions are the resulting events, for instance the update of his personal data or its forwarding.

Statements guaranteeing the sending of a notification (to a user) when a specific event occurs (e.g. when a specific category of personal data is accessed by the entity) are expressed using *StatementNotificationGuarantee(Statement)*.

Accountability occurs at different levels. Some entities may merely declare their intended practices, without providing any companion evidence. Other entities provide data handling logs. In the model this is denoted using function *StatementProof(Statement) : StatementEvidence*. It may not always be possible to check the compliance of data handling logs with obligations. Logs can be in a format which is not standardized, or semantics may not be provided by the entity. Therefore, a distinction is made between three levels of assurance (*StatementEvidence*) for data handling statements:

1. A statement is (purely) *Declarative* if data handling logs relevant to the statement are not made available by the entity publishing the statement.
2. If data handling logs are provided together with the statement but cannot be checked straight away, the statement obtains the status *LoggedUnverified*.
3. If a statement is provided together with logs that have been checked for compliance (e.g. through a trusted log analysis software), the statement is said to be *LoggedVerified*. This is the highest level of accountability for a data handling statement, since actual behavior has both been recorded and shown to be compliant with the statement.

7.3.4 Trust Perception and Global Accountability Inference

While organizations may feature complex hierarchies with heterogeneous data handling practices, individuals care about what happens to their personal data

IDP Listing 7.4: Partial system model containing the domains that represents the statements of the entities involved in the video surveillance scenario.

`type` Statement $= \{StatR1; StatR2; StatR3; StatR4; \ldots\}$

`type` Purpose $= \{Evidence; DetectIncident; Commerce; Identification\}$

`type` Condition $= \{RequestLegalAuthority; NoLegalInvestigation\}$

`type` Action $= \{Collecting; Monitoring; Storing; Forwarding; Accessing\}$

`type` Duration `isa int` $= \{30; 60; 90\}$

`type` Permission `constructed from` $\{Always; Never\}$

globally. A panoramic overview of the worst-case scenario in terms of data processing (i.e. *what are the weakest global guarantees*) is relevant to individuals, since they must often decide whether to interact with an entire organization. Most of the time, they cannot cherry-pick with which subcontractors to share their data with.

Global accountability inference is a central feature of this framework that builds such a synthetic statement for data subjects. It deduces global guarantees from the local accountability statements of all entities involved in the system. These (subjective) guarantees depend on trust perceptions of data subjects.

Individuals display different levels of trust in the entities that handle their personal data. The framework's user model reflects this socio-technical aspect by modeling three levels of trust, corresponding to three typical types of individuals:

- *Naive* individuals always trust data handling statements, even if statements are purely declarative (i.e. no evidence in the form of a log is provided);
- *Regular* individuals only trust statements co-occurring with relevant data handling logs;
- *Privacy-aware* individuals are most skeptical and trust only statements for which verified logs have been provided by issuing entities.

IDP Listing 7.5: Partial system model containing the relations that represents the statements of the entities involved in the video surveillance scenario.

```
type StatementEvidence constructed from {
        Declarative; LoggedUnverified; LoggedVerified}

StatementFrom(Statement) : Entity = {
        StatR1 -> RailwayCompany; StatR2 -> RailwayCompany;
        StatR3 -> RailwayCompany; StatR4 -> RailwayCompany; ...}

StatementSubject(Statement, DataCategory) = {
        (StatR1, Face); (StatR1, BlurredFace); (StatR1, Gait); (StatR1, Height);
        (StatR1, Behavior); (StatR2, PictureIncident);
        (StatR3, PictureIncident); (StatR4, PersData); ...}

StatementPurpose(Statement, Purpose) = {
        (StatR1, DetectIncident); (StatR2, Evidence); (StatR3, Commerce); ...}

partial StatementCondtion(Statement) : Condition = {
        StatR2 -> RequestLegalAuthority; ...}

StatementPermission(Statement) : Permission = {
        StatR1 -> Always; StatR2 -> Always; StatR3 -> Never;
        StatR4 -> Always; ...}

partial StatementAction(Statement) : Action = {
        StatR1 -> Collecting; StatR2 -> Forwarding; StatR3 -> Collecting; ...}

StatementDestination(Statement, Organization) = {
        (StatR2, LegalAuthority); ...}

partial StatementRetentionLimit(Statement) : Duration = {
        StatR4 -> 60; ...}

StatementNotificationGuarantee(Statement) = {   }

StatementProof(Statement) : StatementEvidence = {
        StatR1 -> LoggedUnverified; StatR2 -> Declarative;
        StatR3 -> LoggedUnverified; StatR4 -> LoggedUnverified; ...}
```

Furthermore, the user model includes *UserTrust(Organization)*, the user's high-level trust perception towards organizations. It represents his trust in declared data handling practices of related organizations. This also implies that all operators they employ and components they own are trusted by him. A more advanced trust model could be used. For example, a more advanced trust model could include the user's trust towards loggings. Nevertheless, a binary trust model is used to keep the complexity of models manageable. The modeled video surveillance scenario features the aforementioned three user models: naive (*U1*), regular (*U2*) and privacy-aware (*U3*). It also assumes that no organization is trusted, i.e. *UserTrust(Organization)* is the empty set.

This impact of these user trust models on the perception of global accountability guarantees is shown in Table 7.2. For instance, a *naive user* considers he is guaranteed that merely declared statements of an entity *E*, owned or employed by organization *O*, correspond with actual data handling practices. By contrast, a *regular user* only considers merely declared statements to be guaranteed if he trusts *O* (i.e. *UserTrust(O)*), and assumes statements provided together with logs to be guaranteed, whether these logs are checked for compliance or not.

Table 7.2: Global statement evidence deduction rules — the global evidence for the statement *S* by the entity *E* owned by the organization *O* is *(U)ncertain* or *(G)uaranteed* for the modeled user.

StatementProof(S)=	Declared	Logged-unverified	Logged-and-verified
Naive user	G	G	G
Regular user	$F(E) : \{G, U\}^\star$	G	G
Privacy-aware user	$F(E) : \{G, U\}^\star$	$F(E) : \{G, U\}^\star$	G

$^\star F(E) =' G' \Leftrightarrow \mathit{UserTrust}(O) \wedge (\mathit{ComponentOf}(E) = O \vee \mathit{EmployeeOf}(E) = O)$
$^\star F(E) =' U' \Leftrightarrow \neg \mathit{UserTrust}(O) \wedge (\mathit{ComponentOf}(E) = O \vee \mathit{EmployeeOf}(E) = O)$

Global statement computations are performed differently for *duties* (i.e. statements featuring *Always*) and for *prohibitions* (i.e. statements featuring *Never*). Beside these categories, models also include statements expressing *notification guarantees* and *global retention limits*. Similar to duties, these also feature *Always*. Nevertheless, these are treated differently in computations.

Global statements are expressed in terms of *global data categories* (i.e. users are concerned what happens to their personal data). Let *S* be an individual statement of entity *E*, *CanCollect(E,DC)* a data category *DC* that can be collected by an entity *E*, and *Sub(S,DC)* representing that data category *DC* is a subject of *S*. Table 7.3 summarizes the worst-case deduction rules that depend on the *global statement evidence* for the computation of *GlobalDataCategory(S,DC)*, the global data categories derived from *S*.

Table 7.3: Worst-case computation rules for deducing *GlobalDataCategory(S,DC)*, the global data categories *DC* deduced from the individual statement *S* of entity *E*, with *Sub(S,DC)* the subject *DC* of statement *S*, and $CanCollect(E, DC)$ the data categories collectable by *E*.

Global statement evidence of S:	**Uncertain**	**Guaranteed**
$Duty(S)$	$CanCollect(E, DC)$	$\psi(S, E, DC)^\star$
$Prohibition(S)$	$\psi(S, E, DC)^\star$	$Sub(S, DC)$
$NotificationGuarantee(S)$	$Sub(S, DC)$	$\psi(S, E, DC)^\star$
$RetentionLimit(S)$	$Sub(S, DC)$	$\psi(S, E, DC)^\star$

$^\star\psi(S, E, DC) \equiv CanCollect(E, DC) \wedge Sub(S, DC)$

Duties. Global statements using *Always* are built as follows:

- The *global purposes of use* for a global data category are constructed from the union of all purposes of (individual) duties *S*, with *GlobalDataCategory(S,DC)*. These represent worst-case global purposes which are conjunctive. For instance, personal data is collected for commercial and statistical reasons. If no purpose is explicitly specified, then all purposes are assumed to be permitted globally.

- The *global conditions of use* for a data category are constructed from the disjunction of all conditions of duties *S*, with *GlobalDataCategory(S,DC)*. If at least one unconditional statement exists, no overall conditional statement is generated.

- The *global actions* for a data category are built from the union of all actions of individual duties *S*, with *GlobalDataCategory(S,DC)*.

- The *global level of assurance* (i.e. global evidence) for a data category is *Uncertain* if at least one uncertain statement (in the sense of Table 7.3) exists for this data category. Else, the global statement is considered *Guaranteed*.

- The *global notification guarantee* for events relative to a data category is built from the conjunction of all individual notification guarantees *S* relative to that data category, with *GlobalDataCategory(S,DC)*.

- The *global retention limit* for a data category is the maximum of all retention limts *S* existing for the data category, with *GlobalDataCategory(S,DC).*

Prohibitions. Global statements using *Never* are built as follows:

- The *global purposes of use* for a global data category are constructed from the union of all purposes of individual prohibitions *S*, with *GlobalDataCategory(S,DC).* These represent worst-case global purposes which are disjunctive. For instance, personal data is never collected for commercial or statistical reasons. Individual prohibitions without explicit purpose are omitted during the deduction of global purposes (i.e. worst-case).
- The *global conditions of use* for a data category are constructed from the conjunction of all conditions of prohibitions *S*, with *GlobalDataCategory(S,DC).* Unconditional statements are omitted.
- The *global actions* for a data category is computed as for duties, *mutatis mutandis.*
- The *global level of assurance* for a data category is computed as for duties, *mutatis mutandis.*

In global statements, trust is expressed in a binary way (i.e. *GAPEvidence*): statements are, from the point of view of the data subject, either *Uncertain* or *Guaranteed.* Both global notification guarantees and global retention limits, part of the GAP, are expressed as duties. Global statements present guarantees as a function of categories of personal data. Once global statements have been computed, they are represented as in Listing 7.6. They are categorized as follows:

- *Global duties about collecting personal data* — declaring actions about the use, collection, or storage of personal data.
- *Global duties about distributing personal data* — declaring actions that forward data to external organizations.
- *Global prohibitions for collecting personal data* — expressing that the use, collection, or storage of personal data is forbidden.
- *Global prohibitions for distributing personal data* — forbidding the forwarding of personal data to an organization.

- *Global notification guarantees* — declaring the sending of a notification upon the occurrence of a specific event.
- *Global retention limits* — expressing the time limit after which all categories of personal data must be deleted.

IDP Listing 7.6: Modeling concepts representing the GAP.

```
type GAPEvidence constructed from { Uncertain; Guaranteed }

GAPCollectData(DataCategory)
GAPCollectDataAction(DataCategory, Action)
GAPCollectDataForPurposeOf(DataCategory, Purpose)
GAPCollectDataCondition(DataCategory, Condition)
GAPCollectDataProof(DataCategory, GAPEvidence)

GAPForwardDataTo(DataCategory, Organization)
GAPForwardDataAction(DataCategory, Action)
GAPForwardDataForPurposeOf(DataCategory, Purpose)
GAPForwardDataCondition(DataCategory, Condition)
GAPForwardDataProof(DataCategory, GAPEvidence)

GAPNeverCollectData(DataCategory)
GAPNeverCollectDataForPurposeOf(DataCategory, Purpose)
GAPNeverCollectDataCondition(DataCategory, Condition)
GAPNeverCollectDataProof(DataCategory, GAPEvidence)

GAPNeverForwardDataTo(DataCategory, Organization)
GAPNeverForwardDataForPurposeOf(DataCategory, Purpose)
GAPNeverForwardDataCondition(DataCategory, Condition)
GAPNeverForwardDataProof(DataCategory, GAPEvidence)

GAPNotificationGuarantee(DataCategory)
GAPNotificationGuaranteeCondition(DataCategory, Condition)
GAPNotificationGuaranteeProof(DataCategory, GAPEvidence)

GAPRetentionLimit(DataCategory, Duration)
GAPRetentionLimitCondition(DataCategory, Condition)
GAPRetentionLimitProof(DataCategory, GAPEvidence)
```

7.4 Computation and Evaluation

A possible use of the framework with an IDP realization[2] is illustrated. The realization infers the GAPs of the user models *U1*, *U2*, and *U3* described earlier, representing individuals under video surveillance in a railway station. The model was generated in less than a second on a personal computer. First, the resulting profiles of naive, regular, and privacy-aware data subjects are compared. Next, the statements of the entities are discussed and users are modeled.

7.4.1 Trust-Dependent GAP Inference

First, given the type of user and his trust perception towards organizations, the user's global statement evidence is deduced using the rules of Table 7.2. For instance, *U2* (i.e. regular user) is sufficiently guaranteed that data practices comply with declared ones if they are only logged. Instead, *U3* is satisfied when statements are *logged and verified* by an auditor or just logged in case organizations are trusted by him. This global evidence is then used for the deduction of the GAP using the rules of Table 7.3. The inferred GAPs are summarized in Table 7.4. None of these contain global prohibitions. However, individual statements of entities include two prohibitions (i.e. *R.3* and *I.3*). The reason for this is that worst-case computation rules give priority to global duties containing data categories that are subject of both duties and prohibitions. Semantically, this corresponds to a user who is more concerned about the categories of data that are used rather than about the unused ones.

Global duties for collecting data are perceived differently by *U1*, *U2*, and *U3*. Since *U1* is satisfied with statements that are purely declarative, he believes that data collection duties are respected by the organizations. *U2* is only partially convinced with the same guarantees, since he requires at least data handling logs while the security company's duty *S.1* is purely declarative. Therefore, *Time* and *Location*, subjects of *S.1*, are considered as global duty data categories that are uncertain. *U3* needs the strongest guarantees. He is not convinced for any of the data categories part of the GAP. He expects for all data collection duties that logs exist and that they are verified by an auditor. For instance, because duty *R.1* is *logged-unverified*, the duty subjects, such as *Face* and *BlurredFace*, are not sufficiently guaranteed for *U3*. Furthermore, due to *R.1* an additional data category *PictureIncident* is deduced in *U3*'s GAP.

[2] A detailed IDP realization — together with the output containing the GAPs for the three user models — can be found at `https://github.com/inferring-accountability/inferring-accountability/`.

This follows from the computation rule in Table 7.3 for duties with global evidence that is uncertain for *U3*, and the given railway station's camera capability *ComponentCanCollect(Camera,PictureIncident).* Also, comparing the GAP of *U3* with the others, more purposes for collecting data are deduced. In particular, besides that *BlurredFace* and *Gait* are collected for incident detection (i.e. *DetectIncident*), these are used as *Evidence* of incidents as well.

Global data forward duties are computed from the declarative duty *R.2* and the merely logged duty *S.2*. Both duties declare to forward data to other stakeholders in the system. The results show that *U1* is satisfied with the guarantees provided that the system respects data forwarding declarations. The same guarantees are too weak to convince *U2* and *U3*. Both doubt that actual data practices correspond with declared ones. At first glance, one could expect that *U2* assumes that *Time* and *Location*, part of the GAP, are used as declared by *S.2*. However, *S.2* is redundant with the purely declared duty *R.2* because *Time* and *Location* are subjects of *R.2* as well. This can be deduced using the computation rules of Table 7.3 and from the railway company's image database capabilities, namely *ComponentCanCollect(ImageDB,Time)* and *ComponentCanCollect(ImageDB,Location).*

Global retention limits are computed from *R.4*, *I.4*, and *D.3*. Results show that retention limits are conditional (i.e. *NoLegalInvestigation*) for all data categories in the GAP of *U1* and *U2*. The GAP of *U3* shows an additional unconditional retention limit for data category *PersData* (i.e. personal data). This is deduced from *R.4*, which provided evidence not fulfilling *U3*'s expectations. Indeed, *R.4* is just logged, and not verified. Furthermore, *U2* only has partial guarantees for *Time* and *Location* since evidence of *R.3* sufficiently guarantees him. In case of *U3*, *R.3* provides insufficient evidence. Hence, usage of *Time* and *Location* are not sufficiently guaranteed according to *U3*'s GAP.

7.4.2 Statements Modeling and User Models

The statements presented in the scenario are atomic declarations, i.e. they consist of single actions on subject data categories. Concepts in the framework were defined for modeling atomic statements. However, statements may contain multiple declarations. The concepts defined lack expressiveness for modeling these. These statements must be represented by the atomic parts from which they are composed. For instance, the image database is associated with a declarative statement announcing the storage of personal data of categories *blurred face* and *gait* for a maximum of 30 days and for the purpose of statistics and marketing. This is modeled as (*a*) a *duty*

Table 7.4: Inferred GAP synthesizing global accountability in the camera surveillance system for user models *U1* (1), *U2* (2), and *U3* (3). The numbers in the table indicate the user models for which the statements in the left column, represented relatively to the different data categories, are valid.

