Draft requirements for next generation policies

Editors: Carine Bournez (W3C)
        Gregory Neven (IBM)
Reviewers: Markulf Kohlweiss (KUL)
           Martin Pekárek (TILT)
Identifier: H5.1.1
Type: Heartbeat
Class: Public
Date: 11 December 2008
Status: Draft version.
        Final version in July 2009.

Abstract

One of the core activities in the PrimeLife project is the design and implementation of a versatile privacy policy language. Policy languages are a key element in any privacy-aware information infrastructure. Machine-interpretable languages have the major advantage over natural languages that, if designed properly, they allow automated negotiation, reasoning, composition, and enforcement of policies. This document provides the first step in the development of such a language. It describes a number of relevant use case scenarios involving privacy policies and derives from these a list of requirements that should be met by the PrimeLife policy language.

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 216483 for the project PrimeLife.
# Members of the PrimeLife Consortium

1. IBM Research GmbH
2. Unabhängiges Landeszentrum für Datenschutz
3. Technische Universität Dresden
4. Karlstads Universitet
5. Università degli Studi di Milano
6. Johann Wolfgang Goethe - Universität Frankfurt am Main
7. Stichting Katholieke Universiteit Brabant
8. GEIE ERCIM
9. Katholieke Universiteit Leuven
10. Università degli Studi di Bergamo
11. Giesecke & Devrient GmbH
12. Center for Usability Research & Engineering
13. Europäisches Microsoft Innovations Center GmbH
14. SAP AG
15. Brown University

<table>
<thead>
<tr>
<th>Consortium Member</th>
<th>Abbreviation</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Research GmbH</td>
<td>IBM</td>
<td>Switzerland</td>
</tr>
<tr>
<td>Unabhängiges Landeszentrum für Datenschutz</td>
<td>ULD</td>
<td>Germany</td>
</tr>
<tr>
<td>Technische Universität Dresden</td>
<td>TUD</td>
<td>Germany</td>
</tr>
<tr>
<td>Karlstads Universitet</td>
<td>KAU</td>
<td>Sweden</td>
</tr>
<tr>
<td>Università degli Studi di Milano</td>
<td>UNIMI</td>
<td>Italy</td>
</tr>
<tr>
<td>Johann Wolfgang Goethe - Universität Frankfurt am Main</td>
<td>GUF</td>
<td>Germany</td>
</tr>
<tr>
<td>Stichting Katholieke Universiteit Brabant</td>
<td>TILT</td>
<td>Netherlands</td>
</tr>
<tr>
<td>GEIE ERCIM</td>
<td>W3C</td>
<td>France</td>
</tr>
<tr>
<td>Katholieke Universiteit Leuven</td>
<td>K.U.Leuven</td>
<td>Belgium</td>
</tr>
<tr>
<td>Università degli Studi di Bergamo</td>
<td>UNIBG</td>
<td>Italy</td>
</tr>
<tr>
<td>Giesecke &amp; Devrient GmbH</td>
<td>GD</td>
<td>Germany</td>
</tr>
<tr>
<td>Center for Usability Research &amp; Engineering</td>
<td>CURE</td>
<td>Austria</td>
</tr>
<tr>
<td>Europäisches Microsoft Innovations Center GmbH</td>
<td>EMIC</td>
<td>Germany</td>
</tr>
<tr>
<td>SAP AG</td>
<td>SAP</td>
<td>Germany</td>
</tr>
<tr>
<td>Brown University</td>
<td>UBR</td>
<td>USA</td>
</tr>
</tbody>
</table>

Disclaimer: The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law. Copyright 2008 by IBM Research GmbH, Unabhängiges Landeszentrum für Datenschutz, Università degli Studi di Milano, GEIE ERCIM, Katholieke Universiteit Leuven, Università degli Studi di Bergamo, Europäisches Microsoft Innovations Center GmbH, SAP AG.
List of Contributors

Contributions from several PrimeLife partners are contained in this document. Individual participants in the Activity 5 are (at the time of writing):

Claudio Ardagna (UNIMI), Carine Bournez (W3C), Laurent Bussard (EMIC), Michele Bezzi (SAP), Jan Camenisch (IBM), Aleksandra Kuczerawy (KUL), Sebastian Meissner (ULD), Gregory Neven (IBM), Stefano Paraboschi (UNIBG), Eros Pedrini (UNIMI), Ulrich Pinsdorf (EMIC), Franz-Stefan Preiss (IBM), Slim Trabelsi (SAP), Christina Tziviskou (UNIBG), Pierangela Samarati (UNIMI), Jan Schallaboeck (ULD), Stuart Short (SAP), Dieter Sommer (IBM), Thomas Roessler (W3C), Sabrina de Capitani di Vimercati (UNIMI), Mario Verdicchio (UNIBG), Rigo Wenning (W3C)

The authors would like to acknowledge other PrimeLife partners and external groups for their contributions to the scenarios and use cases: TILT, TUD, the W3C Policy Languages Interest Group (PLING).

This deliverable was rendered from HTML pages using Prince XML from YesLogic Pty Ltd. YesLogic has donated a license of Prince XML to W3C.
# Table Of Contents

## 1 Introduction
- 1.1 About this document .......................................... 6
- 1.2 Liaisons to other groups ....................................... 6
  - 1.2.1 W3C PLING: The Policy Languages Interest Group .... 6
  - 1.2.2 ISO/IEC JTC 1/SC 27/WG 5 on Identity Management and Privacy Technologies

## 2 Scenarios
- 2.1 UC1. Wiki scenario ........................................... 10
  - 2.1.1 UC1.1. Register to the Wiki ................................ 10
  - 2.1.2 UC1.2. Edit policy for number of sites .................. 10
  - 2.1.3 UC1.3. Create a new site with policy ..................... 11
  - 2.1.4 UC1.4. Access Control based on reputation ............... 11
- 2.2 UC2. Blog scenario ........................................... 12
  - 2.2.1 UC2.1. Creating a blog .................................. 12
  - 2.2.2 UC2.2. Posting an entry .................................. 12
  - 2.2.3 UC2.3. Posting a blog comment ........................... 13
  - 2.2.4 UC2.4. Reading an entry .................................. 13
- 2.3 UC3. Enterprise workflow scenarios ......................... 13
  - 2.3.1 UC3a. Computer part workflow ............................ 13
  - 2.3.2 UC3b. Company HR workflow ............................... 14
- 2.4 UC4. Social Network Sites .................................... 14
  - 2.4.1 UC4.1. Register to an SNS ................................ 15
  - 2.4.2 UC4.2. Establish connections .............................. 15
  - 2.4.3 UC4.3. Post content to own profile ....................... 15
  - 2.4.4 UC4.4. Post content to another profile .................. 16
  - 2.4.5 UC4.5. Control access to and use of SNS content (profile information or other content) .............. 16
  - 2.4.6 UC4.6. Report on access to and use of SNS content (profile information or other content) ......................... 16
  - 2.4.7 UC4.7. Transfer profile to another SNS (data portability) .................................................... 17
  - 2.4.8 UC4.8. Integration of mobile SNS applications ........... 17
  - 2.4.9 UC4.9. Close an SNS profile ................................ 17
- 2.5 UC5. Anonymous credentials and identity management ........ 17
  - 2.5.1 UC5.1 Login use case .................................... 18
  - 2.5.2 UC5.2 Buying use case .................................... 19
  - 2.5.3 UC5.3 Gift certificate use case ............................ 20
  - 2.5.4 System overview .......................................... 21
- 2.6 UC6. User Interface requirements ............................. 23
  - 2.6.1 UC6.1. Trust and assurance HCI ........................... 23
  - 2.6.2 UC6.2. User interfaces for policy display and administration ........................................... 23
- 2.7 UC7. Services Composition Scenarios ......................... 24
  - 2.7.1 UC7.1. ePortfolio scenario ................................ 24
  - 2.7.2 UC7.2. Data Flow in Travel Booking ...................... 25
  - 2.7.3 UC7.3 Data Flow in Human Resources ..................... 26
2.7.4 UC7.4. Privacy tradeoff in search/discovery ........................................ 26
2.8 UC12. Logging/Auditing Use Case ......................................................... 27
2.9 UC13. Privacy of location-based information ........................................ 27