	PictureIncident	*Face*	*BlurredFace*	*Gait*	*Height*	*Behavior*	*Time*	*Location*	*PersData*
Global Collection duties									
Actions									
Collecting	3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	
Accessing		1,2,3					1,2,3	1,2,3	
Storing	3	1,2,3	3	3	3	3	1,2,3	1,2,3	
Monitoring	3	3	1,2,3	1,2,3	1,2,3	1,2,3	3	3	
Purposes									
All		1,2,3							
DetectIncident	3		1,2,3	1,2,3	1,2,3	1,2,3	3	3	
Evidence	3		3	3	3	3	1,2,3	1,2,3	
Conditions									
Unconditional	3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	
Guaranteed		1,2	1,2	1,2	1,2	1,2	1	1	
Global Forward duties									
Actions									
Forwarding									
to *LegalAuthority*	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	
Purposes									
Evidence	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	
Conditions									
RequestLegalAuthority	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	
Guaranteed	1	1	1	1	1	1	1	1	
Global Retention Limits									
Duration (days)									
60 days	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3			3
90 days							1,2	1,2	
Conditions									
UnConditional									3
NoLegalInvestigation	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	
Guaranteed	1	1	1	1	1	1	1,2	1,2	

declaring that data categories *blurred face* and *gait* are *stored*, and (*b*) a *retention limit* specifying that data is kept for a maximum of 30 days. In the model, these items correspond to separate elements of the *Statement* domain. Though both statements have the same purposes, these must be expressed separately. In particular, *StatementPurpose(Statement,Purpose)* relates the purposes *statistics* and *marketing* to the *duty* and *retention limit* with *2 statements* × *2 purposes* relations. Decomposing combined statements may imply that statement relations grow combinatorially. Similarly, each *Statement* element must be related to the *Entity "ImageDatabase"*, the *Permission "Always"*, and to the *StatementEvidence "Declarative"*. Furthermore, each duty is related to the *Action "Store"*.

User model. The coarse-grained user categorization facilitates user modeling since modelers only need to specify user types via a single constant, for instance $\mathit{TypeOfUser} = \mathit{NaiveUser}$. The model's user types intuitively represent typical real-world users, determining how data subjects appreciate statements and evidence from organizations. They reflect the fact that skeptical users are more difficult to convince of the compliance of actual data handling with declared practices. The user model also addresses the high-level trust perception of users. Namely, $\mathit{UserTrust}(O)$ expresses that a user trusts the organization O.

Reusing framework components. The framework consists of modular components, making possible isolated changes to one part while leaving the others intact. A given system model (e.g. the railway station camera surveillance scenario) is unaffected when new types of users are introduced. Similarly, if an auditor collects different samples of statement evidence, only changes to the evidence in the statement model are required.

Detecting conflicts. The user model could be extended with user privacy preferences containing prohibitions. This aspect could be used by auditors wanting to verify e.g. whether a system, run by a commercial organization, is not collecting sensitive health information. The individual statements of system entities are another flexible facet. Typically, these statements form a large set of opaque and potentially inconsistent declarations. Automated verification can be added to the system-independent part of the framework for easy conflict detection.

7.5 Conclusions

An accountability inference model is defined and realized using the IDP knowledge base system. Trust perceptions are taken into account to compute global accountability statements from the individual statements made by interacting entities. Different levels of evidence of individual statements are distinguished, influencing the resulting global accountability statements. The approach is illustrated with a scenario involving stakeholders in a railway surveillance infrastructure. The framework is not tied to any particular scenario and can be extended easily. The representation of data handling evidence is only implicit, and therefore coarse-grained. A more refined approach would model the semantics of log compliance explicitly. This level of detail seems difficult to implement within a first-order logic-based framework. In the current version of the framework, the auditor acting on behalf of an individual is not notified of privacy policy conflicts automatically. Including this aspect would remove the need for manual compatibility checking.

Chapter 8

General Conclusion

The scope of this dissertation is the individual's personal information collected in complex electronic services. The goal of this research is twofold. Firstly, it provides end-users with meaningful feedback that clearly informs them about the privacy impact of using the aforementioned services. Secondly, it assists service designers by providing feedback about the impact of their design decisions in regard to the individual's privacy. This chapter reflects on the research scope, the used approach and framework, followed by future research and valorization tracks.

8.1 Reflection on the Research Scope

The class of services considered in this thesis is represented by the service blueprint as presented in Chapter 3. The blueprint comprises a single user who interacts with a service front-end which, in its turn, calls underlying third-party sub-services. The scenarios discussed in Chapters 3, 4, 5 and 6 (i.e. respectively a travel reservation system, a web shop, loyalty systems and a public transport system) show that the blueprint can be used to represent services from different application domains. The considered class of services represents a substantial group of contemporary Internet services, such as entertainment services (e.g. Spotify and Netflix), e-Commerce services (e.g. eBay and Amazon), search engines (e.g. Google), and e-Governance services. The blueprint, however, is restricted to a single user interacting with a service front-end. Extensions to the service blueprint are needed to represent users exchanging personal information with each other via a service.

For instance, a user sharing his personal pictures with his friends on Facebook cannot be represented. Another example are members in a peer-to-peer network sharing fragments of a file that another user can download via the peer-to-peer service. To model these multi-user applications, the user trust model must be extended with additional users (i.e. additional users must be added to the *User* domain) and new trust relationships must be defined so that trust between users can be modeled as well. For example, to capture a privacy-aware user who shares no personal attributes with other less privacy-aware users.

The collection of personal information in electronic services is subjected to privacy regulations. The privacy protection principles in the European General Data Protection Regulations include data minimization and service provider accountability. The first principle mentioned, which focuses on the disclosure of a user's personal attributes, is the basis for the privacy analysis approach as presented in Part I and II of this dissertation. The second principle mentioned, which focuses on the compliance of a service provider's data handling practices, is used in the approach presented in Part III. Both approaches are complementary as services may require to reveal personal attributes to meet the functional requirements. For example, a user must reveal his name and address to get his purchases delivered at home. By keeping service providers accountable for their data handling practices, service providers are enforced to respect data protection regulations and their declarations on data handling practices. Service providers can log their data handling activities to prove their compliance with data handling practice declarations towards privacy auditing authorities. However, logs are no guarantee of compliance. As a consequence, the validity of logs depends on the perception of auditors and individuals whether logs are trustworthy. Nevertheless, it is in the interest of service providers to respect a user's privacy as it is hard to acquire a good reputation.

8.2 Reflection on the Modeling Framework

The framework presented in Part I and II analyzes the user's privacy at service level (i.e. application level), while the framework in Part III is used to provide feedback on data handling practices at system level. The former framework provides privacy-related feedback based on the user's personal attributes collected by services. The latter provides global feedback on a system's data handling practices, such as the purposes and conditions for collecting a user's personal attributes and the period during which personal attributes are stored. In both frameworks, data handling practices are derived from publicly available privacy policies. However, practical restrictions complicate the modeling of these policies. Typically, these policies are written in human language. The translation into an IDP specification is difficult. One of the reasons is that service policies can be interpreted in different ways as they are often unclear. Privacy regulators could require to unambiguously specify service policies in a machine interpretable format (see Section 2.2.3) so that the translation into IDP input model specifications could be automated.

Privacy regulations, such as the EU general data protection regulation, require that service providers integrate privacy safeguards into their design process and that service providers can demonstrate compliance with data handling practice declarations and privacy regulations. In this context, both frameworks are relevant, even in cases where privacy properties of services are trivial to derive manually. Moreover, formal modeling frameworks could become essential for service providers, for instance, to be awarded with a privacy seal indicating their data handling practices comply with the considered privacy protection framework.

Appendices C and D show the IDP realization of the privacy analysis framework as presented in Part I, the input model that represents the web shop case of Section 4.1, and the output model containing the generated privacy feedback of the modeled web shop. The IDP listing separates generic modeling parts from the parts that are specific to the modeled system and user. This supports the *reusability* of the framework components. This key requirement facilitates comparing alternative systems, different credential technologies, and users with different trust perceptions since changes specific to a modeled system and user are isolated to only the system and user model, respectively, part of the input model. More specifically, the framework contains a system independent part (i.e. theory part) that describes generic properties of services. The privacy in a system can be analyzed for different user types by only changing the user model. This enables a modeler to easily compare the acceptance of the modeled service for different user types. The user model represents the user's trust perceptions. To keep the complexity in the models manageable, trust perceptions are represented by binary trust relations (i.e. an organization can be trusted or untrusted) expressing the user's trust in organizations for storing and forwarding his personal attributes. However, binary trust relationships may be too coarse-grained to model user types more accurately or to capture specific situations. An example of a situation in which binary trust representations lack expressiveness, is in the case of the QR code scanner used in the first loyalty scheme as presented in Section 5.1. In the loyalty scheme, the cashier scans the QR code printed on the customer's loyalty card. Although the scanner is only capable of reading the QR code in reality (i.e. limited by the hardware), in the model it is implicitly assumed that all personal attributes printed on the card are collected as well. The difference between reality and what is modeled leads, in case of untrusted organizations, to user profiles containing personal attributes that in reality cannot be collected by the QR code scanner. Hence, additional relations must be defined for capturing situations in which binary trust relations lack expressiveness. For example, concepts must be defined to express a component's data collection limitations.

To guarantee the quality of output models, it must be verified how well the privacy feedback in the output model corresponds to reality. Since output models are inferred from an input model using logic rules that comprise the theory, the quality of output models depend on how exact an input model and theory are. To assure that the privacy feedback correctly represents real-world privacy properties of a modeled system, two conditions must be fulfilled. Firstly, the input model must exactly correspond to the modeled system and user. Secondly, the theory must correctly describe which personal attributes of the user are collected and stored, and how they flow through the modeled system. However, the size of the IDP listing as presented in Appendices C and D clearly illustrate the complexity of the framework and the input and output model. As a consequence, it is infeasible to formally proof that the privacy feedback in the output model correctly represents real-world privacy properties. A pragmatic model-verification based approach is used for verifying the quality of the models. Firstly, a theory must be defined for automatically detecting inconsistent input model specifications. However, this insufficiently guarantees whether specifications correctly represent the modeled system. The remedy is to enforce service providers to specify their privacy policies in a machine interpretable policy language such as P3P. This supports to automatically generate input models that correspond with the modeled system. Secondly, an indirect verification approach is used for verifying the theory that describes the generic behavior and inference strategy. A set of invariants (i.e. logic rules that in all models are true), representing the properties inherent in composite services, can be verified using a set of different output models to verify whether the theory is correct. If the theory is correct, invariants will evaluate to true. The more invariants and output models verified, the more assured a modeler can be that the theory is correct. An invariant is given in the following example. Given the ordered set of services $s_0 <_G s_1 <_G s_2 <_G \ldots$ in which the services collect personal attributes forwarded by their predecessor. If an untrusted organization owns a user profile P_i associated to a service s_i part of the above-mentioned set, then $P_j \subseteq P_i$ holds for all user profiles P_j associated with services s_j, with $s_i <_G s_j$.

The aforementioned verifications (i.e. input model consistency and output model verification) are two examples of *model-checking problems*, supported by the IDP system. Model checking is computationally feasible as it is a polynomial problem in function of the size of the input or output model. The privacy analysis of a modeled system uses *finite model generation*. By default, IDP generates a finite output model by expanding the input model (i.e. a structure containing partial knowledge) so that the output model satisfies the modeling theory. This is a NP problem in the size of the input model that even for medium-sized systems, such as the privacy-preserving public transport ticketing system as presented in Chapter 6, might be infeasible to compute. The computational feasibility also depends on the information that is known and the feedback that must be generated. If all information is known, so that defined predicates can be pre-calculated, then IDP uses the computational more feasible PROLOG based (i.e. XSB) technique [108, 152] to generate privacy feedback. The following example illustrates when XSB and model expansion is used by IDP. In the first case, a set of collaborating organizations is known beforehand. The modeler queries the model for the user's anonymity level. IDP uses XSB to compute the anonymity level as all the information of the system is given. In the second case, a set of collaborating organizations must be found that satisfies the constraint that the user is identifiable towards a specific organization. This is a search problem in which a set of collaborating organizations must be found that satisfies the aforementioned constraint.

In Part II, a linear time model is added to the framework to represent a user who interacts with multiple service front-ends. Alternatively, time can be implicit by using predicates representing the relative order in which services are invoked. For example, predicate $ServiceInvokedNext(ServiceIdentifier, ServiceIdentifier)$ expresses that the user invokes the service referred by the second argument after he invoked the service referred by the first element. However, this lacks expressiveness in the following situations. Firstly, a single service front-end x can be invoked multiple times by the user. In this situation, the predicate cannot represent a service y which is invoked by the user after he invoked service x the first time. Secondly, credential ownership is dynamic as credentials can be issued or revoked at specific time points as a consequence of service invocations. Therefore, credential ownership and service invocations must be expressed towards a shared time reference so that both can easily be related to each other.

McCarthy and Hayes [123] showed that it is hard to represent in logic what is not affected by an action. This is known as the *frame problem* for which Reiter [149] presented a set of solutions fulfilled by the modeling decisions made in Chapter 6. Firstly, services invoked by a user are modeled as actions that cannot be invoked simultaneously. Secondly, the invocation of sub-services are represented as inertial fluents (i.e. sub-services can only be invoked as a causation of a previously invoked service). Finally, credential ownership is also represented by inertial fluents (i.e. they are the result of services which execute).

8.3 Reflection on the Modeling Approach

Expressiveness. A logic-based modeling approach is selected to automate the analysis of privacy in electronic services. The IDP system is selected for the realization of the privacy analysis framework because of its intuitive modeling features. Since it is a declarative language based on first-order logic, it can easily be adopted. Moreover, the IDP language enables modelers to concentrate solely on describing services and not on how solutions are computed. The IDP language extends its expressiveness by supporting *inductive definitions.* Definitions are superior to constraint rules for expressing composite services. The former are composed of construction rules that describe how objects are built; the latter only describe what is excluded from the solution. Hence, composite services can be better described by means of construction rules rather than with constraints. This is acknowledged by the IDP listing in Appendix C in which the theory is completely described using inductive definitions. Moreover, inductive definitions are necessary for expressing the transitive closure since first-order logic lacks expressiveness. The transitive closure property is used for describing how personal attributes flow through composite services. Although the IDP language is supposed to be intuitive, still its expressiveness is limited as it is based on first-order logic. For example, first-order logic cannot directly represent a set containing different elements. A modeler must first define a reference associated with the set and then define a predicate to relate each member element to this reference.

Capabilities. The IDP system supports non-monotonic reasoning on a knowledge base containing incomplete knowledge (i.e. an input model in which not all facts are given). The IDP system generates output models (i.e. solutions) that are supersets of the input models. Besides it can be used to extract privacy properties from a given input model that completely specifies a modeled system, it can also solve *search problems*, in which parts of the input specification is missing. Model-checking is computational more feasible than search problems. In the former case, it is likely that the IDP system can check models using XSB to pre-calculate definitions (see Section 8.2). In the latter case, the IDP system uses model expansion for computing solutions which often is infeasible. If the computation is infeasible, then the modeler can perform the following steps. Firstly, try to pre-calculate definitions for which all the information is known. A set of hints that describes a partial solution can be added to the input model as well. Secondly, try to split the theory into independent parts so that the model generation can be performed in separate steps. The intermediate output models generated in one step, are used as input in the next step. A step-by-step approach can reduce the number of unknown variables so that the search space is limited and solutions can be computed with less memory. However, this technique cannot be used in case of search problems, such as finding a set of collaborating organizations so that the user is identifiable towards a specific organization. The reason is that the computation of the set of collaborations cannot be performed without considering the user who must be identifiable towards the specific organization. A solution is to first generate all possible sets of collaborations and then verify for which set the user is identifiable towards the specific organization. However, in some cases this may be infeasible as well since there might exist too many combinations.

8.4 Future Research Tracks

The framework as presented in Part I and II supports a single user who interacts with a service front-end. Future research can extend this to multiple users to support the privacy analysis of users in social networks or peer-to-peer networks. In the current framework, the user model represents the user's initial state (i.e. the credentials he owns) and his trust perceptions (i.e. storage and distribution trust). The user model must be adapted to support multiple users. The following example illustrates potential trust relationships between users who should be considered in the user model. A privacy-aware user only shares information with friends in his social network. His friends may be less concerned about their privacy and possibly leak information to other parties (e.g. organizations or persons that are not part of the social network of the sender) that should not receive it. Hence, trust relationships between users must be added to the user model since the user must trust that his friends will not reveal his information to others.

The verification of compliance between the designed services and legislative privacy frameworks is currently not supported in the modeling framework. The IDP language, which is based on first-order logic, lacks expressiveness to completely represent rules part of privacy regulation frameworks (e.g. HIPAA). It is shown that formal methods [25, 72] using temporal logics (e.g. CTL) should be used to represent the rules part of legislative privacy frameworks. Secondly, since different privacy protection frameworks exist, for each privacy protection framework an associated theory part should be added to the modeling framework (i.e. the logic rules in the theory describe the rules in the privacy protection framework).

8.5 Future Valorization Tracks

This section presents a possible use case in which the modeling framework as presented in Part I and II can be useful.

In case of service designers, the modeling framework is integrated in an automated support tool to facilitate the design of electronic services. The tool automatically warns designers if their decisions violate the designer's predefined privacy requirements. The designer must therefore create IDP input models of the services they design. This is an elaborative and error-prone task. Moreover, the output models, which contain the privacy-related feedback, are hard to interpret by non-experts. Hence, the tool should support a more intuitive representation, such as a graphical formalism, that can be easily interpreted by non-expert modelers. Transformation rules must be defined that automatically transform a graphical input model into the corresponding IDP input model and an IDP output model into a graphical output model.

Appendix A

Inductive Definitions

A definition comprises a set of constructive rules defining a concept, e.g. a first-order logic predicate, in terms of other concepts. Inductive definitions extend definitions by supporting recursion. An example is given by (A.1) defining the ancestor of a person x. The inductive definition contains two rules. The first rule expresses that x's parents are his ancestors. The second rule – the recursion – specifies that the ancestors of x' ancestors are his ancestors as well (transitive closure).

$$\begin{cases} \forall x \forall y \big(AncestorOf(y,x) \leftarrow ParentOf(y,x) \big). \\ \forall x \forall y \big(AncestorOf(y,x) \leftarrow \\ \qquad\qquad \exists z (AncestorOf(y,z) \land AncestorOf(z,x) \big). \end{cases} \tag{A.1}$$

Definitions and constraint rules are semantically opposites. Let P be the set of relations represented by predicate P. A definition defining predicate P builds the set of relations from the bottom up. More specifically, P is constructed from the union of the results of each rule in the definition. Constraint rules, on the contrary, restrict the possible relations represented by P'. Each constraint removes some relations of P', and finally results in $P \subseteq P'$. Semantically, the constructive property of definitions are more convenient than constraint rules for modeling electronic services, which are systems constructed of multiple components.

Appendix B

Query Types

Table B.1 presents the queries for analyzing the privacy of composite services. These queries are applied to the loyalty scheme models of Chapter 5. The queries are classified according to multiple categories.

- *Q1* queries the user's anonymity level.
- *Q2* queries the attributes that are released to organizations.
- *Q3* queries the impact of underlying collaborations between organizations.
- *Q4* queries the trust relationships that are required between organizations.

Table B.1: Queries used for the privacy analysis of the loyalty schemes of Chapter 5.

Q1. Query for linkabilities

r1.1 $\forall s \forall o(\mathsf{AnonService}(s,o) \lor \mathsf{PseudoService}(s,o) \lor \mathsf{IdentService}(s,o) \Leftrightarrow \mathsf{UserInvokable}(s))$

r1.2 $\forall s \forall o(\exists id \exists a(\mathsf{UserProfile}(o,id,s,a) \land \mathsf{Identity}(id)) \Leftrightarrow \mathsf{IdentService}(s,o))$

r1.3 $\forall s \forall o(\exists id \exists a(\mathsf{UserProfile}(o,id,s,a) \land \mathsf{Pseudonym}(id)) \Leftarrow \mathsf{PseudoService}(s,o))$

r1.4 $\forall s \forall o \forall id \forall a(\mathsf{UserProfile}(o,id,s,a) \land \mathsf{Identity}(id) \Rightarrow \neg\mathsf{PseudoService}(s,o))$

r1.5 $\forall o \forall id \forall s \forall a(\mathsf{UserProfile}(o,id,s,a) \land (\mathsf{Identity}(id) \lor \mathsf{Pseudonym}(id)) \Rightarrow \neg\mathsf{AnonService}(s,o))$

Q1.1 find **AnonService**, **PseudoService** and **IdentService**

Q2. Query for the presence of attributes in user profiles

r2.1 $\forall o \forall id \forall s(\mathsf{UserProfileMatch}(o,id,s) \Leftrightarrow \forall a(\neg\mathsf{SearchSet}(a) \lor \mathsf{UserProfile}(o,id,s,a)))$

r2.2 $\forall o \forall id \forall s \forall a(\mathsf{UserProfileViolation}(o,id,s,a) \Leftrightarrow (\mathsf{UserProfile}(o,id,s,a) \land \mathsf{ViolationSet}(o,a)))$

Q2.1 find **UserProfileMatch**
e.g., for $\mathsf{SearchSet} = \{Address, Products, Name, CustomerID\}$

Q2.2 find **UserProfileViolation**
e.g., for $\mathsf{ViolationSet} = \{(AD, Name), (AD, Address), (AD, Email)\}$

Q3. Query for collaborations

r3.1 $\forall o_{from} \forall o_{to} \forall id \forall s \forall a(\mathsf{UserProfile}(o_{from}, id, s, a) \land \mathsf{ShareDataWith}(o_{from}, o_{to}) \Rightarrow \mathsf{UserProfile}(o_{to}, id, s, a))$

Q3.1 apply *Q1.1*, *Q2.1* or *Q2.2*
e.g., for $\mathsf{ShareDataWith} = \{(LP, GS), \ldots\}$

Q3.2 find **ShareDataWith**, with $[minimize(\#\{\forall o_{from}, o_{to} : \mathsf{ShareDataWith}(o_{from}, o_{to})\}]$
such that $\mathsf{ShareDataWith}(o_m, o_n), \ldots, \mathsf{ShareDataWith}(o_r, o_s)$
$\models R \land Req_{Ident} \land Req_{Coll}$, with R the theory

$$\text{e.g., for } Req_{Ident} = \begin{cases} \mathsf{IsIdentService} = \{(Checkout, GS)\} \\ \mathsf{IsNotAnonService} = \{(Checkout, LA)\} \end{cases}$$

$$\text{and for } Req_{Coll} = \begin{cases} \mathsf{InclDataShareWith} = \{(LP, GS)\} \\ \mathsf{ExclDataShareWith} = \{(GS, LA), (LA, GS), (LA, LP), (LP, LA)\} \end{cases}$$

Q4. Query for trust relationships

$$r4.1 \begin{cases} \forall o_{ref}(\mathsf{Trusts}(o_{ref}, User) \leftarrow \exists s \exists p \exists a(\mathsf{TrustRelationShipOf}(o_{ref}) \land \mathsf{ServiceOf}(s) = o_{ref} \\ \qquad \land \mathsf{AccessPolicy}(s) = p \land \mathsf{RevealAttribute}(p, a))) \\ \forall o_{ref} \forall o(\mathsf{Trusts}(o_{ref}, o) \leftarrow \exists s \exists p \exists a(\mathsf{TrustRelationShipOf}(o_{ref}) \land \mathsf{ServiceOf}(s) = o_{ref} \\ \qquad \land \mathsf{AttributeDataFlow}(s_x, s, a) \land \mathsf{ServiceOf}(s_x) = o)) \end{cases}$$

with AttributeDataFlow defined as,

$$r4.2 \begin{cases} \forall s_x \forall s_y \forall a(\mathsf{AttributeDataFlow}(s_x, s_y, a) \leftarrow \exists p_x \exists p_y(\mathsf{AccessPolicy}(s_y) = p_y) \\ \qquad \land \mathsf{RevealAttribute}(p_y, a) \\ \qquad \land \mathsf{DistributionPolicy}(s_x) = p_x \\ \qquad \land \mathsf{ForwardAttributeTo}(p_x, a, s_y))) \\ \forall s_x \forall s_y \forall a(\mathsf{AttributeDataFlow}(s_x, s_y, a) \leftarrow \exists s_z \exists p_x \exists p_y(\mathsf{AttributeDataFlow}(s_z, s_y, a) \\ \qquad \land \mathsf{DistributionPolicy}(s_x) = p_x \\ \qquad \land \mathsf{ForwardAttributeTo}(p_x, a, s_z))) \end{cases}$$

Q4.1 find **Trusts**
e.g., for $\mathsf{TrustRelationshipsOf} = \{LP\}$

Appendix C

IDP realization of the Logic Based Framework

This appendix presents the listing of the IDP realization of the framework as presented in Part I[1]. The listing is structured as follows:

- The vocabulary *voc_privacy* (see Listing C.1) is a generic part that contains all the symbols part of the lexicon in which models can be expressed. The vocabulary can be subdivided into the *general use symbols* section, the *user model* section, the *identifier model* section, the *system model* section and the *internal symbols* section that contains the symbols that are internally used in the theory part.
- The *theory* (see Listing C.2) is a generic part composed of multiple subtheories. The theory part *theo_general* defines the predicates that are generally used in other theory parts. The theory part *theo_behavior* contains the logic rules that describe the generic behavior of the modeled credential technologies. For example, it contains the rule that describes that all attributes of an X.509 certificate are revealed when showing the certificate. The theory part *theo_build_profiles* describes how user profiles are inferred.
- The structure *struct_input_model* (see Listing C.3) corresponds to the input model representing the web shop case of Section 4.1 and is specific to the modeled system and user. Hence, the structure is subdivided into

[1]The IDP realization of the framework is publicly available at `https://github.com/decroik/inspect-privacy-and-trust/`.

the *user model* section, the *system model* section, and the *identifier model* section.

- The *main procedure* is a script (Lua) specifying the required steps for generating the output models.

The aforementioned modeling parts separate the listing into a generic part and parts that are specific to a system or user. It enables modelers to conveniently compare alternative systems, the impact of using different credential technologies, and users with different trust perceptions. For instance, changes are isolated to the *system model* section in the structure *struct_input_model* if the privacy in two alternative systems is compared. If users with different trust perceptions must be modeled, then only the *user model* section in the structure *struct_input_model* must be changed.

Listing C.1: Listing of the vocabulary part of the IDP realization of the framework as presented in Part I

```
vocabulary voc_privacy {

  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //  Vocabulary : General symbols
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  type AttrSrc                        //  The source of an personal attribute

  type Stakeholder isa AttrSrc        // System stakeholders
  type User isa Stakeholder           // The user
  type Organization isa Stakeholder   // The organizations

  type Attribute                      // Personal attributes
  type Property isa Attribute         // Personal properties

  type Credential isa AttrSrc         // Authentication tokens,
  // e.g. X.509 certificates


  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //  Vocabulary : Input model - User Model
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  // The user's trust perceptions
  // (1) : The set of storage trusted organizations
  // (2) : The set of distribution trusted organizations
  StorageTrust(Organization)
  DistributionTrust(Organization)

  // The user's initial state
  type X509 isa Credential
  type Idemix isa Credential

  // A credential is a container of personal attributes
  CredAttr(Credential, Attribute)

  // A credential is issued by exactly one organization
  CredIssuer(Credential) : Organization


  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // Vocabulary : Input model - Identifier Model
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  type Identifier                           // Identifiers
  type Identity isa Identifier                // Identities
  type Pseudonym isa Identifier               // Pseudonyms

  IdentifierSetAttr(Identifier, Attribute)    // The set of attributes of
  // an identifier is composed


  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // Vocabulary : Input model - System Model
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  type ServiceType                              // The services in the system
  type ServiceIdentifier isa AttrSrc            // The identifiers referring
  // to the modeled services

  // A service provided by an organization is referred in the model by exactly
  // one identifier
  partial Service(ServiceType, Organization) : ServiceIdentifier

  // The services that are part of the same session.  It is assumed that the
  // services in a session share all the attributes they uses with all other
  // services that are part of the same session
  SessionMember(SessionIdentifier, ServiceIdentifier)

  // Service policies are comprised of:
  type AccessPolicyId                           // The access policies
  type StoragePolicyId                          // The storage policies
  type DistributionPolicyId                     // The distribution policies
  type OutputPolicyId                           // The output policies

  // Modeling the service policy statements associated with a service
  // (1) : The credentials necessary to access the associated services
  // (2) : The attributes that must be revealed from the attribute source and
  //       that are asserted by a stakeholder
  // (3) : The properties that must be revealed from the attribute source and
  //       that are asserted by a stakeholder
  // (4) : The attributes generated by a service
  // (5) : The attributes which are asserted by a stakeholder that are stored
  // (6) : The attributes which are asserted by a stakeholder that are forwarded
  //       to a service
  // (7) : The attributes that are the output of the associated service
  // (8) : The credentials that are the output of the associated service
  OwnCredential(AccessPolicyId,Credential)
  RevealAttr(AccessPolicyId, AttrSrc, Attribute, Stakeholder)
  ProveProperty(AccessPolicy, Property, AttrSrc, Attribute, Stakeholder)
  GenerateAttr(AccessPolicyId, Attribute)
  StoreAttr(StoragePolicyId, Attribute, Stakeholder)
  DistrAttrTo(DistributionPolicyId, ServiceIdentifier, Attribute, Stakeholder)
  OutputAttribute(OutputPolicyId, Attribute)
  OutputCred(OutputPolicyId, Credential)

  // A service policy is assigned to each service.  Each service
  // is associated with exactly one:
  // (1) : access policy
  // (2) : storage policy
  // (3) : distribution policy
  // (4) : output policy
  // The representations below result in a more efficient computation than
  // ServicePolicy(ServiceIdentifier, AccessPolicyId, StoragePolicyId,
  //               DistributionPolicyId, OutputPolicyId)
  partial AccessPolicy(ServiceIdentifier) : AccessPolicyId
  partial StoragePolicy(ServiceIdentifier) : StoragePolicyId
  partial DistributionPolicy(ServiceIdentifier) : DistributionPolicyId
  partial OutputPolicy(ServiceIdentifier) : OutputPolicyId


  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //  Vocabulary : Internal symbols
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // The predicates below are helper predicates and are only used internally
  // The relations can be deduced from a set of given relations.

  // An attribute that is part of the set of attributes that represent an
  // identifier
  IsIdentifierSetAttr(Identifier,Attribute)

  // The storage and distribution trust in services that are deduced from the
  // set of storage and distribution trusted organizations respectively
  StorageTrustServInstance(ServiceIdentifier)
  DistributionTrustServInstance(ServiceIdentifier)
```

```

  // A service directly invoked by a user
  ServiceInvokedByUser(ServiceIdentifier)

  // The successor of a service (1st argument) is the service (2nd argument)
  // that is invoked by it
  Successor(ServiceIdentifier, ServiceIdentifier)

  // The service (1st argument) that is invoked by the service represented by
  // the second argument
  ServiceInvokedBy(ServiceIdentifier, ServiceIdentifier)

  // The attributes generated by a service are revealed to it
  GeneratedByService(ServiceIdentifier, Attribute, Stakeholder)

  // The attributes revealed to a service that is invoked
  RevealedToService(ServiceIdentifier, Attribute, Stakeholder)

  // Attributes forwarded by a service (1st argument) to the successor (2nd
  // argument).
  ForwardedByServiceToSucc(ServiceIdentifier, ServiceIdentifier, Attribute,
  Stakeholder)

  // The attributes that according to the storage policy are stored when using
  // a service
  ServiceStoredPolicyAttr(ServiceIdentifier, Attribute, Stakeholder)

  // The attributes that can be stored because they are revealed towards the
  // service
  ServiceStoredRevealAttr(ServiceIdentifier, Attribute, Stakeholder)

  // A path between two services in which no distribution trust is present
  UntrustedPath(ServiceIdentifier, ServiceIdentifier)

  // Attributes that are distributes to a service (1st argument) by preceding
  // services (2nd argument)
  ServiceStoredForwardAttr(ServiceIdentifier, ServiceIdentifier, Attribute,
  Stakeholder)

  // The attributes (asserted by a stakeholder) of a sub-profile associated with
  // a service (1st argument) that is a sub-service of the service represented
  // by the 2nd argument
  SubProfile(ServiceIdentifier, ServiceIdentifier, Attribute, Stakeholder)

  // The identifier to which the above-mentioned sub-profiles can be linked
  SubProfileId(ServiceIdentifier, ServiceIdentifier, Identifier)


  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // Vocabulary : Output symbols
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  // An organization's user profile associated with an invoked service is the
  // collection of stored attributes that are asserted by a stakeholder and that
  // can be linked to a common identifier.
  UserProfile(Organization,Identifier,ServiceIdentifier,Attribute,Stakeholder)
}
```