3 Other perspectives ..................................................................................... 29
3.1 UC8. Privacy Policy Management ......................................................... 29
3.2 UC9. Federated Policy Management .................................................... 30
3.3 UC10. Authentication Strength Policies ................................................ 31
3.4 UC11. Corporate Security Policies ....................................................... 32

4 Requirements ............................................................................................. 33
4.1 General principles .................................................................................. 33
  4.1.1 General principles for Data Handling .............................................. 33
  4.1.2 General principles for Access Control ............................................ 34
4.2 Requirements for language model and expressivity ............................... 35
  4.2.1 Requirements for Data Handling policies ....................................... 35
  4.2.2 Requirements for Access Control policies ...................................... 38
  4.2.3 Requirements for Trust policies ...................................................... 41
4.3 Requirements for Policy Composition .................................................. 43
  4.3.1 Composition of Data Handling policies ........................................... 43
  4.3.2 Composition of Access Control policies ........................................ 44
  4.3.3 Composition and Trust policies ....................................................... 45
4.4 Requirements for use of anonymous credentials ................................... 45
Chapter 1

Introduction

1.1 About this document

The design and implementation of a versatile privacy policy language is one of the core activities in the PrimeLife project. Policy languages are a crucial tool in any privacy-aware information infrastructure. Machine-interpretable languages have the major advantage over natural languages that, if designed properly, they allow automated negotiation, reasoning, composition, and enforcement of policies. This document provides the first step in the development of such a language. This document is the result of a joint effort of all partners involved in Activity 5 of the PrimeLife project to collect use case scenarios and derive from these concrete requirements for the language to be designed.

1.2 Liaisons to other groups

1.2.1 W3C PLING: The Policy Languages Interest Group

While collecting requirements for the PrimeLife policy language, a connection was made to the W3C Policy Languages Interest Group. This group was started following a W3C Workshop on Languages for Privacy Policy Negotiation and Semantics-Driven Enforcement. This workshop was organised with the help of the PRIME project. Participants found out that there is already a great variety of very specific policy languages but that those do not work together. PLING is supposed -among other things- to serve as a platform for exchanges about the different policy languages. Even after the end of the PRIME project, some PRIME actors, together with stakeholders from industry and academia, continued working in PLING. It has started to listen to the industry and to collect use cases, information on existing policy languages and remarkable cases. At the W3C Technical Plenary meeting in Mandelieu in 2008, members of PLING had a joint meeting with participants of PrimeLife Activity 5 to think about further cooperation.
1.2.2 ISO/IEC JTC 1/SC 27/WG 5 on Identity Management and Privacy Technologies

As an official liaison partner to ISO/IEC JTC 1/SC 27/WG 5 “Identity Management and Privacy Technologies”, PrimeLife is actively contributing to the development of standards in this Working Group. Hans Hedbom from Karlstad University was appointed Liaison Officer. Kai Rannenberg, the Convener of SC 27/WG 5, is the PrimeLife coordinator for partner Johann Wolfgang Goethe University in Frankfurt/Main. Jan Schallaböck from Unabhängiges Landeszentrum für Datenschutz, a PrimeLife partner, is the Secretary of SC 27/WG 5.

Those close relations ensure that requirements stemming from the work of ISO/IEC JTC1/SC 27/WG 5 on Privacy and Identity Management are taken into account by PrimeLife, and equally that research results of PrimeLife find their way into the work on the standards. Of special interest is the work on ISO/IEC 24760 Identity Management Framework, ISO/IEC 29100 Privacy Framework and ISO/IEC 29101 Privacy Reference Architecture.

1.3 Legal requirements

The processing (including collection, storage, retrieval, transferral, and other means of handling) of any data that can be linked to a person (personal data, for a more thorough definition see Article 29 WP, Op. 136 [Art.29 Opinion 136]) by another entity (a data processor) has to be legitimate. If processing takes place without obeying legitimacy those subjected to the processing (data subjects) would lose trust in markets and tend to not give away their data when acting in these markets. Data protection thus is a market enabler, but above all it is recognized as a fundamental right, and is acknowledged by many constitutions in the European Union, as well as in the Charter of Fundamental Rights of the European Union (cf. Art. 8 thereof) and the European Convention on Human Rights (Art. 8 thereof).

On the operational level the Directive 95/46/EC [Directive 95/46/EC] on the protection of personal data (Data Protection Directive), and the Directive 2002/58/EC [Directive 2002/58/EC] on Privacy and Electronic Communications (E-Privacy Directive) provide the baseline for compliance. In many areas sector specific regulation and national implementation of directives need to be taken into account. Both directives as well as most of the other regulation follow a set of well established principles, with the principle of fair and lawful processing, the purpose limitation principle, data minimization, and the transparency principle at their core to name a few.

Fair and lawful processing

Conceptually European law effectively prohibits any processing except where there is a legal basis (Art. 6 of 95/46/EC). This means that in a professional context handling data without a legal basis is illegal. Noncompliance can result in penalties and may even lead to data protection authorities shutting down IT systems (see § 38 of the German “Bundesdatenschutzgesetz”). A legal basis can be derived either from legal regulation directly, eg. when the law prescribes the storage of specific data for example for law enforcement purposes. In many cases however the lawfulness can be achieved by receiving a consent, in an unambiguous form, from the person whose data is concerned, i.e. from the data subject (Art. 7(a) of 95/46/EC).

Purpose limitation
The processing of personal data - even if acquired lawfully (which should result in legitimacy) - is still a subject to further regulation. It regularly needs to follow, amongst others, the principle of purpose limitation (Art. 6.1(b) of 95/46/EC). Put simply, the purpose limitation principle states, that data may only be collected, stored, processed or transferred for those purposes the data subject has given consent to, or the law allows for. No further processing that would be incompatible with the original purpose is allowed.

Data minimization

This includes, that if no purpose is at hand, the data has to be deleted or not even collected in the first place (Art. 6.1(c) 95/46/EC). But data minimization can be understood even in a broader sense to construct systems in such a way, that a processing of personal data can be avoided (so called data avoidance). While the former is a legal requirement, the latter is not mandatory by law in all cases, but only where such technology is available under reasonable conditions (cf. Recital 46 of 95/46/EC). However, it can be sensible to develop and use such approaches even in an enterprise context when there is no legal necessity, as it may lower compliance costs.

Transparency and subject access rights

The principles regarding the processing itself are adjunct by specific, enforceable rights for the data subject (e.g. Art. 12 and 14 of 95/46/EC). The conceptual idea behind these rights is that the data subject should be able to find out what others know about him or her. In case this knowledge is illegitimate, the data subject should be able to stop the respective data controller from using this knowledge, by blocking, correction or deletion of the personal data.