Listing C.2: Listing of the theory part of the IDP realization of the framework as presented in Part I

```
theory theo_general : voc_privacy {

  // The user's storage trust into a service is derived from the set of storage
  // trusted organizations
  {
    ! si : StorageTrustServInstance(si) <- ? st[ServiceType] o :
           Service(st,o) = si &
           StorageTrust(o).
  }

  // The user's distribution trust into a service is derived from the set of
  // distribution trusted organizations
  {
    ! si : DistributionTrustServInstance(si) <- ? st[ServiceType] o :
           Service(st,o) = si &
           DistributionTrust(o).
  }

  // The attributes that are generated by a service are revealed to it as well
  {
    ! si a e : GeneratedByService(si,a,e) <- ? ap s[ServiceType] :
            AccessPolicy(si) = ap &
            GenerateAttr(ap,a) &
            Service(s,e) = si.
  }

  // Linkabilities
  //  (1) : Pseudonyms
  //  (2) : Identities
  {
    ! id a : IsIdentifierSetAttr(id,a) <- Pseudonym(id) &
            IdentifierSetAttr(id,a).

    ! id a : IsIdentifierSetAttr(id,a) <- Identity(id) &
            IdentifierSetAttr(id,a).
  }
}

theory theo_behavior : voc_privacy {

  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // Theory : Generic behavior of credential technologies
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  // The attributes that are revealed to a service
  {
    //  (1)   : The attributes revealed to a service according to the associated
    //          access policies
    //  (2)   : Attributes that are generated by the service are revealed to
    //          the service.
    //  (3)   : The Session identifier is a linkable attribute that associates
    //          all the attributes that are revealed to the services that are
    //          part of the same session
    //  (4)   : The properties revealed to a service according to the associated
    //          access policies

    ! si a e : RevealedToService(si,a,e) <- ? ap cred :
    AccessPolicy(si) = ap &
    RevealAttr(ap,cred,a,e).

    ! si a e : RevealedToService(si,a,e) <- GeneratedByService(si,a,e).

    ! si a e : RevealedToService(si,a,e) <-
            ? s[ServiceType] sessid[SessionIdentifier] :
                SessionMember(sessid,si) & Service(s,e) = si &
                IdentifierSetAttr(sessid,a).

    ! si prop e : RevealedToService(si,prop,e) <- ? ap sp dp op as a :
            ServicePolicies(si,ap,sp,dp,op) &
            ProveProperty(ap,prop,as,a,e).

    // Technology dependent part: certificates
    //  (1)   : X.509 - all attributes are revealed when showing an X.509
    //                  certificate
    //  (2)   : X.509 - non-selective attribute disclosure property - redundant
    //                  with (1).
    //  (3)   : X.509 - all attributes are revealed when proving a property -
    //                  redundant with (1)
    ! si a e : RevealedToService(si,a,e) <- ? cred[X509] ap :
            AccessPolicy(si) = ap &
            OwnCredential(ap,cred) &
            CredAttr(cred,a) &
            CredIssuer(cred) = e.

    //    Idemix - no additional rules
  }


theory theo_build_profiles : voc_privacy {

  // A service (2nd argument) is a successor if it is invoked by the preceding
  // service (1st argument).  The successors be deduced from the distribution
  // policy. Multiple successors can exist for a single predecessor.
  {
    ! si succ : Successor(si,succ) <- ? dp a e :
            DistributionPolicy(si) = dp &
            DistrAttrTo(dp,succ,a,e).
  }

  // A service is invoked by a user if it is not a successor.
```

```
{
  ! si : ServiceInvokedByUser(si) <- ~(? pred : Successor(pred,si)).
}

// Service are invoked by a preceding service
//  (1)   : A service (2nd argument) invokes its successor(s) (1st argument)
//  (2)   : A service (1st arugment) that is indirectly invoked by a preceding
//          service.
{
  ! succ si : ServiceInvokedBy(succ,si) <- Successor(si,succ).

  ! si_2 si_0 : ServiceInvokedBy(si_2,si_0) <- ? si_1 :
          ServiceInvokedBy(si_1,si_0) &
          ServiceInvokedBy(si_2,si_1).
}

// Attributes that are stored as declared in the storage policy declarations
{
  ! si a e : ServiceStoredPolicyAttr(si,a,e) <- ? sp :
          StoragePolicy(si) = sp &
          StoreAttr(sp,a,e).
}

// In case of no storage trust, all the attributes revealed are assumed to be
// stored by the service.
{
  ! si a e : ServiceStoredRevealAttr(si,a,e) <- RevealedToService(si,a,e).
}

// The attributes forwarded to a successor (2nd argument) by an
// invoked service (1st argument) are :
//    (1)   : the attributes that are forwarded as declared in the
//            distribution policies
//    (2)   : the attributes that can be collected by a service are all
//            forwarded to the successors in case the user has no distribution
//            trust
{
  ! si succ a e : ForwardedByServiceToSucc(si,succ,a,e) <- ? dp :
          DistributionTrustServInstance(si) &
          DistributionPolicy(si) = dp &
          DistrAttrTo(dp,succ,a,e).

  ! si succ a e : ForwardedByServiceToSucc(si,succ,a,e) <- ? dp ax ex :
          ~DistributionTrustServInstance(si) &
          DistributionPolicy(si) = dp &
          DistrAttrTo(dp,succ,ax,ex) &
          RevealedToService(si,a,e).
}

// A path between two services in which no distribution trust is present
{
  ! start succ : UntrustedPath(start,succ) <-
          Successor(start,succ) &
          ~DistributionTrustServInstance(succ).

  ! start si : UntrustedPath(start,si) <- ? si_x :
          UntrustedPath(start,si_x) & Successor(si_x,si) &
          ~DistributionTrustServInstance(si_x).
}

// The attributes forwarded by sub-services as a causation of a user invoked
// service (2nd argument) to a sub-service (2nd argument) are stored.  It is
// assumed that the organization is not storage trusted by the user.
//  (1)   : if the service (1st argument) is a successor of a user
//          invoked service (2nd argument)
//  (2)   : if the service (1st argument) is a sub-service of a user invoked
//          service
//  (3)   : if the service (1st argument) is a sub-service of a user invoked
//          service (2nd service) that forwards attributes in the case of an
//          untrusted path
//  (4)   : if the service (1st argument) is a sub-service that receives
//          attributes from other sub-services of a common user invoked
//          service (2nd argument)
{
  ! si su a e : ServiceStoredForwardAttr(si,su,a,e) <-
          ServiceInvokedByUser(su) &
          ForwardedByServiceToSucc(su,si,a,e).

  ! si su a e : ServiceStoredForwardAttr(si,su,a,e) <- ? src :
          ServiceInvokedByUser(su) &
          ServiceInvokedBy(src,su) &
          ForwardedByServiceToSucc(src,si,a,e).

  ! si su a e : ServiceStoredForwardAttr(si,su,a,e) <- ? succ :
```

```
            ServiceInvokedByUser(su) &
            UntrustedPath(su,si) &
            ServiceInvokedBy(si,succ) &
            ForwardedByServiceToSucc(su,succ,a,e).

    ! si su a e : ServiceStoredForwardAttr(si,su,a,e) <- ? src succ :
            ServiceInvokedByUser(su) &
            UntrustedPath(src,si) &
            ServiceInvokedBy(src,su) &
            ServiceInvokedBy(si,succ) &
            ForwardedByServiceToSucc(src,succ,a,e).
  }

  // Build sub-profiles containing the attributes collected from a sub-service
  // (1st argument) of a user invoked service (2nd argument).  Three
  // cases are considered:
  //  (1)   : the sub-profiles contain the attributes that are specified in
  //          the storage policies associated with a user invoked service
  //  (2)   : the sub-profiles contain the attributes revealed by a user to a
  //          user invoked service in the case there is no storage trust
  //  (3)   : the sub-profiles contain the attributes that are mentioned in
  //          the storage policy of the associated sub-service
  //  (4)   : attributes revealed by the user (if no storage trust) to a
  //          sub-services
  //  (5)   : attributes that are forwarded to a sub-service by a preceding
  //          sub-service in the case the user has no storage trust
  {
    ! su a e :    SubProfile(su,su,a,e) <- ServiceInvokedByUser(su) &
            ServiceStoredPolicyAttr(su,a,e).

    ! su a e :    SubProfile(su,su,a,e) <- ~StorageTrustServInstance(su) &
            ServiceInvokedByUser(su) &
            ServiceStoredRevealAttr(su,a,e).

    ! si su a e : SubProfile(si,su,a,e) <- ServiceInvokedByUser(su) &
            ServiceInvokedBy(si,su) &
            ServiceStoredPolicyAttr(si,a,e).

    ! si su a e : SubProfile(si,su,a,e) <- ~StorageTrustServInstance(si) &
            ServiceInvokedByUser(su) &
            ServiceInvokedBy(si,su) &
            ServiceStoredRevealAttr(si,a,e).

    ! si su a e : SubProfile(si,su,a,e) <- ~StorageTrustServInstance(si) &
            ServiceInvokedByUser(su) &
            ServiceInvokedBy(si,su) &
            ServiceStoredForwardAttr(si,su,a,e).
  }

  //  The identifiers to which a sub-profile can be linked.
  {
    ! si su id : SubProfileId(si,su,id) <- ? a e :
            SubProfile(si,su,a,e) &
            ~(? a_x : SubProfileIdentifierSetAttr(id,a_x) &
            ~(? e_x : SubProfile(si,su,a_x,e_x))).
  }

  // User profiles
  // (1) : a user profile associated with a user invoked service is the union
  //       of all sub-profiles that are associated with the same user invoked
  //       service and that can be linked to a common identifier
  // (2) : a user profile can be merged with another user profile in case
  //       both user profiles can be linked to a common identifier
  // (3) : a user profile can be merged with a user profile of a collaborating
  //       organization that can be linked to a common identifiers
  {
    ! o id a e su : UserProfile(o,id,su,a,e) <- ? si s[ServiceType] :
            ServiceInvokedByUser(su) &
            SubProfile(si,su,a,e) &
            SubProfileId(si,su,id) &
            Service(s,o) = si.

    ! o id a e su : UserProfile(o,id,su,a,e) <- ? id_x :
            UserProfile(o,id_x,su,a,e) &
            ~(? a_x : SubProfileIdentifierSetAttr(id,a_x) &
            ~(? e_x : UserProfile(o,id_x,su,a_x,e_x))).

    ! o id a e su : UserProfile(o,id,su,a,e) <- ? o_src :
            UserProfile(o_src,id,su,a,e) &
            ShareDataTo(o_src,o).
  }
}
```

Listing C.3: Listing of the input model part of the IDP realization representing the web shop case as presented in Sectioin 4.1

```
structure struct_input_model : voc_privacy {

  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // System model
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  AttrSrc = {
    User;
    GroceryStore;
    LoyaltyProgProvider;
    LoyaltyAppProvider;
    Advertiser;
    MobilePlatform;
    MobileOperator;

    Checkout;
    RequestLoyaltyInfo; ShowLoyaltyInfo;
    CollectLoyaltyPoints;
    PrintTicket;
    CollectDataFromLPP;

    LoyaltyCard;
    ProfLoyaltyProgramProvider;
  }

  Stakeholder = {
    User;
    GroceryStore;
    LoyaltyProgProvider;
    LoyaltyAppProvider;
    Advertiser;
    MobilePlatform;
    MobileOperator;
  }

  User = {
    User;
  }

  Organization = {
    GroceryStore;
    LoyaltyProgProvider;
    LoyaltyAppProvider;
    Advertiser;
    MobilePlatform;
    MobileOperator;
  }

  Attribute = {
    // From user
    Name; Address; EMail; PhoneNumber;
    PersonalProp;

    // From loyalty program provider
    QRCode; PointsToEarn; PointsEarned; PointBalance;

    // From the grocery store
    Bill; Products;

    // Session specific
    SessGroceryStore;
  }

  Property = {
  }

  // Services & Service Policies
  ServiceType = {
    CheckoutType;
    RequestLoyaltyInfoType; ShowLoyaltyInfoType;
    CollectLoyaltyPointsType;
    PrintTicketType;
    CollectDataType;
  }

  ServiceIdentifier = {
    Checkout;
    RequestLoyaltyInfo; ShowLoyaltyInfo;
    CollectLoyaltyPoints;
    PrintTicket;
    CollectDataFromLPP;
  }

  SessionMember = {
    NymSessGroceryStore,Checkout;
    NymSessGroceryStore,ShowLoyaltyInfo;
    NymSessGroceryStore,PrintTicket;
  }

  Service = {
    Checkout,CheckoutType,GroceryStore;
    RequestLoyaltyInfo,RequestLoyaltyInfoType,LoyaltyProgProvider;
    ShowLoyaltyInfo,ShowLoyaltyInfoType,GroceryStore;
    CollectLoyaltyPoints,CollectLoyaltyPointsType,LoyaltyProgProvider;
    PrintTicket,PrintTicketType,GroceryStore;
    CollectDataFromLPP,CollectDataType,Advertiser;
  }

  AccessPolicy = {
    CheckoutAP; RequestLoyaltyInfoAP;
    ShowLoyaltyInfoAP; CollectLoyaltyPointsAP; PrintTicketAP;
    NoneAP;
  }

  StoragePolicy = {
    CollectLoyaltyPointsSP;
    NoneSP;
  }

  DistributionPolicy = {
    CheckoutDP; RequestLoyaltyInfoDP;
    ShowLoyaltyInfoDP; CollectLoyaltyPointsDP;
    NoneDP;
  }

  OutputPolicy = {
    ShowLoyaltyInfoOP; PrintTicketOP;
    NoneOP;
  }

  OwnAuthToken = {
    CheckoutAP,LoyaltyCard;
  }

  RevealAttr = {
    CheckoutAP,LoyaltyCard,QRCode,LoyaltyProgProvider;
    CheckoutAP,User,Products,GroceryStore;

    RequestLoyaltyInfoAP,Checkout,QRCode,LoyaltyProgProvider;
    RequestLoyaltyInfoAP,Checkout,Bill,GroceryStore;

    ShowLoyaltyInfoAP,RequestLoyaltyInfo,PointBalance,LoyaltyProgProvider;
    ShowLoyaltyInfoAP,RequestLoyaltyInfo,PointsToEarn,LoyaltyProgProvider;

    CollectLoyaltyPointsAP,ShowLoyaltyInfo,QRCode,LoyaltyProgProvider;
    CollectLoyaltyPointsAP,ShowLoyaltyInfo,Bill,GroceryStore;

    PrintTicketAP,CollectLoyaltyPoints,PointBalance,LoyaltyProgProvider;
    PrintTicketAP,CollectLoyaltyPoints,PointsEarned,LoyaltyProgProvider;
  }

  GenerateAttr = {
    CheckoutAP,Bill;
    RequestLoyaltyInfoAP,PointsToEarn;
    CollectLoyaltyPointsAP,PointsEarned;
  }

  StoreAttr = {
    CollectLoyaltyPointsSP,Bill,GroceryStore;
    CollectLoyaltyPointsSP,QRCode,LoyaltyProgProvider;
    CollectLoyaltyPointsSP,PointsEarned,LoyaltyProgProvider;
    CollectLoyaltyPointsSP,PointBalance,LoyaltyProgProvider;
  }

  DistrAttrTo = {
    CheckoutDP,RequestLoyaltyInfo,QRCode,LoyaltyProgProvider;
    CheckoutDP,RequestLoyaltyInfo,Bill,GroceryStore;

    RequestLoyaltyInfoDP,ShowLoyaltyInfo,PointsToEarn,LoyaltyProgProvider;
    RequestLoyaltyInfoDP,ShowLoyaltyInfo,PointBalance,LoyaltyProgProvider;

    ShowLoyaltyInfoDP,CollectLoyaltyPoints,QRCode,LoyaltyProgProvider;
    ShowLoyaltyInfoDP,CollectLoyaltyPoints,Bill,GroceryStore;

    CollectLoyaltyPointsDP,PrintTicket,PointBalance,LoyaltyProgProvider;
    CollectLoyaltyPointsDP,PrintTicket,PointsEarned,LoyaltyProgProvider;

    CollectLoyaltyPointsDP,CollectDataFromLPP,Bill,GroceryStore;
  }

  OutputAttr = {
    ShowLoyaltyInfoOP,PointsToEarn; ShowLoyaltyInfoOP,PointBalance;
    PrintTicketOP,PointsEarned; PrintTicketOP,PointBalance; PrintTicketOP,Bill;
  }

  ServicePolicies = {
    Checkout,CheckoutAP,NoneSP,CheckoutDP,NoneOP;
    RequestLoyaltyInfo,RequestLoyaltyInfoAP,NoneSP,RequestLoyaltyInfoDP,NoneOP;
    ShowLoyaltyInfo,ShowLoyaltyInfoAP,NoneSP,ShowLoyaltyInfoDP,
    ShowLoyaltyInfoOP;
    CollectLoyaltyPoints,CollectLoyaltyPointsAP,CollectLoyaltyPointsSP,
    CollectLoyaltyPointsDP,NoneOP;
    PrintTicket,PrintTicketAP,NoneSP,NoneDP,PrintTicketOP;
    CollectDataFromLPP,NoneAP,NoneSP,NoneDP,NoneOP;
  }


  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // User model
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

  // Trust perception by the user
  StorageTrust = {
    GroceryStore;
    MobilePlatform;
    MobileOperator;
  }

  DistributionTrust = {
    GroceryStore;
    MobilePlatform;
    MobileOperator;
  }

  // Initial state of the user
  ClaimBasedTech = { PlasticCard; X509; Idemix; }
  X509Tech = X509
  IdemixTech = Idemix
  Credential = { LoyaltyCard; }

  CredAttr = {
    LoyaltyCard,QRCode;
  }

  CredTech = {
    LoyaltyCard,PlasticCard;
  }

  CredIssuer = {
    LoyaltyCard,LoyaltyProgProvider;
  }

  ProfileAccessType = { Public; Restricted;}
  PublicAccess = Public
  RestrictedAccess = Restricted

  Profile = {
    ProfLoyaltyProgramProvider;
  }

  ProfAttr = {
    ProfLoyaltyProgramProvider,Address,User;
    ProfLoyaltyProgramProvider,EMail,User;
    ProfLoyaltyProgramProvider,PhoneNumber,User;
    ProfLoyaltyProgramProvider,PersonalProp,User;
    ProfLoyaltyProgramProvider,QRCode,LoyaltyProgProvider;
    ProfLoyaltyProgramProvider,PointBalance,LoyaltyProgProvider;
  }

  ProfOwner = {
    ProfLoyaltyProgramProvider,LoyaltyProgProvider;
  }

  ProfAccessType = {
    ProfLoyaltyProgramProvider,Restricted;
  }

  ProfAccess = {
    ProfLoyaltyProgramProvider,LoyaltyProgProvider;
  }

  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  // Identifier model
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  Identifier = {
    Identity1; Identity2;
    Nym1; Nym2; Nym3;
    NymSessGroceryStore;
  }

  Identity = { Identity1; Identity2; }

  Pseudonym = { Nym1; Nym2; Nym3;
    NymSessGroceryStore;
  }

  IdentifierSetAttr = {
    Identity1,Name; Identity2,Address;
    Nym1,PhoneNumber; Nym2,EMail; Nym3,QRCode;
    NymSessGroceryStore,SessGroceryStore;
  }
}


include <mx>
procedure main() {
  stdoptions.nbmodels = 1
  stdoptions.xsb = true

  theory = merge(theo_general,theo_behavior)
  theory = merge(theory,theo_build_profiles)

  models = modelexpand(theory,struct_input_model)

  printmodels(models)
}
```

Appendix D

IDP Output Model

This appendix presents the resulting IDP output model[1] associated with the web shop (see Section 4.1) input model as presented in Appendix C. It is worth mentioning that the listing shows the partial output model only containing the user profiles of the organizations in the web shop scenario. The user profiles are represented by relations *UserProfile*(*Organization*, *Identifier*, *ServiceIdentifier*, *Attribute*, *Stakeholder*).