In information systems the protection of any data does not always prove to be easy. This is especially true for the protection of personal data. A core difficulty lies in the diversity of processing step this data may undergo, while at the same time being subjected to the above principles. For compliance it has to be ensured that any algorithm, any service of an IT system, that processes a specific set or piece of personal data, is within the limits of the legal foundation (e.g. the consent) for the processing, and does not violate the purpose limitation principle. At the same time, it needs to be ensured that the data subject is able to find out what happened to his or her data, who accessed it, and what it has been or will be used for.

This deliverable on policy requirements collects a thorough set of requirements from a number of use cases to facilitate compliance and enhance privacy protection of the user.

1.4 Definitions of policies

We first define three types of policies that, in our view, are important parts of any privacy policy: data handling, access control, and trust policies. This by no means implies that we consider these to be separate, independent policies that together form the privacy policy. Rather, we see them as three minimal aspects that have to be covered by any policy language. There may be other aspects, and the three aspects mentioned here may not be orthogonal. In fact, the exact relationship between these three policy types will be a topic of intense research in WP5.2.
1.4.1 Data Handling policies

A data handling policy (DHP) is a set of rules stating how a piece of sensitive data should be treated. In the context of privacy, we are mostly interested in the case where that piece of data is personally identifiable information (PII). The data handling policy specifies, amongst other things, for what purposes the data can be used (e.g. research, marketing), to which third parties the data can be disclosed (e.g. all, nobody, only auditors), how long the data can be stored, etc. server-side data handling policies are sometimes referred to as obligations, client-side data handling policies are often called preferences.

1.4.2 Access control policies

An access control policy (ACP) protects access to an object by specifying which subjects should be granted which type of access to it. The object being protected can be a piece of data like a file, a database record, or a webpage, but it can also be a more abstract functionality like a service or a remote procedure call. The subject can be specified by means of a unique identifier (e.g. user name), by roles (e.g. administrator), by groups that he belongs to (e.g. helpdesk), or by other attributes (e.g. age, reputation, ...). A subject can be any type of entity that is capable of making a request; it could be a natural person but could also be a running process or a device, or a combination of those. The possible types of access (e.g. read, write) depend on the resource that is being protected. Finally, the decision to allow or deny access can be based on the subject's properties, the content of the resource, details of the access request (e.g. parameter values passed in a remote procedure call), and secondary information like the current time, processor load, etc.

1.4.3 Trust policies

A trust policy is a set of rules by which a subject specifies the conditions it accepts as evidence that other subjects are trusted to perform which actions on what data objects in the system, in terms of attributes, roles, or unique identifiers. A trust policy typically prescribes that such subjects must reach a minimum level of trust to be allowed to perform specific actions. The level of trust is usually measured in terms of possession of certificates. Organizations issuing these certificates are called certification authorities (CAs). The rules in a trust policy of a subject require the possession of certificates issued by the CAs trusted by the subject itself. In order to gain the subject's trust, a CA must provenly authenticate applicants before issuing certificates. The verification of the validity of a certificate may depend on yet another certificate. Certificate chains can thus be created, in which a certificate is verified by its predecessor in the sequence. The chain ends at an anchor certificate: a root certificate provided by a trusted CA, a cross-certified certificate verified by more than one certificate chain, or a locally defined source of trust.
Chapter 2

Scenarios

2.1 UC1. Wiki scenario

2.1.1 UC1.1. Register to the Wiki

Issue a credential upon a nickname. The credential issuing is protected by a policy which grants access if the nickname is not already in use and if some administrator has approved the issuing (e.g. the existence of another credential).

One possible chain of activity:

1. Hannes clicks on the "Register new user" button
2. PRIME-AskSendData pops up asking for a nickname
3. After sending "hann" (stored as PII in the server), the server calls its policy evaluation for issuing credentials.
4. The policy evaluation finds a policy stating that a nickname has to be present, which is
   - unique among all nicknames in the PRIME PII database
   - unique among all nicknames in the external wiki database (using a custom condition function?) and
   - certified by an Identity Provider.
5. Since the nickname is not certified, deny is returned.
6. Hannes gets a page stating that the nickname has to be approved.
7. He contacts an Identity Provider to get his nickname signed.
8. After the administrator at the Identity Provider checks Hannes' data, he signs the nickname.
9. Hannes tries to fetch the login again and now the credential is presented.

2.1.2 UC1.2. Edit policy for number of sites

Create/edit a policy for existing or not-yet existing wiki pages. Access can be set for:

- all sites,
• all sites that are "owned" (=created) by some user (this is a list of sites that can be queried at runtime, but is unknown at deployment),
• one specific or a static list of specific sites (which could be implemented by inserting separate policies for all those sites).

One possible chain of activity:

1. The admin sets up all wiki sites to be readable by default.
2. Some actions as viewing all sites starting with "User:" are only readable by the user with the appropriate nickname (the word following "User:"). Note: This should not be done by using the current policy-tag "subject", but by the presence of a correct value for the category "nickname".
3. All sites created by a user are - by default - only editable by this user.
4. All these settings can be overridden by more specialised settings.

2.1.3 UC1.3. Create a new site with policy

Restrict the access to a wiki page to users belonging to a group. (Same problem as above: the list cannot be known at deployment and can change during runtime.) It may be possible to implement credentials (list members get credentials and the presence of a credential is necessary).

One possible chain of activity:

1. Mariangela creates a new site in the wiki. Directly above the "Submit" button, there is a selection combo field with the caption "Who else can access this site?" and the following values
   ◦ nobody
   ◦ everyone
   ◦ users of group...
   ◦ the following users...
2. When choosing either of the last two options, she can select a group or a specific list of users.
3. After submitting, the server creates a new policy in the PRIME server containing the selection of her for the created site.
4. She can also change this setting later by clicking "Edit Policies".
5. The policy gets evaluated when someone tries to access this specific site. All other policies specified by other sources are evaluated as well.
6. Mariangela chooses to restrict access to group 'Friends on MyPrime'.

We are still unclear about the actual implementation of this use case. Could either be implemented by having the policy engine query the actual member id's of the group from an external source (over RequestContext or specify together with the policy evaluation request somehow..) or whether a credential stating the membership to the "Friends on MyPrime" group is used in the policy evaluation.

2.1.4 UC1.4. Access Control based on reputation

Restrict the access to a wiki page to pseudonymous users with some reputation value, whose value is higher than a specific number. The number may be written in the policy or come from external source (e.g. the reputation of the creator).
1. Josha wants to edit a wiki about recommendations of music bands. The site requires membership of an anonymous reputation system.
2. Josha can only edit sites, where he can prove a higher reputation than some values set.
3. Josha does not want to reveal his exact reputation value (to remain anonymous). Thus the created claim request by the policy engine asks for a proof of "higher than value X".

2.2 UC2. Blog scenario

The main policy here is a trust policy specifying who I trust to provide or certify a certain type of content. For this to work, we need at least a taxonomy of content types and fields of expertise. For example, if I trust someone to be knowledgeable about football, I don't necessarily trust his comments on basketball. Someone trusted in the whole field of sports however could comment on both.

The blog should also support (dis)approval comments or scores on blog entries, so that users can base their opinion about an entry on the opinions of other users that they trust. A user's policy can then for example state that only entries are trusted that received 5-star scores from at least 3 trusted users.