Listing D.1: IDP output representing the resulting user profiles of the organizations in the web shop case of Section 4.1

```
UserProfile[voc_privacy::Organization,voc_privacy::Identifier,
voc_privacy::ServiceIdentifier,
voc_privacy::Attribute,voc_privacy::Stakeholder] =
{
  PS,Identity1,BasicPurchaseServ,Address,User
  PS,Identity1,BasicPurchaseServ,EMail,User
  PS,Identity1,BasicPurchaseServ,Name,User
  PS,Identity1,BasicPurchaseServ,OrderNr,WS
  PS,Identity1,BasicPurchaseServ,URL,PS

  PS,Identity1,BasicReductionPurchaseServ,Address,User
  PS,Identity1,BasicReductionPurchaseServ,EMail,User
  PS,Identity1,BasicReductionPurchaseServ,Name,User
  PS,Identity1,BasicReductionPurchaseServ,OrderNr,WS
  PS,Identity1,BasicReductionPurchaseServ,URL,PS

  PS,Identity1,PrivPurchaseServ,Address,User
  PS,Identity1,PrivPurchaseServ,EMail,User
  PS,Identity1,PrivPurchaseServ,Name,User
  PS,Identity1,PrivPurchaseServ,OrderNr,WS
  PS,Identity1,PrivPurchaseServ,URL,PS

  PS,Identity1,PrivReductionPurchaseServ,Address,User
  PS,Identity1,PrivReductionPurchaseServ,EMail,User
  PS,Identity1,PrivReductionPurchaseServ,Name,User
  PS,Identity1,PrivReductionPurchaseServ,OrderNr,WS
  PS,Identity1,PrivReductionPurchaseServ,URL,PS

  PS,Nym2,BasicPurchaseServ,Address,User
  PS,Nym2,BasicPurchaseServ,EMail,User
  PS,Nym2,BasicPurchaseServ,Name,User
  PS,Nym2,BasicPurchaseServ,OrderNr,WS
```

[1]The IDP realization of the framework is publicly available at `https://github.com/decroik/inspect-privacy-and-trust/`.

```
PS,Nym2,BasicPurchaseServ,URL,PS

PS,Nym2,BasicReductionPurchaseServ,Address,User
PS,Nym2,BasicReductionPurchaseServ,EMail,User
PS,Nym2,BasicReductionPurchaseServ,Name,User
PS,Nym2,BasicReductionPurchaseServ,OrderNr,WS
PS,Nym2,BasicReductionPurchaseServ,URL,PS

PS,Nym2,PrivPurchaseServ,Address,User
PS,Nym2,PrivPurchaseServ,EMail,User
PS,Nym2,PrivPurchaseServ,Name,User
PS,Nym2,PrivPurchaseServ,OrderNr,WS
PS,Nym2,PrivPurchaseServ,URL,PS

PS,Nym2,PrivReductionPurchaseServ,Address,User
PS,Nym2,PrivReductionPurchaseServ,EMail,User
PS,Nym2,PrivReductionPurchaseServ,Name,User
PS,Nym2,PrivReductionPurchaseServ,OrderNr,WS
PS,Nym2,PrivReductionPurchaseServ,URL,PS

PS,Nym3,BasicPurchaseServ,Address,User
PS,Nym3,BasicPurchaseServ,EMail,User
PS,Nym3,BasicPurchaseServ,Name,User
PS,Nym3,BasicPurchaseServ,OrderNr,WS
PS,Nym3,BasicPurchaseServ,URL,PS

PS,Nym3,BasicReductionPurchaseServ,Address,User
PS,Nym3,BasicReductionPurchaseServ,EMail,User
PS,Nym3,BasicReductionPurchaseServ,Name,User
PS,Nym3,BasicReductionPurchaseServ,OrderNr,WS
PS,Nym3,BasicReductionPurchaseServ,URL,PS

PS,Nym3,PrivPurchaseServ,Address,User
PS,Nym3,PrivPurchaseServ,EMail,User
PS,Nym3,PrivPurchaseServ,Name,User
PS,Nym3,PrivPurchaseServ,OrderNr,WS
PS,Nym3,PrivPurchaseServ,URL,PS

PS,Nym3,PrivReductionPurchaseServ,Address,User
PS,Nym3,PrivReductionPurchaseServ,EMail,User
PS,Nym3,PrivReductionPurchaseServ,Name,User
PS,Nym3,PrivReductionPurchaseServ,OrderNr,WS
PS,Nym3,PrivReductionPurchaseServ,URL,PS

PS,Nym4,BasicPurchaseServ,Address,User
PS,Nym4,BasicPurchaseServ,EMail,User
PS,Nym4,BasicPurchaseServ,Name,User
PS,Nym4,BasicPurchaseServ,OrderNr,WS
PS,Nym4,BasicPurchaseServ,URL,PS

PS,Nym4,BasicReductionPurchaseServ,Address,User
PS,Nym4,BasicReductionPurchaseServ,EMail,User
PS,Nym4,BasicReductionPurchaseServ,Name,User
PS,Nym4,BasicReductionPurchaseServ,OrderNr,WS
PS,Nym4,BasicReductionPurchaseServ,URL,PS

PS,Nym4,PrivPurchaseServ,Address,User
PS,Nym4,PrivPurchaseServ,EMail,User
PS,Nym4,PrivPurchaseServ,Name,User
PS,Nym4,PrivPurchaseServ,OrderNr,WS
PS,Nym4,PrivPurchaseServ,URL,PS

PS,Nym4,PrivReductionPurchaseServ,Address,User
PS,Nym4,PrivReductionPurchaseServ,EMail,User
PS,Nym4,PrivReductionPurchaseServ,Name,User
PS,Nym4,PrivReductionPurchaseServ,OrderNr,WS
PS,Nym4,PrivReductionPurchaseServ,URL,PS

WS,Identity1,BasicPurchaseServ,Address,Government
WS,Identity1,BasicPurchaseServ,Address,User
WS,Identity1,BasicPurchaseServ,AgeLimit,Government
WS,Identity1,BasicPurchaseServ,DoB,Government
WS,Identity1,BasicPurchaseServ,EMail,User
WS,Identity1,BasicPurchaseServ,Name,Government
WS,Identity1,BasicPurchaseServ,Name,User
WS,Identity1,BasicPurchaseServ,OrderNr,WS
WS,Identity1,BasicPurchaseServ,SSN,Government
WS,Identity1,BasicPurchaseServ,URL,PS

WS,Identity1,BasicReductionPurchaseServ,Address,Government
WS,Identity1,BasicReductionPurchaseServ,Address,University
WS,Identity1,BasicReductionPurchaseServ,Address,User
WS,Identity1,BasicReductionPurchaseServ,AgeLimit,Government
WS,Identity1,BasicReductionPurchaseServ,DoB,Government
```

```
WS,Identity1,BasicReductionPurchaseServ,EMail,User
WS,Identity1,BasicReductionPurchaseServ,Institute,University
WS,Identity1,BasicReductionPurchaseServ,Name,Government
WS,Identity1,BasicReductionPurchaseServ,Name,University
WS,Identity1,BasicReductionPurchaseServ,Name,User
WS,Identity1,BasicReductionPurchaseServ,OrderNr,WS
WS,Identity1,BasicReductionPurchaseServ,SSN,Government
WS,Identity1,BasicReductionPurchaseServ,URL,PS

WS,Identity1,PrivPurchaseServ,Address,User
WS,Identity1,PrivPurchaseServ,AgeLimit,Government
WS,Identity1,PrivPurchaseServ,EMail,User
WS,Identity1,PrivPurchaseServ,Name,User
WS,Identity1,PrivPurchaseServ,OrderNr,WS
WS,Identity1,PrivPurchaseServ,URL,PS

WS,Identity1,PrivReductionPurchaseServ,Address,University
WS,Identity1,PrivReductionPurchaseServ,Address,User
WS,Identity1,PrivReductionPurchaseServ,AgeLimit,Government
WS,Identity1,PrivReductionPurchaseServ,EMail,User
WS,Identity1,PrivReductionPurchaseServ,Institute,University
WS,Identity1,PrivReductionPurchaseServ,Name,University
WS,Identity1,PrivReductionPurchaseServ,Name,User
WS,Identity1,PrivReductionPurchaseServ,OrderNr,WS
WS,Identity1,PrivReductionPurchaseServ,URL,PS

WS,Nym1,BasicPurchaseServ,Address,Government
WS,Nym1,BasicPurchaseServ,Address,User
WS,Nym1,BasicPurchaseServ,AgeLimit,Government
WS,Nym1,BasicPurchaseServ,DoB,Government
WS,Nym1,BasicPurchaseServ,EMail,User
WS,Nym1,BasicPurchaseServ,Name,Government
WS,Nym1,BasicPurchaseServ,Name,User
WS,Nym1,BasicPurchaseServ,OrderNr,WS
WS,Nym1,BasicPurchaseServ,SSN,Government
WS,Nym1,BasicPurchaseServ,URL,PS

WS,Nym1,BasicReductionPurchaseServ,Address,Government
WS,Nym1,BasicReductionPurchaseServ,Address,University
WS,Nym1,BasicReductionPurchaseServ,Address,User
WS,Nym1,BasicReductionPurchaseServ,AgeLimit,Government
WS,Nym1,BasicReductionPurchaseServ,DoB,Government
WS,Nym1,BasicReductionPurchaseServ,EMail,User
WS,Nym1,BasicReductionPurchaseServ,Institute,University
WS,Nym1,BasicReductionPurchaseServ,Name,Government
WS,Nym1,BasicReductionPurchaseServ,Name,University
WS,Nym1,BasicReductionPurchaseServ,Name,User
WS,Nym1,BasicReductionPurchaseServ,OrderNr,WS
WS,Nym1,BasicReductionPurchaseServ,SSN,Government
WS,Nym1,BasicReductionPurchaseServ,URL,PS

WS,Nym2,BasicPurchaseServ,Address,Government
WS,Nym2,BasicPurchaseServ,Address,User
WS,Nym2,BasicPurchaseServ,AgeLimit,Government
WS,Nym2,BasicPurchaseServ,DoB,Government
WS,Nym2,BasicPurchaseServ,EMail,User
WS,Nym2,BasicPurchaseServ,Name,Government
WS,Nym2,BasicPurchaseServ,Name,User
WS,Nym2,BasicPurchaseServ,OrderNr,WS
WS,Nym2,BasicPurchaseServ,SSN,Government
WS,Nym2,BasicPurchaseServ,URL,PS

WS,Nym2,BasicReductionPurchaseServ,Address,Government
WS,Nym2,BasicReductionPurchaseServ,Address,University
WS,Nym2,BasicReductionPurchaseServ,Address,User
WS,Nym2,BasicReductionPurchaseServ,AgeLimit,Government
WS,Nym2,BasicReductionPurchaseServ,DoB,Government
WS,Nym2,BasicReductionPurchaseServ,EMail,User
WS,Nym2,BasicReductionPurchaseServ,Institute,University
WS,Nym2,BasicReductionPurchaseServ,Name,Government
WS,Nym2,BasicReductionPurchaseServ,Name,University
WS,Nym2,BasicReductionPurchaseServ,Name,User
WS,Nym2,BasicReductionPurchaseServ,OrderNr,WS
WS,Nym2,BasicReductionPurchaseServ,SSN,Government
WS,Nym2,BasicReductionPurchaseServ,URL,PS

WS,Nym2,PrivPurchaseServ,Address,User
WS,Nym2,PrivPurchaseServ,AgeLimit,Government
WS,Nym2,PrivPurchaseServ,EMail,User
WS,Nym2,PrivPurchaseServ,Name,User
WS,Nym2,PrivPurchaseServ,OrderNr,WS
WS,Nym2,PrivPurchaseServ,URL,PS

WS,Nym2,PrivReductionPurchaseServ,Address,University
```

```
WS,Nym2,PrivReductionPurchaseServ,Address,User
WS,Nym2,PrivReductionPurchaseServ,AgeLimit,Government
WS,Nym2,PrivReductionPurchaseServ,EMail,User
WS,Nym2,PrivReductionPurchaseServ,Institute,University
WS,Nym2,PrivReductionPurchaseServ,Name,University
WS,Nym2,PrivReductionPurchaseServ,Name,User
WS,Nym2,PrivReductionPurchaseServ,OrderNr,WS
WS,Nym2,PrivReductionPurchaseServ,URL,PS

WS,Nym3,BasicPurchaseServ,Address,Government
WS,Nym3,BasicPurchaseServ,Address,User
WS,Nym3,BasicPurchaseServ,AgeLimit,Government
WS,Nym3,BasicPurchaseServ,DoB,Government
WS,Nym3,BasicPurchaseServ,EMail,User
WS,Nym3,BasicPurchaseServ,Name,Government
WS,Nym3,BasicPurchaseServ,Name,User
WS,Nym3,BasicPurchaseServ,OrderNr,WS
WS,Nym3,BasicPurchaseServ,SSN,Government
WS,Nym3,BasicPurchaseServ,URL,PS

WS,Nym3,BasicReductionPurchaseServ,Address,Government
WS,Nym3,BasicReductionPurchaseServ,Address,University
WS,Nym3,BasicReductionPurchaseServ,Address,User
WS,Nym3,BasicReductionPurchaseServ,AgeLimit,Government
WS,Nym3,BasicReductionPurchaseServ,DoB,Government
WS,Nym3,BasicReductionPurchaseServ,EMail,User
WS,Nym3,BasicReductionPurchaseServ,Institute,University
WS,Nym3,BasicReductionPurchaseServ,Name,Government
WS,Nym3,BasicReductionPurchaseServ,Name,University
WS,Nym3,BasicReductionPurchaseServ,Name,User
WS,Nym3,BasicReductionPurchaseServ,OrderNr,WS
WS,Nym3,BasicReductionPurchaseServ,SSN,Government
WS,Nym3,BasicReductionPurchaseServ,URL,PS

WS,Nym3,PrivPurchaseServ,Address,User
WS,Nym3,PrivPurchaseServ,AgeLimit,Government
WS,Nym3,PrivPurchaseServ,EMail,User
WS,Nym3,PrivPurchaseServ,Name,User
WS,Nym3,PrivPurchaseServ,OrderNr,WS
WS,Nym3,PrivPurchaseServ,URL,PS

WS,Nym3,PrivReductionPurchaseServ,Address,University
WS,Nym3,PrivReductionPurchaseServ,Address,User
WS,Nym3,PrivReductionPurchaseServ,AgeLimit,Government
WS,Nym3,PrivReductionPurchaseServ,EMail,User
WS,Nym3,PrivReductionPurchaseServ,Institute,University
WS,Nym3,PrivReductionPurchaseServ,Name,University
WS,Nym3,PrivReductionPurchaseServ,Name,User
WS,Nym3,PrivReductionPurchaseServ,OrderNr,WS
WS,Nym3,PrivReductionPurchaseServ,URL,PS

WS,Nym4,BasicPurchaseServ,Address,Government
WS,Nym4,BasicPurchaseServ,Address,User
WS,Nym4,BasicPurchaseServ,AgeLimit,Government
WS,Nym4,BasicPurchaseServ,DoB,Government
WS,Nym4,BasicPurchaseServ,EMail,User
WS,Nym4,BasicPurchaseServ,Name,Government
WS,Nym4,BasicPurchaseServ,Name,User
WS,Nym4,BasicPurchaseServ,OrderNr,WS
WS,Nym4,BasicPurchaseServ,SSN,Government
WS,Nym4,BasicPurchaseServ,URL,PS

WS,Nym4,BasicReductionPurchaseServ,Address,Government
WS,Nym4,BasicReductionPurchaseServ,Address,University
WS,Nym4,BasicReductionPurchaseServ,Address,User
WS,Nym4,BasicReductionPurchaseServ,AgeLimit,Government
WS,Nym4,BasicReductionPurchaseServ,DoB,Government
WS,Nym4,BasicReductionPurchaseServ,EMail,User
WS,Nym4,BasicReductionPurchaseServ,Institute,University
WS,Nym4,BasicReductionPurchaseServ,Name,Government
WS,Nym4,BasicReductionPurchaseServ,Name,University
WS,Nym4,BasicReductionPurchaseServ,Name,User
WS,Nym4,BasicReductionPurchaseServ,OrderNr,WS
WS,Nym4,BasicReductionPurchaseServ,SSN,Government
WS,Nym4,BasicReductionPurchaseServ,URL,PS

WS,Nym4,PrivPurchaseServ,Address,User
WS,Nym4,PrivPurchaseServ,AgeLimit,Government
WS,Nym4,PrivPurchaseServ,EMail,User
WS,Nym4,PrivPurchaseServ,Name,User
WS,Nym4,PrivPurchaseServ,OrderNr,WS
WS,Nym4,PrivPurchaseServ,URL,PS

WS,Nym4,PrivReductionPurchaseServ,Address,University
```

```
  WS,Nym4,PrivReductionPurchaseServ,Address,User
  WS,Nym4,PrivReductionPurchaseServ,AgeLimit,Government
  WS,Nym4,PrivReductionPurchaseServ,EMail,User
  WS,Nym4,PrivReductionPurchaseServ,Institute,University
  WS,Nym4,PrivReductionPurchaseServ,Name,University
  WS,Nym4,PrivReductionPurchaseServ,Name,User
  WS,Nym4,PrivReductionPurchaseServ,OrderNr,WS
  WS,Nym4,PrivReductionPurchaseServ,URL,PS
}
```