Additionally, blog entries should be accompanied with a DHP describing the allowed usage of the information (read, quote, modify,...).

2.2.1 UC2.1. Creating a blog

Actors: Author and Server

1. The Author either logs in under a previously created pseudonym, thereby making the blog to be created linkable to the other blogs under this pseudonym, or creates a new pseudonym.
2. The Author optionally adds pseudonyms of co-authors who should also be given write access to the blog.
3. The Author reveals to the Server those of his attributes that he deems relevant for the credibility of the blog and is willing to reveal, e.g. his profession, diploma, group membership,... The Server may also require him to have some attributes in order to be allowed to start a blog on a certain topic. This information might be evaluated by the server administrator or also revealed to other users.

2.2.2 UC2.2. Posting an entry

Actors: Author and Server

1. The Author logs in to the Server (possibly under a pseudonym) and requests write access to the blog he wants to post to. The Server authenticates the Author and checks he is registered as one of the authorized authors of this blog.
2. The Author writes his entry and authenticates it, e.g. by signing the content. He also attaches a data handling policy specifying for what purposes it can be used, which parts can be quoted, etc.
3. The Server makes the new entry publicly available on the website, together with the origin authentication and DHP data.
2.2.3 UC2.3. Posting a blog comment

Actors: Commenter and Server

1. The Commenter either logs in to the Server under an existing pseudonym that he previously created (thereby allowing this comment to be linked to his previous comments under this pseudonym), or creates a fresh pseudonym for this comment.
2. If this is a new pseudonym, the Commenter reveals (and proves) those of his attributes that he thinks are relevant for the credibility of this comment.
3. The Commenter enters his comments, possibly including an appreciation score for the article. This score information might be revealed to the public or restricted to given parties (e.g. the Author).
4. The Server makes the comment publicly available on the blog, together with (proofs of) the User's attributes.

2.2.4 UC2.4. Reading an entry

Actors: Reader and Server

1. The Reader specifies in his policy under what conditions he trusts the blog entries and comments, and hence under what conditions they will be shown on his screen. These conditions can involve statements about the content of the entry, the Author's attributes, the contents and/or scores from the commenters, and the Commenters' attributes.
2. The Reader requests to read a blog entry. The Server sends the entire entry with all comments to the Reader, but only those that satisfy the Reader's policy will show up on his screen.
3. Optionally, there may be an "overrule" option by which the Reader can also read untrusted entries and comments. The lower level of trust should be made clear by the interface though, e.g. by displaying the text in a different color.

2.3 UC3. Enterprise workflow scenarios

2.3.1 UC3a. Computer part workflow

HansDampf had over-clocked his computer. At some point in time, suspicious smoke came out of the box. Since then the computer did strange things. It required HansDampf to click an additional OK-box for every new application he wanted to start. It also turned itself off after 3 hours of uninterrupted web browsing. It would also let HansDampf download only 3 emails per hour and let him make only one blog entry per day. HansDampf concluded that overclocking had turned his computer insane and brought it to customer service.

At the customer service of ninja-computers, they recorded the name, address, phone number, region and birth date of HansDampf as well as the password for the encrypted file system and credit card payment information. The computer was disassembled and the parts were sent to certain departments. The customer service took care to only send the very needed information to the respective repair stations. Thus the password for the encrypted file system was only given to the hardware repair. The phone number would only be given to those mechanics who would need further information from HansDampf.
It turned out that some mechanics thought the CPU got severely damaged and would have to be replaced. The power unit needed some external repair and the hard disk had to be sanitized to rectify remains of viruses stored. A mechanic found that the strange melting produced by over-clocking the CPU put it in an unstable state. They tried to fix the CPU by introducing different statements about the correctness of CPU calculations and moved the spurious remains on the hard disk into hidden areas.

Each mechanic should be able to specify which type of contents he trusts. Trust can be based on the type and content of an item, and on the attributes of the mechanic who entered the entry. There is also some "recursion" in trust; for example, if one trust mechanic Y but does not trust the analysis and tests made by repair unit X, and mechanic Y based his diagnosis on analysis and tests made by repair unit X, then one may not trust this diagnosis by mechanic Y.

In addition to trust policies, there is also a need for access control policies as not all suppliers will need to know the password for the crypted filesystem. Additionally, as HansDampf has little confidence, he wanted to specify who is allowed to see what. For example, the disk repair should get the password but not the payment information.

It should also be possible to specify what happens to the data, e.g. whether it can be used for research purposes, whether it can be merged with other data, etc.

### 2.3.2 UC3b. Company HR workflow

A US-based company has numerous subsidiaries worldwide. Alice and Bob, two engineers respectively based in German and Swiss subsidiaries apply to the same internal job offer at US headquarter.

Each subsidiary maintains an employee database containing employees' names, addresses, bank account numbers, skill sets, salaries, and previous evaluation results. Since databases are respectively hosted in Germany and Switzerland, they are subject to stringent and different privacy rules. The full databases can never be transmitted to the US.

The final decisions who will be hired is made by the US board based on a blinded database containing only skill sets, salary ranges, and evaluation results. The evaluation results are signed by the manager who was in charge of the employee when the evaluation was made.

Once the decision is made, the legal department of the concerned subsidiary receives details regarding the position change in order to inform the current manager, start the relocation process, and so on.

### 2.4 UC4. Social Network Sites

This section lists a number of use cases focused on the use of Social Network Sites (SNSs). We have limited the description of the use cases to the actual interaction of the user with the system, which will normally take place using an Internet connection. The use case describes the interaction only, and does not go into details concerning the technical implementation of a certain use case. First, we do not want to push the technicians responsible for the actual implementation into a certain preconceived direction. Second, we do not have enough technical knowledge to make suggestions that would actually help technicians with the implementation of the use cases.
One of the key privacy issues in SNSs is the differentiated access to content posted on the SNS. So far, we have identified two general ways in which this access can be managed:

- On the level of connections: each (new) connection is designated certain access rights. For instance, friends could have different access rights than colleagues.
- On the level of content: each new piece of content is supplied with an overview of the (classes of) connections that may access this content.

We have not resolved which one of the two is preferable from the point of view of effectiveness, efficiency, and user friendliness.

2.4.1 UC4.1. Register to an SNS

1. When registering, a user supplies a selected username and password.
2. The SNS checks whether the username is still available. If not, the user must select another username.
3. The user gets first time access to an empty profile page.

2.4.2 UC4.2. Establish connections

1. A user searches the SNS for potential connections. The means through which the user finds these connections is less relevant: based on exact e-mail addresses, joint groups, via existing connections, particular search terms, etc. The result is the presentation of one or more potential candidates for connecting.
2. The user selects the name of the potential connection, selects the types of information the potential new contact will have access to (see key issue above) and clicks: “Connect!”
3. The invited person receives an invitation to connect with a link to the profile of the inviting person, which can be opened before acknowledging the invitation. The profile preview (that can be checked before accepting the invitation) reflects the exact view that the invitee will have based on the settings the inviter selected when drafting the invitation.
4. The invited person can click “Accept!”, “Decline!”, or first visit the preview.
5. In the case the invitee accepts, he/she has to select the types of information the new contact will have access to (see above).
6. The inviter gets updated on the decision of the invitee, and the connection is registered in the SNS system, viewable from the profile pages from both the inviter and the invitee.

2.4.3 UC4.3. Post content to own profile

1. A user selects a piece of content that he wants to post to his own profile (e.g. a photograph or a blog entry).
2. The user needs to login, to ascertain that he is the owner of the profile.
3. The user is presented with a number of connection(s) (classes), that potentially have access to the photograph.
4. The user selects which connection(s) (classes) will have access.
5. The content gets posted on the profile page of the user.
6. Depending on the preferences of the profile owner, a message containing information of the upload is sent to selected connections.
7. Depending on their preferences, the selected connections receive a message that new content has been added.

Concerning points 3. and 4. Of this use case, we refer to the key issue about differentiated access described in the introduction of this scenario.

2.4.4 UC4.4. Post content to another profile

1. A user selects a piece of content that he wants to post to another profile (e.g. a photograph or a comment).
2. The user needs to login, to ascertain that he is a legitimate SNS member.
3. The settings of the destination profile owner are checked whether he allows the uploading of content by this particular uploader to his profile.
4. If yes, the content gets posted on the profile page.
5. Depending on the preferences of the profile owner, a message containing information of the upload is sent to selected connections.
6. Depending on their preferences, the selected connections receive this message that new content has been added.

In principle, the profile owner to whose profile information is being uploaded, should be in control who will be able to access the newly posted information. This would be in line with points 3 and 4 of the last use case titled Post content to own profile. In order to keep the use case manageable, we decided not to include this option here.

2.4.5 UC4.5. Control access to and use of SNS content (profile information or other content)

The goal of this use case is to differentiate access to and use of SNS content between different parties:

• (Different categories of) other SNS members
• The SNS provider
• Third parties (including APIs such as automated mash-ups).

1. Content is posted on the profile (see use case above).
2. Access to the content is controlled (see remark in introduction).
3. Another SNS member, either a connection or not, the SNS provider or a third party wants to use the content (copy, modify, add comments). Obviously, the involved party needs to have obtained access to the content based on step 2 first.
4. The profile (and content) owner wants to define the allowed use of the content and differentiate this between parties (different connections, other SNS members, SNS provider, third parties).
5. Depending on the rights given to the party involved he can or cannot perform the desired action.

2.4.6 UC4.6. Report on access to and use of SNS content (profile information or other content)

The goal of this use case is to generate an overview of accessed profile information (logging), once again differentiated per user, such as other SNS members, the SNS provider and third parties (including APIs).
1. After profile information (or other content) has been accessed or used, according to the steps in the use case above, the profile owner wants to have an overview of the accessed information and what has been done with it. This enables the profile owner to check what happens with his data/content.
2. Either each time information is accessed or used a report is sent to the profile owner, or this is done on a daily/weekly/monthly basis. A distinction between access and use is desirable.

2.4.7 UC4.7. Transfer profile to another SNS (data portability)

A user decides to use more than one SNS or to switch to another SNS. In order to save time he wants to transfer (parts of his) profile information to the new SNS.

1. The user first creates an account on the new SNS.
2. The identity (as a profile owner) of the user is checked and the (old) SNS allows the transfer of the profile information.
3. A connection between the two SNS services is established.
4. The profile information is transferred, preferably immediately into the new profile.

2.4.8 UC4.8. Integration of mobile SNS applications

1. A user has an application on his SNS profile that uses (location) data from a mobile device (for instance a map that shows his location).
2. The location data from the mobile device are sent to the SNS. This may be a continuous process, or only be occurring at several (user-triggered) intervals.
3. The data is verified and connected to the correct SNS profile.
4. The data are transferred to the application.
5. The SNS processes and displays the data.

2.4.9 UC4.9. Close an SNS profile

1. A user decides to stop using the SNS.
2. He selects the option to close his profile.
3. All profile information is erased. (Important: all information that originated from his profile page, but has been distributed throughout the SNS and beyond, should be erased as well).

2.5 UC5. Anonymous credentials and identity management

After a recommendation from her sister Alicia, Alice considers to purchase a box of white wine at ‘CyberWinery.com’. The figure below shows the main entities involved in the scenario and the data flows in a classical, non-privacy-preserving setting. For example, prior to the purchase Alice is likely to first create an account at CyberWinery, thereby disclosing personal data. The account will store purchase data, personal preferences, and possibly even credit card data. CyberWinery has outsourced warehousing and delivery to ‘LogisticsProvider’, which requires data from CyberWinery (like a delivery address). For the payment by credit card, CyberWinery checks Alice's credit card details for authorization at ‘CreditProcessor’, which stores the transaction details for their own business and accounting purposes. Other services may also be present, such as CyberBooks which is recommended by
Alicia for the purchase of the Good Wine Guide. This purchase again requires Alice to register with her personal data and CreditProcessor and LogisticsProvider are also possibly involved in this transaction.

Alice, a vigilant and sensitive ‘Netizen’, is aware of the risks involved in online transactions and knows that the loss of personal information can cause severe financial and reputational damages which are difficult to repair, and she has heard that the use of personal data by others may lead to discrimination, exclusion, and social sorting. Because of this, she adheres to the principle to share a minimum amount of personal data on the Internet. Fortunately for Alice, CyberWinery uses anonymous credential technology, which assures her that she can make use of a secure privacy-enhancing identity infrastructure that complies with current data protection legislation. CyberWinery also has respected trust marks and provides clear information about the buying process.

2.5.1 UC5.1 Login use case

CyberWinery lets Alice control the amount of identity information that she gives away by allowing purchases via pseudonymous accounts that are created using anonymous credentials. CyberWinery demands Alice to prove that she is either over 18 and an EU national, or over 21, but this is possible even within Alice's choice to be pseudonymous. Alice only needs to attribute a number of anonymous credentials (issued by Trusted Third Parties, such as her bank or the State) to her chosen pseudonym.
Alice starts the interaction by issuing a request (1) to create a new account. The CyberWinery expresses its requirements to create a new account as an access control policy (ACP) with the account creation service as the resource. CyberWinery’s ACP specifies (2a) that it needs a pseudonym and a proof that either the user is over 18 and of EU nationality, or over 21. CyberWinery’s trust policy (TP) specifies (2b) that it trusts any official government to certify the attributes age and nationality of users. Alice chooses (3) to use pseudonym ‘alice83’ and proves to the CyberWinery that she has a valid credential from the US government stating that she is over 21. The CyberWinery verifies the proof and creates (4) the requested account.

Alternatively, for some reason Alice may in (3) prefer to prove the statement (yearofbirth=1971). For example, Alice may not be using standard certificates rather than anonymous credentials and happens to have a certificate of that statement lying around. The CyberWinery has to be intelligent enough to realize that (yearofbirth=1971) implies ((age>18 AND nationality in EU) OR (age>21)), and should grant access.

### 2.5.2 UC5.2 Buying use case

Alice brows the catalogue, chooses a box of white whine, puts it in her virtual shopping cart and proceeds to the checkout. She now has to give her credit card information to CyberWinery to pay for the wine, and her address to complete the shipping.