Bibliography

[1] Comodo Secure. `https://www.comodo.com`. [Online; accessed 20-04-2015]. pages 3, 20

[2] PrivAcy pReserving Infrastructure for Surveillance (PARIS). `http://www.paris-project.org`. [Online; accessed 24-09-2015]. pages 124

[3] TRUSTe. `https://www.truste.com`. [Online; accessed 20-04-2015]. pages 3, 20

[4] Summary of the HIPAA privacy rule. `http://www.hhs.gov/ocr/privacy/hipaa/understanding/summary/`, 2003. [Online; accessed 13-04-2015]. pages 27

[5] Attribute-based Credential for Trust (ABC4Trust). `https://abc4trust.eu/`, 2010. [Online; accessed 23-03-2015]. pages 14

[6] ENDORSE. `http://ict-endorse.eu/`, 2010. [Online; accessed 23-03-2015]. pages 19

[7] ISO/IEC 29100:2011 – Information technology – Security techniques – Privacy framework. [Online; accessed 24-09-2015]. pages 19

[8] Proposal for a Regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation), COM/2012/011, 2012. pages 2

[9] CPLEX CP Optimizer. `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/`, 2015. [Online; accessed 03-04-2015]. pages 32

[10] Google Dashboard. `https://www.google.com/settings/dashboard`, 2015. [Online; accessed 24-06-2015]. pages 3

[11] IDP. `https://dtai.cs.kuleuven.be/software/idp`, 2015. [Online; accessed 05-04-2015]. pages 30

[12] Abe, M., and Fujisaki, E. How to Date Blind Signatures. In *Advances in Cryptology - ASIACRYPT '96*, K. Kim and T. Matsumoto, Eds., vol. 1163 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1996, pp. 244–251. pages 14

[13] Abrial, JR. *Modeling in Event-B: System and Software Engineering.* Cambridge University Press, 2010. pages 33

[14] Agre, P. E. THE ARCHITECTURE OF IDENTITY: Embedding Privacy in Market Institutions. *Information, Communication & Society 2*, 1 (1999), 1–25. pages 10

[15] Almasizadeh, J., and Azgomi, M. A. Mean Privacy: A Metric for Security of Computer Systems. *Computer Communications 52*, 0 (2014), 47 – 59. pages 25

[16] Alpár, G. *Attribute-Based Identity Management.* PhD thesis, Radboud University, 2015. pages 14

[17] Alviano, M., Calimeri, F., Charwat, G., Dao-Tran, M., Dodaro, C., Ianni, G., Krennwallner, T., Kronegger, M., Oetsch, J., Pfandler, A., Pührer, J., Redl, C., Ricca, F., Schneider, P., Schwengerer, M., Spendier, L., Wallner, J., and Xiao, G. The Fourth Answer Set Programming Competition: Preliminary Report. In *Logic Programming and Nonmonotonic Reasoning*, P. Cabalar and T. Son, Eds., vol. 8148 of *Lecture Notes in Computer Science.* Springer Berlin Heidelberg, 2013, pp. 42–53. pages 32

[18] Amadini, R., Gabbrielli, M., and Mauro, J. An Empirical Evaluation of Portfolios Approaches for Solving CSPs. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, C. Gomes and M. Sellmann, Eds., vol. 7874 of *Lecture Notes in Computer Science.* Springer Berlin Heidelberg, 2013, pp. 316–324. pages 33

[19] Anderson, A. A Comparison of Two Privacy Policy Languages: EPAL and XACML. Tech. rep., Mountain View, CA, USA, 2005. pages 16

[20] Antignac, T., and Le Métayer, D. Privacy Architectures: Reasoning about Data Minimisation and Integrity. In *Security and Trust Management*, S. Mauw and C. Jensen, Eds., vol. 8743 of *Lecture Notes in Computer Science.* Springer International Publishing, 2014, pp. 17–32. pages 28

[21] Antignac, T., and Le Métayer, D. Privacy by Design: From Technologies to Architectures. In *Privacy Technologies and Policy*, B. Preneel and D. Ikonomou, Eds., vol. 8450 of *Lecture Notes in Computer Science.* Springer International Publishing, 2014, pp. 1–17. pages 28

[22] Ardagna, C. A., De Capitani di Vimercati, S., Neven, G., Paraboschi, S., Preiss, F.-S., Samarati, P., and Verdicchio, M. Enabling Privacy-preserving Credential-based Access Control with XACML and SAML. In *Proceedings of the 2010 10th IEEE International Conference on Computer and Information Technology* (Washington, DC, USA, 2010), CIT '10, IEEE Computer Society, pp. 1090–1095. pages 16, 53, 69

[23] Ashley, P., Hada, S., Karjoth, G., Powers, C., and Schunter, M. Enterprise Privacy Authorization Language (EPAL 1.2). *Submission to W3C* (2003). pages 15

[24] Austin, T. H., Yang, J., Flanagan, C., and Solar-Lezama, A. Faceted Execution of Policy-agnostic Programs. In *Proceedings of the Eighth ACM SIGPLAN Workshop on Programming Languages and Analysis for Security* (New York, NY, USA, 2013), PLAS '13, ACM, pp. 15–26. pages 27

[25] Barth, A., Datta, A., Mitchell, J. C., and Nissenbaum, H. Privacy and Contextual Integrity: Framework and Applications. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy* (Washington, DC, 2006), SP '06, IEEE Computer Society, pp. 184–198. pages 27, 149

[26] Basin, D., Klaedtke, F., Marinovic, S., and Zalinescu, E. Monitoring Compliance Policies over Incomplete and Disagreeing Logs. In *Runtime Verification*, S. Qadeer and S. Tasiran, Eds., vol. 7687 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 151–167. pages 123

[27] Beckers, K., Fassbender, S., Heisel, M., and Meis, R. A Problem-based Approach for Computer-aided Privacy Threat Identification. In *Privacy Technologies and Policy*, B. Preneel and D. Ikonomou, Eds., vol. 8319 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2014, pp. 1–16. pages 23

[28] Beckers, K., Fassbender, S., and Schmidt, H. An Integrated Method for Pattern-based Elicitation of Legal Requirements Applied to a Cloud Computing Example. In *Availability, Reliability and Security (ARES), 2012 Seventh International Conference on* (Aug 2012), pp. 463–472. pages 24

[29] Berthold, O., Federrath, H., and Köhntopp, M. Project "Anonymity and Unobservability in the Internet". In *Proceedings of the Tenth Conference on Computers, Freedom and Privacy: Challenging the Assumptions* (New York, NY, USA, 2000), CFP '00, ACM, pp. 57–65. pages 14

[30] Berthold, O., Pfitzmann, A., and Standtke, R. The Disadvantages of Free MIX Routes and How to Overcome Them. In *Designing Privacy Enhancing Technologies*, H. Federrath, Ed., vol. 2009 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2001, pp. 30–45. pages 24

[31] Blanchet, B. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. *Computer Security Foundations Workshop, IEEE 0* (2001), 0082. pages 27

[32] Blocki, J., Christin, N., Datta, A., Procaccia, A. D., and Sinha, A. Audit Games. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence* (2013), F. Rossi, Ed., IJCAI '13, AAAI Press, pp. 41–47. pages 123

[33] Blocki, J., Christin, N., Datta, A., and Sinha, A. Adaptive Regret Minimization in Bounded-memory Games. In *Decision and Game Theory for Security*, S. Das, C. Nita-Rotaru, and M. Kantarcioglu, Eds., vol. 8252 of *Lecture Notes in Computer Science*. Springer International Publishing, 2013, pp. 65–84. pages 123

[34] Bogaerts, B., Jansen, J., Bruynooghe, M., de Cat, B., Vennekens, J., and Denecker, M. Simulating Dynamic Systems using Linear Time Calculus Theories. *CoRR abs/1405.1523* (2014). pages 98

[35] Böhm, A., Murtz, B., Sommer, G., and Wermuth, M. Location-based Ticketing in Public Transport. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE* (2005), IEEE, pp. 194–197. pages 106

[36] Brassard, G., Chaum, D., and Crépeau, C. Minimum Disclosure Proofs of Knowledge. *Journal of Computer and System Sciences 37*, 2 (1988), 156 – 189. pages 14

[37] Breaux, T., Hibshi, H., and Rao, A. Eddy, a Formal Language for Specifying and Analyzing Data Flow Specifications for Conflicting Privacy Requirements. *Requirements Engineering 19*, 3 (2014), 281–307. pages 27

[38] Breaux, T. D., and Rao, A. Formal Analysis of Privacy Requirements Specifications for Multi-tier Applications. In *RE'13: Proceedings of the 21st IEEE International Requirements Engineering Conference (RE'13)* (Washington, DC, July 2013), IEEE Society Press, pp. 14–23. pages 27

[39] Butin, D., Chicote, M., and Le Métayer, D. Log Design for Accountability. In *2013 IEEE Security & Privacy Workshop on Data Usage Management* (2013), IEEE Computer Society, pp. 1–7. pages 122

[40] Butin, D., and Le Métayer, D. Log Analysis for Data Protection Accountability. In *FM 2014: Formal Methods*, C. Jones, P. Pihlajasaari, and J. Sun, Eds., vol. 8442 of *Lecture Notes in Computer Science*. Springer, 2014, pp. 163–178. pages 122

[41] Calimeri, F., Faber, W., Gebser, M., Ianni, G., Kaminski, R., Krennwallner, T., Leone, N., Ricca, F., and Schaub, T. ASP-Core-2 Input Language Format. `https://www.mat.unical.it/aspcomp2013/ASPStandardization`, 2013. pages 32

[42] Calypso Networks Association. Calypso Functional Specification, Card Application. pages 93, 106

[43] Camenisch, J., Mödersheim, S., Neven, G., Preiss, F.-S., and Sommer, D. A Card Requirements Language Enabling Privacy-preserving Access Control. In *Proceedings of the 15th ACM Symposium on Access Control Models and Technologies* (New York, NY, USA, 2010), SACMAT '10, ACM, pp. 119–128. pages 16, 53, 69

[44] Camenisch, J., and Van Herreweghen, E. Design and Implementation of the Idemix Anonymous Credential System. In *Proceedings of the 9th ACM Conference on Computer and Communications Security* (New York, NY, USA, 2002), V. Atluri, Ed., CCS '02, ACM, pp. 21–30. pages 14, 42

[45] Cantor, S., and Scavo, T. Shibboleth Architecture Technical Overview Working Draft 02. `http://shibboleth.internet2.edu/shibboleth-documents.html`. [Online; accessed 24-09-2015]. pages 11

[46] Cavoukian, A., et al. Privacy by Design: The 7 Foundational Principles. *Information and Privacy Commissioner of Ontario, Canada* (2009). pages 17

[47] Cavoukian, A., Taylor, S., and Abrams, M. Privacy by Design: Essential for Organizational Accountability and Strong Business Practices. *Identity in the Information Society 3*, 2 (2010), 405–413. pages 5, 122

[48] Chaum, D. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Commun. ACM 24*, 2 (Feb. 1981), 84–90. pages 14

[49] Chaum, D. Blind Signatures for Untraceable Payments. In *Advances in Cryptology*, D. Chaum, R. Rivest, and A. Sherman, Eds. Springer US, 1983, pp. 199–203. pages 14

[50] Cheang, B., Li, H., Lim, A., and Rodrigues, B. Nurse Rostering Problems – a Bibliographic Survey. *European Journal of Operational Research 151*, 3 (2003), 447–460. pages 32

[51] Clark, K. Negation as Failure. In *Logic and Data Bases*, H. Gallaire and J. Minker, Eds. Springer US, 1978, pp. 293–322. pages 31

[52] Clarke, I., Sandberg, O., Wiley, B., and Hong, T. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *Designing Privacy Enhancing Technologies*, H. Federrath, Ed., vol. 2009 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2001, pp. 46–66. pages 14

[53] Colin J. Bennett. Implementing Privacy Codes of Practice. *Canadian Standards Association* (1995). pages 123

[54] Cortier, V. Electronic Voting: How Logic Can Help. In *Automated Reasoning*, S. Demri, D. Kapur, and C. Weidenbach, Eds., vol. 8562 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 16–25. pages 27

[55] Cortier, V., and Wiedling, C. A Formal Analysis of the Norwegian E-voting Protocol. In *Principles of Security and Trust*, P. Degano and J. Guttman, Eds., vol. 7215 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 109–128. pages 27

[56] Costanzo, D., and Shao, Z. A Separation Logic for Enforcing Declarative Information Flow Control Policies. In *Principles of Security and Trust*, M. Abadi and S. Kremer, Eds., vol. 8414 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2014, pp. 179–198. pages 27

[57] Danezis, G. Measuring Anonymity: a Few Thoughts and a Differentially Private Bound. In *Proceedings of the DIMACS Workshop on Measuring Anonymity* (2013). pages 26

[58] Danezis, G., Dingledine, R., and Mathewson, N. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Security and Privacy, 2003. Proceedings. 2003 Symposium on* (May 2003), pp. 2–15. pages 14

[59] Danezis, G., Domingo-Ferrer, J., Hansen, M., Hoepman, J., Le Métayer, D., Tirtea, R., and Schiffner, S. Privacy and Data Protection by Design – From Policy to Engineering. *CoRR abs/1501.03726* (2015). pages 19

[60] Dantzig, G. B. *Linear Programming and Extensions*. Princeton University Press, 1998. pages 32

[61] De Cat, B., Bogaerts, B., and Denecker, M. MiniSAT (ID) for Satisfiability Checking and Constraint Solving. *ALP Newsletter* (2014). pages 32, 33

[62] De Hert, P., and Gutwirth, S. Privacy, Data Protection and Law Enforcement. Opacity of the Individual and Transparency of Power. *Privacy and the Criminal Law* (2006), 61–104. pages 11

[63] De Montjoye, Y.-A., Radaelli, L., Singh, V. K., and Pentland, A. Unique in The Shopping Mall: On the Reidentifiability of Credit Card Metadata. *Science 347*, 6221 (2015), 536–539. pages 42

[64] De Pooter, S., Wittocx, J., and Denecker, M. A Prototype of a Knowledge-based Programming Environment. In *Applications of Declarative Programming and Knowledge Management*, H. Tompits, S. Abreu, J. Oetsch, J. Pührer, D. Seipel, M. Umeda, and A. Wolf, Eds., vol. 7773 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 279–286. pages 30

[65] Decroix, K., Butin, D., Jansen, J., and Naessens, V. Inferring Accountability from Trust Perceptions (16% acceptance rate). In *Information Systems Security*, A. Prakash and R. Shyamasundar, Eds., vol. 8880 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 69–88. pages 122

[66] Decroix, K., Lapon, J., Decker, B., and Naessens, V. A Framework for Formal Reasoning about Privacy Properties Based on Trust Relationships in Complex Electronic Services (24% acceptance rate). In *Information Systems Security*, A. Bagchi and I. Ray, Eds., vol. 8303 of *Lecture Notes in Computer Science*. Springer, 2013, pp. 106–120. pages 56