By clicking on the checkout button Alice initiates an access request (1) to CyberWinery’s buying service. CyberWinery’s access control policy (2a) states that in order to gain access to this service, Alice has to reveal her credit card number and expiration date, and provide the Logistics Provider (LP) with her address. The trust policy (2b) states that any government is trusted to certify Alice’s address, and that either UBS or CreditSuisse (both are Swiss banks)
are trusted to certify Alice’s credit card info. Alice’s privacy policy however (3a) states that she’s only willing to reveal her credit card info to “good businesses”, which in her trust policy (3b) is defined as businesses with either a gold label from the BetterBusinessBureau (BBB) or a reputation more than 5 stars from PrivacyOrg. Finally, her data handling policy (3c) states that her credit card information can only be used for the purchase and has to be deleted immediately afterwards. Similarly, her address can only be used for shipping and has to be deleted as soon as the item has been shipped. CyberWinery accepts the DHP (4) and proves that it has a BBB gold label. Alice sends over (5) her address, verifiably encrypted under the Logistic Provider’s public key, and proves the correctness of her credit card information.

2.5.3 UC5.3 Gift certificate use case

Bob’s birthday is coming up. Alice would like to buy him a box of 6 bottles of wine, but since she isn’t sure about his taste, she prefers to give him a gift card. CyberWinery has gift cards that allow the recipient to buy 6 bottles of wine, probably within a certain price range. The bottles can be different ones, though, and the 6 bottles do not have to be bought in a single order. If the gift card is one (or more) anonymous credential issued by CyberWinery, then CyberWinery cannot link Alice’s purchase of the gift card to Bob’s spending of it, and also cannot find out whether Bob is buying the different bottles with the same gift card or with different ones. Still, it can detect overspending of a gift card, and refuse to proceed with the transaction if this occurs.

In steps 1-3, Alice buys a gift card for 6 bottles. Payment is done similarly to the buying use case above. In step 4, Alice sends the gift card (in the form of one or more anonymous credential issued by CyberWinery) to Bob. When Bob checks out one bottle of wine (5), CyberWinery requires him to give either his own credit card information or show a valid gift card that has not expired and has not been overspent (6a). In the latter case, Bob also has to submit his real name to a trusted third party TTP, so that he can be held accountable if overspending is detected afterwards. CyberWinery’s trust policy (TP) thereby specifies that (6b) either UBS or ZKB (both are Swiss banks) are trusted to certify Bob’s credit card info, that only gift cards certified by CyberWinery itself are trusted, and that it trusts any official government to certify Bob’s name. Bob chooses to pay with a gift card and proofs this fact to CyberWinery (7). Additionally he sends over his name, verifiably encrypted under the trusted
third party’s public key. CyberWinery checks the proof and proceeds with the transaction if everything is okay.

Overspending can be checked online or offline. If it is done online, then Bob doesn’t really need to send his name to the TTP, because CyberWinery can simply abort the transaction as soon as overspending is detected. If it is done offline however, and it is impossible to cancel the transaction (for example because the wine has already been shipped), then CyberWinery should contact the TTP to obtain Bob’s real name to make him pay for the wine.

2.5.4 System overview

The figures in the scenarios given above illustrate the sequence of messages that are exchanged between the parties performing online transactions. The messages exchanged are access requests, different types of policies, confirmations, proofs, etc. For describing the scenarios, these kinds of illustrations of the message flow are sufficient to give an idea of what the policy language(s) should be capable of. However, with the intention that such systems shall eventually be working and deployed, a closer look at the individual system components is necessary. Although a detailed description of the design of such systems is too early at this stage, we want to give a high level description of the system. This might produce some useful ideas for the development of the policy language(s) and possibly new requirements arise by thinking about the whole system in which the policy language(s) shall finally be embedded.

Client components

Users have a credential repository that contains all the credentials they hold. For example, Alice has a number of anonymous credentials in her credential repository. One is her digital ID card issued by the U.S. government, mentioning her name, address, birthdate, etc. Another is a credit card issued by UBS (a Swiss bank), mentioning her name, the credit card number, and the expiration date.

Associated with each attribute contained in the credentials, a user has a release policy (release policy is the name of an access control policy for PII) specifying under what conditions she is willing to reveal this attribute. A user’s trust policy states the requirements on issuers of certified attributes, i.e. who is, from Alice’s point of view, allowed to certify which attributes. For example, Alice’s release policy, together with her trust policy states that she reveals her credit card data only to sites approved with a gold label from the BetterBusinessBureau or a 5-star reputation from PrivacyOrg.

A specific data handling policy might also be associated with the attributes of the possessed credentials. These policies state what the obligations on the data receiver are, once she received the data. For example, she has to use the data only for specific purposes and must delete the data after a defined period of time.

Server components

A credential repository contains all the credentials that a service provider holds. CyberWinery has for example an anonymous credential which was issued by the BetterBusinessBureau to state that CyberWinery has gold status.
A service provider offers a number of services to which access is restricted. Associated with each resource there is an access control policy specifying what the requirements are in order to get access to the resource. For example, CyberWinery offers the services create_account and buy_service. Associated with the create_account resource, there is a policy specifying that the user needs to provide an uncertified pseudonym, and has to prove that she is either over 18 and national of an EU country, or over 21.

Like users, service providers maintain trust policies that state who, from their point of view, allowed to certify data. For example, for the age and the nationality attribute, CyberWinery states that any national government is trusted to certify this kind of data. Service providers might as well have release policies associated with the attributes contained in their credentials.

**Request processing**

Each party that is involved in an online transaction has to perform a number of steps in order to process an incoming request. The following describes some of these steps to get an impression of how users and service providers handle transactions.

When a user requests access to some resource, the service provider looks up the policy associated with this service. This lookup may include deriving the specific policy from a more general policy for all of its resources. The service provider also determines the trust policies that are associated with the attributes contained in the access control policy that is going to be sent to the user. Depending on the policy, additional steps might have to be performed before the policy can be sent to the other party. Such steps could include resolving special ontologies that were used when creating the policy, etc. As soon as the access control policy for the requested resource is fulfilled, the user gets access to the resource.

A service provider maintains session information to be able to know whether a user already proved to satisfy the requirements contained in the access control policy. In case the user already proved to satisfy parts of the requirements, only the parts she still needs to prove are sent. A user who has already proved to satisfy all requirements gets direct access to the requested resource.

In case a party is requested to reveal or prove some predicate over an attribute, the party checks in the corresponding release policy (i.e., access control policy for PII) if she is willing to do so and what the associated requirements would be. The other party could then for example be required to show certain credentials. Thereby, also the relevant trust policies have to be determined in order to let the other party know which issuers are allowed to certify the required attributes. In addition to the release policy, also the data handling policy applicable to the corresponding attribute has to be determined and sent. Again, the trust policies relevant for the content of the data handling policy need to be determined and finally sent to the other party. If a party is willing to reveal or prove some predicate over an attribute without further requirements, or the requirements have already been fulfilled by the other party, then the attributes of the relevant proofs can be sent.
2.6 UC6. User Interface requirements

2.6.1 UC6.1. Trust and assurance HCI

Trust policies should include statements about trust factors that can influence the trust of end users, such as statement of privacy seals on a service side, reputation metrics, statements about the system configuration. Input for trust negotiation should not only be claims by the service side/communication partner, but also statements by third parties, such as statements from consumer organisations on whether a service is "blacklisted".

2.6.2 UC6.2. User interfaces for policy display and administration

For simplifying the management of data release policies (preferences) for the user, our UI proposals have built on providing a set of predefined data release policies, which can be customised "on the fly". The user's data release policies should be able to state:

- what data categories or what concrete data values may be released for what purposes and to whom or to what types of recipients (e.g., only "trusted" partners)
- conditions under which data may be forwarded to other parties
- type of pseudonyms that should be used under this release policy (e.g., transaction pseudonyms, role-relationship pseudonyms, etc.)
- the expected obligations and reputation of the potential recipient.

Work package 4.3 needs a policy language to describe all the attributes needed by its interfaces. Research is currently heading towards a fine granularity of purposes, so if a transaction with a service is conducted, and this transaction includes the ordering of a product, and at the same time the subscription of an advertisement mailing by the service, this could be expressed by two separate policy statements for the different purposes "order" and "advertisement".

Such a policy would need to be able to express:

- the types of data
- the purpose for which this data is collected and processed
- the processor of the data, following a specific defined structure, including address data
- a field for a full version of the legal policy, possibly with further substructures
- a field for the condensed version of the legal policy, possibly with further substructures
- a field for a short version of the legal policy, possibly with further substructures such as: the identity of the controller and the purposes of processing (w/exceptions), any additional information which in view of the particular circumstances of the case must be provided beforehand to ensure fair processing, and a clear indication must be given as to how the individual can access additional information
- a reference to an icon illustrating the policy
- the recipients or categories of recipients of the data and how the data is handled after it has been transferred to them
- a point of contact must be given for questions and information on redress mechanisms either within the company itself or details of the nearest data protection agency
2.7 UC7. Services Composition Scenarios

This section describes some use cases related to service composition. For additional information on those use cases, look at PrimeLife's work package 6.3, which focuses on service composition. More details can be found in the D6.2.1 [D6.2.1] and upcoming H6.3.1 documents.

2.7.1 UC7.1. ePortfolio scenario

(within the context of a service composition scenario)

- Overview of a service composition

An individual wants to apply for a job using a composition of services that includes the following (see figure above): an ePortfolio manager that contains details concerning the person’s education, employment history, and personal details; a HR service (provided by the employee’s employer) that can send a verification to the ePortfolio manager that the individual was/is an employee and can also confirm that a person has taken training; the University Service can provide proof of qualifications/certifications; the Job Service provides offers based on the individual’s preferences. We assume in this case that the services within the composition are trusted.

- Privacy-enabled Policies

The different services within the composition enable policies to be attached to data; they outline the way the data should be handled within a service composition.

- Use Case
  - an individual would like to use the web service composition to search for new jobs.
  - the ePortfolio combines existing data and aggregated data from different sources (HR and university).
  - the privacy-enabled ePortfolio is provided to a service that offers jobs (i.e. Job Service).
  - the Policy Reconciliation service is the intermediary between the ePortfolio and the Job Service.
Job Service agrees to enforce part of the preferences of the ePortfolio (e.g. do not send data to a third party if the third party plans to use data for statistics and delete data after 10 days).

### 2.7.2 UC7.2. Data Flow in Travel Booking

**Scenario Description**

In composite services, it is frequent that personal data collected by the composition be shared with different services. For instance, in the figure below, “Travel Booking” service collects the user e-mail address (Email) to contact this user to confirm reservation. Third party services may be subsequently called in order to book the travel, e.g. “Hotel Booking” and “Car Booking”. Both may also require the user’s e-mail address to contact him in case of emergency.

![Data Flow Diagram](image_url)

The data handling policy of each service getting specific personal data has to match the user’s preferences. For instance, if the user only provides his e-mail address for booking or statistical purposes and with the obligation to delete it within six months, involved services can only use this e-mail address for specified purposes (i.e. cannot use it for advertisement) and must delete this e-mail address within six months.

The user also needs a way to specify whether secondary use (i.e. Travel Booking sharing user’s e-mail address with third parties) is authorized.

**Requirements**

- The policies of all services are matched with the user’s preferences.
  - Option 1: the user gets a global policy combining third party policies. This requires language and tools to merge and generalize policies.
  - Option 2: the user “delegates” to the front-end service the right to select third party services and hand user’s personal data to those ones. The policy and preference language must support conditions on third parties. For instance, “my personal data can only be used for statistics and can only be shared with third parties in Europe”.

25
2.7.3 UC7.3 Data Flow in Human Resources

**Description**

The previous use case (data flow in travel booking) describes a common data handling scenario with secondary use. A user (i.e. a human being) is in front of a service that consumes his personal data. The user has data handling preferences and the service has data handling policy. This use case shows that this client-server approach is too restrictive and insufficient.

Let’s illustrate this with a concrete example. An employee provides PII (e.g. nationality, number of children, etc.) to his employer. The employee and employer agree on how this PII will be handled. For instance, a sticky data handling policy could define that PII can be provided to specific third parties for a specific purpose.

In well defined cases, the employer can provide to a subcontractor (e.g. financial auditor or tax advisor) the PII provided by the employee. The employer has to make sure that providing the employee PII to this subcontractor does not violate the sticky data handling policy.

In such scenarios, a human being (e.g. subcontractor) may have to expose and enforce privacy policies. The border between data handling, access control, and rights management may disappear.

**Requirements**

- Any party can act as data source and data consumer. In other words, privacy preferences are not restricted to human beings (users) and privacy policies are not specific to services.
- Access control and data handling are not independent. For instance, the sticky data handling policy associated with PII can directly impact the access control of the employer’s HR service, e.g. any subcontractor can access the employee’s PII if it commits to fulfill some obligations.

2.7.4 UC7.4. Privacy tradeoff in search/discovery

**Scenario description**

When searching for services (e.g. an available room in closest hotels), there may be a tradeoff between quality of service (e.g. functionality) and privacy. For instance, the user of a "SearchHotel" (composite) service will provide his location and know that his query for an available room will be sent to 23 hotels in his neighbourhood. If the user chooses to disclose more personal data (i.e. less privacy) such as smoker/non-smoker and nationality, which are expected by some hotels to check for availability, his query will reach 47 hotels. The user may get a question like: “With your default DH-preferences, 23 out of 78 services will be queried. A majority of users of this service has less restrictive polices matching 47 out of 78 services. Do you want to proceed or change your preferences?”. We could use a more appealing scenario and offer a visual representation of the tradeoff between privacy and functionality.
Requirements

• Specify “optional” part in data handling policies. For instance, the user must provide his location and he can optionally provide other personal data to improve the quality of service.

2.8 UC12. Logging/Auditing Use Case

When personal data is processed the law requires many actions, such as modification - in some cases even access - to be logged for auditing purposes. A typical scenario that arises in this context is the following:

A governmental agency stores personal data for their purposes (could be tax, health, or social security information). It grants access to this information to its employees.

• During the process the data is first provided in writing by a citizen.
• Subsequently, the data is entered into the system by an employee.
• Later the citizen calls in and some information is changed by a different employee.
• Another employee who has had difficulties with the citizen privately, accesses the data and changes it incorrectly.
• Finally a decision on some application by the citizen is taken based on the incorrect data.
• The citizen now wants access to the logging to prove that someone has changed the information without a valid reason.

2.9 UC13. Privacy of location-based information

The diffusion and reliability that mobile technologies have achieved provide the means to exploit location information to improve current location-based services and applications in a novel way. In this case, location is one of the user's attributes that might be passed to the service provider and can therefore represent a piece of PII. In this context, privacy concerns are increasing, calling for more sophisticated solutions to provide users with different and manageable levels of privacy.

Two privacy aspects that might need to be considered.