[67] Decroix, K., Lapon, J., Decker, B. D., and Naessens, V. A Formal Approach for Inspecting Privacy and Trust in Advanced Electronic Services (24% acceptance rate). In *Engineering Secure Software and Systems* (2013), J. Jürjens, B. Livshits, and R. Scandariato, Eds., vol. 7781 of *Lecture Notes in Computer Science*, Springer, pp. 155–170. pages 38

[68] Decroix, K., Lapon, J., Lemaire, L., De Decker, B., and Naessens, V. Formal Reasoning about Privacy and Trust in Loyalty Systems. In *BIS 2015 Workshop Post-conference Proceedings (Published in November 2015)*, W. Abramowicz, Ed., vol. 0000 of *Lecture Notes in Business Information Processing*. Springer International Publishing, 2015, pp. 1–12. pages 71

[69] Denecker, M. Extending Classical Logic with Inductive Definitions. In *Computational Logic - CL 2000, First International Conference, London, UK, July 2000, Proceedings* (2000), J. Lloyd, V. Dahl, U. Furbach, M. Kerber, K.-K. Lau, C. Palamidessi, L. M. Pereira, Y. Sagiv, and P. J. Stuckey, Eds., vol. 1861 of *Lecture Notes in Artificial Intelligence*, Springer, pp. 703–717. pages 30

[70] Denecker, M., Vennekens, J., Vlaeminck, H., Wittocx, J., and Bruynooghe, M. Answer Set Programming's Contributions to Classical Logic. In *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning*, M. Balduccini and T. Son, Eds., vol. 6565 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2011, pp. 12–32. pages 32

[71] Deng, M., Wuyts, K., Scandariato, R., Preneel, B., and Joosen, W. A Privacy Threat Analysis Framework: Supporting the Elicitation and Fulfillment

of Privacy Requirements. *Requirements Engineering 16*, 1 (2011), 3–32. pages 23

[72] DeYoung, H., Garg, D., Jia, L., Kaynar, D., and Datta, A. Experiences in the Logical Specification of the HIPAA and GLBA Privacy Laws. In *Proceedings of the 9th Annual ACM Workshop on Privacy in the Electronic Society* (New York, NY, 2010), WPES '10, ACM, pp. 73–82. pages 27, 149

[73] Díaz, C., Seys, S., Claessens, J., and Preneel, B. Towards Measuring Anonymity. In *PETs*, R. Dingledine and P. Syverson, Eds., vol. 2482 of *LNCS*. Springer, 2003, pp. 54–68. pages 24, 25

[74] Dingledine, R., Mathewson, N., and Syverson, P. Tor: The Second-generation Onion Router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13* (Berkeley, CA, USA, 2004), SSYM'04, USENIX Association, pp. 303–320. pages 2, 13, 52

[75] Dreier, J., Lafourcade, P., and Lakhnech, Y. Formal Verification of e-Auction Protocols. In *Principles of Security and Trust*, D. Basin and J. Mitchell, Eds., vol. 7796 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 247–266. pages 27

[76] Dwork, C. Differential Privacy. In *Automata, Languages and Programming*, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds., vol. 4052 of *LNCS*. Springer, 2006, pp. 1–12. pages 25

[77] Dwyer, C., Hiltz, S. R., and Passerini, K. Trust and Privacy Concern Within Social Networking Sites: A Comparison of Facebook and MySpace. In *Proceedings of the Thirteenth Americas Conference on Information Systems (AMCIS 2007)* (2007). Paper 339. pages 1, 69

[78] Eckersley, P. How Unique Is Your Web Browser? In *Privacy Enhancing Technologies*, M. Atallah and N. Hopper, Eds., vol. 6205 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010, pp. 1–18. pages 42

[79] Fassbender, S., and Heisel, M. From Problems to Laws in Requirements Engineering – Using Model-transformation. In *ICSOFT 2013 - Proceedings of the 8th International Joint Conference on Software Technologies, Reykjavık, Iceland, 29-31 July, 2013* (2013), pp. 447–458. pages 24

[80] Fassbender, S., and Heisel, M. A Computer-aided Process from Problems to Laws in Requirements Engineering. In *Software Technologies*, J. Cordeiro and M. van Sinderen, Eds., vol. 457 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg, 2014, pp. 215–234. pages 24

[81] Fazouane, M., Kopp, H., van der Heijden, R., Le Métayer, D., and Kargl, F. Formal Verification of Privacy Properties in Electric Vehicle Charging. In *Engineering Secure Software and Systems*, F. Piessens, J. Caballero, and N. Bielova, Eds., vol. 8978 of *Lecture Notes in Computer Science*. Springer International Publishing, 2015, pp. 17–33. pages 27

[82] Feige, U., Fiat, A., and Shamir, A. Zero-knowledge Proofs of Identity. *Journal of Cryptology 1*, 2 (1988), 77–94. pages 14

[83] Finzgar, L., and Trebar, M. Use of NFC and QR Code Identification in an Electronic Ticket System for Public Transport. In *Software, Telecommunications and Computer Networks (SoftCOM), 2011 19th International Conference on* (2011), IEEE, pp. 1–6. pages 106

[84] Foss, T. Safe and Secure Intelligent Transport Systems (ITS). In *Transport Research Arena (TRA) 5th Conference: Transport Solutions from Research to Deployment* (2014). pages 106

[85] Garg, D., Jia, L., and Datta, A. Policy Auditing over Incomplete Logs: Theory, Implementation and Applications. In *Proceedings of the 18th ACM Conference on Computer and Communications Security* (New York, NY, USA, 2011), Y. Chen, G. Danezis, and V. Shmatikov, Eds., CCS '11, ACM, pp. 151–162. pages 123

[86] Gebser, M., Kaminski, R., Kaufmann, B., and Schaub, T. Clingo = ASP + Control: Preliminary Report. *CoRR abs/1405.3694* (2014). pages 31

[87] Gelfond, M., and Lifschitz, V. The Stable Model Semantics for Logic Programming. In *Proceedings of International Logic Programming Conference and Symposium* (1988), R. Kowalski, Bowen, and Kenneth, Eds., MIT Press, pp. 1070–1080. pages 31

[88] Gelfond, M., and Lifschitz, V. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing 9*, 3-4 (1991), 365–385. pages 32

[89] Ghìron, S. L., Sposato, S., Medaglia, C. M., and Moroni, A. NFC Ticketing: A Prototype and Usability Test of a NFC-based Virtual Ticketing Application. In *Near Field Communication, 2009. NFC'09. First International Workshop on* (2009), IEEE, pp. 45–50. pages 106

[90] Goldschlag, D., Reed, M., and Syverson, P. Onion Routing. *Commun. ACM 42*, 2 (Feb. 1999), 39–41. pages 13

[91] Guagnin, D., Hempel, L., and Ilten, C. *Managing Privacy Through Accountability.* Palgrave Macmillan, 2012. pages 123

[92] Gürses, S. *Multilateral Privacy Requirements Analysis in Online Social Network Services.* PhD thesis, KU Leuven, Faculty of Engineering Science, Mai 2010. pages 10

[93] Gürses, S., Troncoso, C., and Díaz, C. Engineering Privacy by Design. *Computers, Privacy & Data Protection 14* (2011). pages 3, 17

[94] Hafiz, M. A Pattern Language for Developing Privacy Enhancing Technologies. *Software: Practice and Experience 43*, 7 (2013), 769–787. pages 18

[95] Hardt, D. The OAuth 2.0 Authorization Framework. RFC 6749, 2012. pages 11

[96] Heydt-Benjamin, T. S., Chae, H.-J., Defend, B., and Fu, K. Privacy for Public Transportation. In *Privacy Enhancing Technologies*, G. Danezis and P. Golle, Eds., vol. 4258 of *Lecture Notes in Computer Science.* 2006, pp. 1–19. pages 106

[97] Hoare, C. A. R. An Axiomatic Basis for Computer Programming. *Commun. ACM 12*, 10 (Oct. 1969), 576–580. pages 27

[98] Hoepman, J.-H. Privacy Design Strategies. In *ICT Systems Security and Privacy Protection*, N. Cuppens-Boulahia, F. Cuppens, S. Jajodia, A. Abou El Kalam, and T. Sans, Eds., vol. 428 of *IFIP Advances in Information and Communication Technology*. Springer Berlin Heidelberg, 2014, pp. 446–459. pages 18

[99] Hossain, M. A. Exploring the Perceived Measures of Privacy: RFID in Public Applications. *Australasian Journal of Information Systems 18*, 2 (2014). pages 106

[100] Howard, M., and Leblanc, D. E. *Writing Secure Code*, 2nd ed. Microsoft Press, Redmond, WA, USA, 2002. pages 21

[101] Jackson, D. Alloy: A Lightweight Object Modelling Notation. *ACM Trans. Softw. Eng. Methodol. 11*, 2 (Apr. 2002), 256–290. pages 33

[102] Jackson, D. *Software Abstractions: Logic, Language, and Analysis*. The MIT Press, 2006. pages 33

[103] Jackson, M. Problem Frames and Software Engineering. *Information and Software Technology 47*, 14 (2005), 903 – 912. Special Issue on Problem Frames. pages 23

[104] Jacobs, B. IRMA: I Reveal My Attributes. `https://www.irmacard.org/irma/`. [Online; accessed 20-04-2015]. pages 15

[105] Jafari, M., Fong, P. W., Safavi-Naini, R., Barker, K., and Sheppard, N. P. Towards Defining Semantic Foundations for Purpose-based Privacy Policies. In *Proceedings of the First ACM Conference on Data and Application Security and Privacy* (New York, NY, USA, 2011), CODASPY '11, ACM, pp. 213–224. pages 15

[106] Jafari, M., Safavi-Naini, R., Fong, P. W. L., and Barker, K. A Framework for Expressing and Enforcing Purpose-based Privacy Policies. *ACM Trans. Inf. Syst. Secur. 17*, 1 (Aug. 2014), 3:1–3:31. pages 15

[107] Jakubauskas, G. Improvement of Urban Passenger Transport Ticketing Systems by Deploying Intelligent Transport Systems. *Transport 21*, 4 (2006), 252–259. pages 106

[108] Jansen, J., Jorissen, A., and Janssens, G. Compiling Input* FO(.) Inductive Definitions into Tabled Prolog Rules for IDP3. *Theory and Practice of Logic Programming 13* (7 2013), 691–704. pages 147

[109] Jorns, O., Jung, O., and Quirchmayr, G. A Privacy Enhancing Service Architecture for Ticket-based Mobile Applications. In *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on* (2007), IEEE, pp. 139–146. pages 106

[110] Karpati, P., Redda, Y., Opdahl, A. L., and Sindre, G. Comparing Attack Trees and Misuse Cases in an Industrial Setting. *Information and Software Technology 56*, 3 (2014), 294 – 308. pages 23

[111] Lamport, L. *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers.* Addison-Wesley Longman Publishing Co., Inc., 2002. pages 33

[112] Lamport, L. The TLA Home Page. `http://research.microsoft.com/en-us/um/people/lamport/tla/tla.html`, 2014. [Online; accessed 26-03-2015]. pages 33

[113] Lano, K. *The B Language and Method: a Guide to Practical Formal Development.* Springer Science & Business Media, 2012. pages 33

[114] Le Métayer, D. Privacy by Design: a Formal Framework for the Analysis of Architectural Choices. In *Proceedings of the Third ACM Conference on Data and Application Security and Privacy* (New York, NY, USA, 2013), CODASPY '13, ACM, pp. 95–104. pages 28

[115] Lee, L., and Grimson, W. E. L. Gait Analysis for Recognition and Classification. In *IEEE International Conference on Automatic Face and Gesture Recognition* (2002), pp. 148–155. pages 124

[116] Lee, S., Turner, J., Daskin, M. S., de Mello, T. H., and Smilowitz, K. Improving Fleet Utilization for Carriers by Interval Scheduling. *European Journal of Operational Research 218*, 1 (2012), 261–269. pages 32

[117] Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., and Scarcello, F. The DLV System for Knowledge Representation and Reasoning. *ACM Trans. Comput. Logic 7*, 3 (July 2006), 499–562. pages 31

[118] Leuschel, M., and Butler, M. ProB: A Model Checker for B. In *FME 2003: Formal Methods*, K. Araki, S. Gnesi, and D. Mandrioli, Eds., vol. 2805 of *Lecture Notes in Computer Science.* Springer Berlin Heidelberg, 2003, pp. 855–874. pages 33

[119] Li, N., Li, T., and Venkatasubramanian, S. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on* (April 2007), pp. 106–115. pages 25

[120] Lorch, M., Proctor, S., Lepro, R., Kafura, D., and Shah, S. First Experiences Using XACML for Access Control in Distributed Systems. In *Proceedings of the 2003 ACM Workshop on XML Security* (New York, NY, USA, 2003), XMLSEC '03, ACM, pp. 25–37. pages 16

[121] Lund, M., Solhaug, B., and Stølen, K. A Guided Tour of the CORAS Method. In *Model-Driven Risk Analysis.* Springer Berlin Heidelberg, 2011, pp. 23–43. pages 21

[122] Machanavajjhala, A., Kifer, D., Gehrke, J., and Venkitasubramaniam, M. l-Diversity: Privacy Beyond k-Anonymity. *ACM Trans. Knowl. Discov. Data 1*, 1 (Mar. 2007). pages 25

[123] McCarthy, J., and Hayes, P. J. Some Philosophical Problems from the Standpoint of Artificial Intelligence. *Readings in Artificial Intelligence* (1969), 431–450. pages 147

[124] McLachlan, J., Tran, A., Hopper, N., and Kim, Y. Scalable Onion Routing with Torsk. In *Proceedings of the 16th ACM Conference on Computer and Communications Security* (New York, NY, USA, 2009), CCS '09, ACM, pp. 590–599. pages 14

[125] Mecocci, A., Pannozzo, M., and Fumarola, A. Automatic Detection of Anomalous Behavioural Events for Advanced Real-time Video Surveillance. In *IEEE International Symposium on Computational Intelligence for Measurement Systems and Applications (CIMSA'03)* (2003), pp. 187–192. pages 124

[126] Menezes, A. J., Vanstone, S. A., and Oorschot, P. C. V. *Handbook of Applied Cryptography*, 1st ed. CRC Press, Inc., Boca Raton, FL, USA, 1996. pages 9

[127] Microsoft. Simplified Implementation of the Microsoft SDL – Version 1. `http://www.microsoft.com/security/sdl/`, 2011. [Online; accessed 24-03-2015]. pages 20

[128] Milutinovic, M., Dacosta, I., Put, A., and De Decker, B. An Efficient and Unlinkable Incentives Scheme. Tech. Rep. CW659, Dept. Computer Science, KU Leuven, 2014. pages 93

[129] Milutinovic, M., Decroix, K., Naessens, V., and De Decker, B. Privacy-preserving Public Transport Ticketing System. In *Data and Applications Security and Privacy XXIX*, P. Samarati, Ed., vol. 9149 of *Lecture Notes in Computer Science*. Springer International Publishing, 2015, pp. 135–150. pages xx, 91, 105, 107, 110, 111

[130] MOBIB. MOBIB. `http://www.mobib.be/mobib-card_EN.htm`, 2015. [Online; accessed 30-06-2015]. pages 93

[131] Murdoch, S., and Danezis, G. Low-cost Traffic Analysis of Tor. In *Security and Privacy, 2005 IEEE Symposium on* (May 2005), pp. 183–195. pages 26

[132] Naessens, V., and De Decker, B. A Methodology for Designing Controlled Anonymous Applications. In *Security and Privacy in Dynamic Environments*, S. Fischer-Hübner, K. Rannenberg, L. Yngström, and S. Lindskog, Eds., vol. 201 of *IFIP International Federation for Information Processing*. Springer US, 2006, pp. 111–122. pages 28

[133] Navigo. Navigo. `http://www.navigo.fr`, 2015. [Online; accessed 30-06-2015]. pages 93

[134] Nethercote, N., Stuckey, P., Becket, R., Brand, S., Duck, G., and Tack, G. MiniZinc: Towards a Standard CP Modelling Language. In *Principles and Practice of Constraint Programming – CP 2007*, C. Bessiere, Ed., vol. 4741 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2007, pp. 529–543. pages 32

[135] Norris, J. R. *Markov Chains*. No. 2. Cambridge University Press, 1998. pages 32

[136] NXP Semiconductors. MIFARE Standard 4KByte Card IC Functional Specification, 2012. pages 93, 106

[137] Opdahl, A. L., and Sindre, G. Experimental Comparison of Attack Trees and Misuse Cases for Security Threat Identification. *Information and Software Technology 51*, 5 (2009), 916 – 932. Special Issue: Model-driven Development for Secure Information Systems. pages 23

[138] Organisation for Economic Co-operation and Development. The OECD Privacy Framework. `http://www.oecd.org`, 2013. [Online; accessed 30-04-2015]. pages 2

[139] Paquin, C. U-Prove Technology Overview V1.1 Draft Revision 1. Microsoft Corporation. pages 2, 14, 42

[140] Pardo, R., and Schneider, G. A Formal Privacy Policy Framework for Social Networks. In *Software Engineering and Formal Methods*, D. Giannakopoulou and G. Salaün, Eds., vol. 8702 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 378–392. pages 27

[141] Pearson, S. Privacy Management in Global Organisations. In *Communications and Multimedia Security*, B. Decker and D. Chadwick, Eds., vol. 7394 of *LNCS*. Springer, 2012, pp. 217–237. pages 20

[142] Pelletier, M.-P., Trépanier, M., and Morency, C. Smart Card Data Use in Public Transit: A Literature Review. *Transportation Research Part C: Emerging Technologies 19*, 4 (2011), 557–568. pages 106, 107

[143] Pfitzmann, A., and Hansen, M. A Terminology for Talking about Privacy by Data Minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management, Aug 2010. v0.34. pages 24, 42, 86, 95

[144] Pulls, T., Peeters, R., and Wouters, K. Distributed Privacy-preserving Transparency Logging. In *Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society* (New York, NY, USA, 2013), WPES '13, ACM, pp. 83–94. pages 122

[145] Put, A., Dacosta, I., Milutinovic, M., and De Decker, B. PriMan: Facilitating the Development of Secure and Privacy-preserving Applications. In *ICT Systems Security and Privacy Protection*, N. Cuppens-Boulahia, F. Cuppens, S. Jajodia, A. Abou El Kalam, and T. Sans, Eds., vol. 428 of *IFIP Advances in Information and Communication Technology*. Springer Berlin Heidelberg, 2014, pp. 403–416. pages 18

[146] Rebollo-Monedero, D., Parra-Arnau, J., Díaz, C., and Forné, J. On the Measurement of Privacy as an Attacker's Estimation Error. *International Journal of Information Security 12*, 2 (2013), 129–149. pages 26

[147] Recordon, D., and Reed, D. OpenID 2.0: A Platform for User-centric Identity Management. In *Proceedings of the Second ACM Workshop on Digital Identity Management* (New York, NY, USA, 2006), DIM '06, ACM, pp. 11–16. pages 11

[148] Reiter, M. K., and Rubin, A. D. Crowds: Anonymity for Web Transactions. *ACM Trans. Inf. Syst. Secur. 1*, 1 (Nov. 1998), 66–92. pages 14, 24

[149] Reiter, R. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems.* MIT press, 2001. pages 147

[150] Rodrigues, R., Barnard-Wills, D., Wright, D., De Hert, P., and Papakonstantinou, V. EU Privacy Seals Project. *Publications Office of the European Union* (2013). pages 19

[151] Sabouri, A. D2.2 Architecture for Attribute-based Credential Technologies – Final Version, ABC4Trust Deliverable. `https://abc4trust.eu/`, 2014. [Online; accessed 23-03-2015]. pages 14

[152] Sagonas, K., Swift, T., and Warren, D. S. XSB as an Efficient Deductive Database Engine. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 1994), SIGMOD '94, ACM, pp. 442–453. pages 147

[153] Schimmer, L. Peer Profiling and Selection in the I2P Anonymous Network. In *Extended Abstracts of the Fourth Privacy Enhancing Technologies Convention (PET-CON 2009.1).–Dresden: TU, Fak. Informatik* (2009), pp. 59–70. pages 14

[154] Schneier, B. Attack Trees. *Dr. Dobb's journal 24*, 12 (1999), 21–29. pages 23

[155] Sen, S., Guha, S., Datta, A., Rajamani, S., Tsai, J., and Wing, J. Bootstrapping Privacy Compliance in Big Data Systems. In *Security and Privacy (SP), 2014 IEEE Symposium on* (May 2014), pp. 327–342. pages 123

[156] Serjantov, A., and Danezis, G. Towards an Information Theoretic Metric for Anonymity. In *Proceedings of the 2nd International Conference on Privacy Enhancing Technologies* (Berlin, Heidelberg, 2003), R. Dingledine and P. Syverson, Eds., PET'02, Springer-Verlag, pp. 41–53. pages 25, 87

[157] Shannon, C. E. A Mathematical Theory of Communication. *SIGMOBILE Mob. Comput. Commun. Rev. 5*, 1 (Jan. 2001), 3–55. pages 24

[158] Shin, D.-H. The Effects of Trust, Security and Privacy in Social Networking: A Security-based Approach to Understand the Pattern of Adoption. *Interacting with Computers 22*, 5 (2010), 428 – 438. Modelling User Experience – An Agenda for Research and Practice. pages 1

[159] Sindre, G., and Opdahl, A. Eliciting Security Requirements with Misuse Cases. *Requirements Engineering 10*, 1 (2005), 34–44. pages 23

[160] Spiekermann, S. The Challenges of Privacy by Design. *Commun. ACM 55*, 7 (July 2012), 38–40. pages 17

[161] Stufflebeam, W. H., Antón, A. I., He, Q., and Jain, N. Specifying Privacy Policies with P3P and EPAL: Lessons Learned. In *Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society* (New York, NY, USA, 2004), WPES '04, ACM, pp. 35–35. pages 16

[162] Sweeney, L. k-Anonymity: a Model for Protecting Privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems 10*, 5 (Oct. 2002), 557–570. pages 25

[163] Syverson, P. Why I'm Not an Entropist. In *Security Protocols XVII*, B. Christianson, J. Malcolm, V. Matyás, and M. Roe, Eds., vol. 7028 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 213–230. pages 26

[164] Ta, V., and Antignac, T. Privacy by Design: On the Conformance Between Protocols and Architectures. *CoRR abs/1501.03593* (2015). pages 28

[165] Talukder, A., Maurya, V., Santhosh, B., Jangam, E., Muni, S., Jevitha, K., Saurabh, S., and Pais, A. Security-aware Software Development Life Cycle (SaSDLC) – Processes and Tools. In *Wireless and Optical Communications Networks, 2009. WOCN '09. IFIP International Conference on* (April 2009), pp. 1–5. pages 23

[166] The Smart Card Alliance: Hong Kong Octopus Card. The Smart Card Alliance, 2006. pages 106

[167] The Smart Card Alliance: Smart Card Talk Standards. The Smart Card Alliance, 2006. January issue. pages 106

[168] Tndel, I., Jensen, J., and Rstad, L. Combining Misuse Cases with Attack Trees and Security Activity Models. In *Availability, Reliability, and Security, 2010. ARES '10 International Conference on* (Feb 2010), pp. 438–445. pages 23

[169] Tóth, G., Hornák, Z., and Vajda, F. Measuring Anonymity Revisited. In *Proceedings of the Ninth Nordic Workshop on Secure IT Systems* (2004), Espoo, Finland, pp. 85–90. pages 25

[170] Trans Link Systems BV. OV-chipkaart. `https://www.ov-chipkaart.nl/aanvragen`, 2015. [Online; accessed 30-06-2015]. pages 92

[171] Transport for London. Oyster Online. `https://account.tfl.gov.uk/oyster`, 2015. [Online; accessed 30-06-2015]. pages 93

[172] Tschantz, M. C., Datta, A., and Wing, J. M. Formalizing and Enforcing Purpose Restrictions in Privacy Policies. In *IEEE Symposium on Security and Privacy* (2012), IEEE Computer Society, pp. 176–190. pages 27, 53

[173] Tschantz, M. C., Datta, A., and Wing, J. M. Purpose Restrictions on Information Use. In *Computer Security: ESORICS 2013*, J. Crampton, S. Jajodia, and K. Mayes, Eds., vol. 8134 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 610–627. pages 123

[174] U.S. DEPARTMENT OF COMMERCE. Safe Harbor Privacy Principles. `http://www.export.gov/safeharbor/eu/eg_main_018475.asp`, 2000. [Online; accessed 26-03-2015]. pages 3

[175] U.S. Federal Trade Commission. Privacy Online – a Report to Congres, 1998. pages 17

[176] van Blarkom, G. W., Borking, J. J., and Olk, J. G. E. Handbook of Privacy and Privacy-enhancing Technologies: the Case of Intelligent Software Agents. Tech. rep., Privacy Incorporated Software Agent Consortium, Den Haag, 2003. pages 13

[177] Van Gelder, A., Ross, K. A., and Schlipf, J. S. The Well-founded Semantics for General Logic Programs. *J. ACM 38*, 3 (July 1991), 619–649. pages 30

[178] Veeningen, M., de Weger, B., and Zannone, N. Data Minimisation in Communication Protocols: a Formal Analysis Framework and Application to Identity Management. *International Journal of Information Security 13*, 6 (2014), 529–569. pages 27

[179] Verslype, K., De Decker, B., Naessens, V., Nigusse, G., Lapon, J., and Verhaeghe, P. A Privacy-preserving Ticketing System. In *Data and Applications Security XXII*, V. Atluri, Ed. Springer, 2008, pp. 97–112. pages 106

[180] Viola, P., and Jones, M. Robust Real-Time Face Detection. *International Journal of Computer Vision 57*, 2 (2004), 137–154. pages 124

[181] Vullers, P., and Alpár, G. Efficient Selective Disclosure on Smart Cards Using Idemix. In *Policies and Research in Identity Management*, S. Fischer-Hübner, E. de Leeuw, and C. Mitchell, Eds., vol. 396 of *IFIP Advances in Information and Communication Technology*. Springer Berlin Heidelberg, 2013, pp. 53–67. pages 15

[182] W3C. Platform for Privacy Preferences Project. `http://www.w3.org/P3P/`, Feb. 2015. [Online; accessed 24-09-2015]. pages 15, 53, 69

[183] Weitzner, D. J., Abelson, H., Berners-Lee, T., Feigenbaum, J., Hendler, J., and Sussman, G. J. Information Accountability. *Commun. ACM 51*, 6 (2008), 82–87. pages 122

[184] Winters, N. Personal Privacy and Popular Ubiquitous Technology. *Proceedings of Ubiconf* (2004). pages 106

[185] Wittocx, J., Mariën, M., and Denecker, M. The IDP System: a Model Expansion System for an Extension of Classical Logic. In *Proceedings of the 2nd Workshop on Logic and Search, Logic and Search* (2008), M. Denecker, Ed., ACCO, pp. 153–165. pages 30, 59

[186] Wolsey, L. A. *Integer Programming*, vol. 42. Wiley New York, 1998. pages 32

[187] Woodcock, J., and Davies, J. *Using Z: Specification, Refinement, and Proof.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996. pages 33

[188] Wright, D., and de Hert, P. Introduction to Privacy Impact Assessment. In *Privacy Impact Assessment*, D. Wright and P. Hert, Eds. Springer, 2012, pp. 3–32. pages 123

[189] Wuyts, K., Scandariato, R., and Joosen, W. Empirical Evaluation of a Privacy-focused Threat Modeling Methodology. *Journal of Systems and Software 96*, 0 (2014), 122 – 138. pages 23

[190] Wuyts, K., Scandariato, R., and Joosen, W. LIND(D)UN Privacy Threat Tree Catalog. CW Reports CW675, Department of Computer Science, KU Leuven, September 2014. pages 23

[191] Yang, J., Yessenov, K., and Solar-Lezama, A. A Language for Automatically Enforcing Privacy Policies. In *Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (New York, NY, USA, 2012), POPL '12, ACM, pp. 85–96. pages 27

[192] Young, A. L., and Quan-Haase, A. Information Revelation and Internet Privacy Concerns on Social Network Sites: a Case Study of Facebook. In *Proceedings of the Fourth International Conference on Communities and Technologies* (New York, NY, USA, 2009), C&T '09, ACM, pp. 265–274. pages 1, 69

[193] Zimmeck, S., and Bellovin, S. M. Privee: An Architecture for Automatically Analyzing Web Privacy Policies. In *23rd USENIX Security Symposium (USENIX Security 14)* (San Diego, CA, Aug. 2014), USENIX Association, pp. 1–16. pages 15

List of Publications

Article in academic book, internationally recognised scientific publisher

Milutinovic, M., Decroix, K., Naessens, V., Bart, D. (2013). Commercial Home Assistance Services. In: Cruz-Cunha M., Miranda I., Gonçalves P. (Eds.), bookseries: Handbook of Research on ICTs and Management Systems for Improving Efficiency in Healthcare and Social Care, vol: 1, Handbook of Research on ICTs and Management Systems for Improving Efficiency in Healthcare and Social Care, Chapt. 8. Hershey: IGI Global, 156-179.

Milutinovic, M., Decroix, K., Naessens, V., De Decker, B. (2013). Commercial Home Assistance Services. In: Cruz-Cunha M., Miranda I. (Eds.), Handbook of Research on ICTs and Management Systems for Improving Efficiency in Healthcare and Social Care, Chapt. 8 IGI Global, 156-179.

Papers at international scientific conferences and symposia, published in full in proceedings

Milutinovic, M., Decroix, K., Naessens, V., De Decker, B. (2015). Privacy-preserving Public Transport Ticketing System. In Samarati, P. (Ed.), Data and Applications Security and Privacy XXVIII - 29th Annual IFIP WG 11.3 Working Conference, DBSec 2015 FairFax, VA ,USA, July 13-15, 2015 (pp. 135-150). Springer International Publishing.

Decroix, K., Lapon, J., Lemaire, L., De Decker, B., Naessens, V. (2015). Formal Reasoning about Privacy and Trust in Loyalty Systems. In Abramowicz, W. (Ed.), BIS 2015 Workshop Post-conference Proceedings. PTDCS 2015. Poznań, 24-26 June 2015 Springer. (Published in November 2015)

Decroix, K., Butin, D., Jansen, J., Naessens, V. (2014). Inferring Accountability from Trust Perceptions (16% acceptance rate). In Prakash, A. (Ed.), Shyamasundar, R. (Ed.), Information Systems Security: Vol. 8880. ICISS 2014. Hyderabad, 16-20 December 2014 (pp. 69-88) Springer-Verlag.

Decroix, K., Lapon, J., De Decker, B., Naessens, V. (2013). A Framework for Formal Reasoning about Privacy Properties Based on Trust Relationships in Complex Electronic Services (24% acceptance rate). In Bagchi, A. (Ed.), Ray, I. (Ed.), Information Systems Security: Vol. 8303. ICISS 2013. Kolkata, 16-20 December 2013 (pp. 106-120). Berlin Heidelberg: Springer-Verlag.

Decroix, K., Lapon, J., De Decker, B., Naessens, V. (2013). A Formal Approach for Inspecting Privacy and Trust in Advanced Electronic Services (24% acceptance rate). In Jürgens, J. (Ed.), Livshits, B. (Ed.), Scandariato, R. (Ed.), Engineering Secure Software and Systems: Vol. 7781 (5). ESSoS 2013. Paris, 27 February 2013 - 1 March 2013 (pp. 155-170). Berlin Heidelberg: Springer-Verlag.

Decroix, K., Seghers, B., Naessens, V. (2012). Managing Mobile Applications on Top of Alternative Java Virtual Machines. In De Strycker, L. (Ed.), Proceedings of the Fifth European Conference on the Use of Modern Information and Communication Technologies. ECUMICT. Ghent, 22-23 March 2012 (pp. 197-207). Ghent: Nevelland Graphics cvba-so.

Milutinovic, M., Decroix, K., Naessens, V., De Decker, B. (2012). Commercial Home Assistance (eHealth) Services. In Camenisch, J. (Ed.), Kesdogan, D. (Ed.), Proceedings of the IFIP WG 11.4 Workshop on Open Problems in Network Security (iNetSec 2011): Vol. 7039. iNetSec 2011. Lucerne, Switzerland, 9 June 2011 (pp. 28-42) Springer Verlag.

Decroix, K., Milutinovic, M., De Decker, B., Naessens, V. (2011). A Generic Architecture for Integrating Health Monitoring and Advanced Care Provisioning. In De Decker, B. (Ed.), Lapon, J. (Ed.), Naessens, V. (Ed.), Uhl, A. (Ed.), Communications and Multimedia Security: Vol. 7025 (12). CMS 2011. Ghent, Belgium, 19-21 October 2011 (pp. 163-170). Berlin, Heidelberg: Springer.

Meeting abstracts, presented at international scientific conferences and symposia, published or not published in proceedings or journals

Decroix, K., De Decker, B., Naessens, V. (2011). Designing Privacy-enhancing Mobile Applications. Procedings of PrimeLife International Summer School. IFIP Summer School 2011. Trento, 5-9 September 2011, 157-170.

Internal report

Milutinovic, M., Decroix, K., Naessens, V., De Decker, B. (2011). Commercially-run Home Assistance Centres. CW Reports, CW612, 33 pp. Leuven, Belgium: Department of Computer Science, KU Leuven.

External reports: reports by order of - or published by - an external organisation

Milutinovic, M., Decroix, K., Naessens, V., De Decker, B. (2012). Privacy Preserving Mechanisms for a Pervasive eHealth System. Proceedings of the 7th International Summer School Organised Jointly by the IFIP Working Group 9.2, 9.6/11.7, 11.4, 11.6: Springer.

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
IMINDS-DISTRINET
Celestijnenlaan 200A box 2402
B-3001 Heverlee
koen.decroix@cs.kuleuven.be
http://wms.cs.kuleuven.be/cs/