• Location information, just as any other PII might be subject to restrictions (cannot be known or further communicated). From a policy language point of view, a language for location-based applications can give the users the ability to specify protection degrees on her location information. For instance, the user - whose identity might be known - does not want her location to be known at a fine grained-level (e.g., area of not less that 20km). This might impact the policy language if we want the user to be able to express desiderata for protection degrees on her location information.
• Location information can be exploited for re-identifying the users in scenarios where the user identity is otherwise not known. This aspect might be more related to data protection (than to support required in the policy language). It resembles exposure risks due to user profiling (and user re-identification based on this), as it aims at avoiding re-identification of users joining the locations from where requests originate with information on the locations of each user (available in a database or through observations). However the peculiarity of location information requires considering specific attacks aimed at determining the identity of the users by
exploiting information on their locations and developing effective solutions counteracting them.
The following set of use cases describes larger perspectives on the use of policy languages. They have been brought to our attention by members of the W3C Policy Language Interest Group.

### 3.1 UC8. Privacy Policy Management

*Illustration by D. Sommer*

Original Author: Marco Casassa-Mont, HP Labs, UK (with kind permission from the author, Thank you, Marco)
Description: A recurrent use case that I came across in various contexts (EU PRIME Project, interactions with customers, etc.) is how to use policies to deal with privacy enforcement and compliance checking in organisational contexts. This is pretty much consistent with some of the points already highlighted in Michael Wilson's use case.

Organisations and enterprises collect a lot of personal data and sensitive information in order to enable their businesses. In doing this, they need to comply with laws, legislation (HIPAA, COPPA, EU Data Protection, etc.), standards of business conduct, guidelines, etc.

Three key aspects are of interest: (a) policy representation (b) enforcement of (privacy) policies (c) policy compliance checking

Basic privacy constraints require handling users' consent, allowing access to the data only for agreed purposes and managing the lifecycle (e.g. data transformation, minimisation, deletion, etc.) of personal data driven by privacy principles. However, also other aspects such has security and business constraints need to be kept into account.

Personal data can be stored in a variety of data repositories (databases, LDAP directories, file systems, etc.) and can be accessed by people, applications, services. This data can be processed and disclosed to third parties.

A "blend" of personal preferences, business, security and privacy constraints need to be kept into account into "policies" dictating how to access, use process and disclose this information.

Some common requirements:

- need for more "integration" of business, security and privacy policies - need to leverage state-of-the-art Identity Management solutions (that might use, in some cases, proprietary/ad-hoc policy languages ...) - need to measure and demonstrate compliance to guidelines, laws and legislation

Policies (and policy management frameworks) can play a key role to deal with privacy enforcement and compliance checking tasks. However, one of the current limitations is that these aspects are currently addressed in a "compartmentalised" way, by using different policy languages and policy management systems (for security, privacy, etc.) that do not interoperate i.e. act in stand-alone ways. This creates issues in terms of alignment of policies, their consistency and overall impact.

How to make progress by recognising than on one hand there are multiple policy languages and policy management systems and, on the other hand, more coordination and integration is required?

3.2 UC9. Federated Policy Management

Original Author: Marco Casassa-Mont, HP Labs, UK, (with kind permission from the author, Thank you, Marco)
An enterprise/organisation uses a broad variety of IT tools and solutions. The enterprise IT infrastructure includes systems, tools and solutions deployed at different levels of abstractions: network, system, OS, information, application, service, business, etc.

Many of the involved systems, tools and solutions are configured with and driven by "local policies", defined by using specific (sometimes ad-hoc ...) languages. These "local policies" are often the effect of (human-based) "refinements" and "deployments" of high level business/security/etc. policies. Different policies, policy decision points and policy enforcement points are used for security, business, privacy and other aspects.

- How to make sense of all of them? - How to ensure that their overall impact on the IT infrastructure is consistent with the high level policies and guidelines - How to understand what the impact of changes of "local policies" (let's say at network level or at the application level) is on the high level policies? - How to understand what the impact of changes of high-level policies is on some of these "local policies"?

This is what I call a "federated policy management" use case i.e. a use case where there is the need to understand and keep into account the overall set of policies deployed in an IT infrastructure and have an "integrated, coordinated and consistent" management of these policies.

In this use case many different policy languages are used, operating on different IT entities (at different levels of abstraction) and enforced by different policy enforcement points. It is unlikely that all these existing (local) policy languages are going to be replaced by a unique "comprehensive" language ...

Some requirements are about having a consistent "meta-representation/abstraction" of the core principles/aspects/constraints expressed by various "local policies" along with ways of defining dependencies and relationships between them.

This would help to better understand the overall heterogeneous set of operational policies, link them back to high-level policies and reason on top of it.

### 3.3 UC10. Authentication Strength Policies

Original Author: Giles Hogben, European Network and Information Security Agency

from: http://www.w3.org/Policy/pling/wiki/UseCases [PLING]

Description: Governments and companies operating remote authentication systems often need to set machine readable policies (or even human readable policies used within organisations), specifying the minimum strength of authentication tokens required to access a service. Other features of the authentication context (as it is called in SAML) are also of interest such as the linkability features of the tokens used, and the registration and issuance procedures which surround how the token is bound to the person's identity.

Many proposals abound for sets of levels describing authentication strength, but there is no standardisation in this area as yet.
3.4 UC11. Corporate Security Policies

Original Author: Giles Hogben, European Network and Information Security Agency

from: http://www.w3.org/Policy/pling/wiki/UseCases [PLING]

Description: We see a need for a common format for sharing controls used in corporate security policies. Obviously no company would want to disclose their internal security policy, but they would want to be able to view and modify standard expressions of security policies. This would facilitate certification bodies to apply standard evaluation procedures.
Chapter 4

Requirements

Note: the requirements have unique numbers for future references from other documents. They are not necessarily consecutive nor significant.

4.1 General principles

The following requirements are considered highly desirable for all the scenarios, therefore are described as general principles.

4.1.1 General principles for Data Handling

RG11 Measurability
This property of a policy language is fulfilled when the construction of the language allows to check that the policy has been followed correctly. A mechanism to prove that a rule has been applied is useful but not sufficient to demonstrate this property, since some policies can be applied a long time after the moment when it has been stated.

RG12 Unified model
Data Handling policies being tightly related to Access Control policies, a unified model is a key success factor for a policy language.

RG13 Semantic compatibility
As much as possible, a policy language should be semantically compatible with the Platform for Privacy Preferences (P3P). Details of P3P semantics can be found in section 3 of The P3P 1.0 Specification [P3P 1.0 Specification]).

RG14 Stickable policies
The ‘stickability’ is the property of the policy language that allows to attach a policy to data no matter how, where and when the data is sent.

RG15 Revocability
It must be possible to revoke a Data Handling policy the same way it is possible to revoke a credential.
RG16 Transparency
A policy language must be able to express the data flow trace resulting of the transfers of the data between entities.

RG17 High-level Policies
The policies should be expressible not only on a low, i.e. more technical, level but also on a higher, i.e. more abstract, level. The benefits of this are for example that the policies can become shorter, easier to understand and also easier to formulate. Among other techniques, ontologies could be leveraged to bring the policies on a high(er) level. Instead of talking about credit-card-number, credit-card-name, etc., an ontology could describe such data under the class 'credit-card data', or even more general, 'payment data'. Then the policy can refer to concepts like credit-card or payment data if needed.

4.1.2 General principles for Access Control

RG21 Data minimization
The policy language should support -- and encourage -- minimization of the amount of personal information that is revealed in order to gain access to a resource. The architecture should definitely not assume that all information about the subject is readily available when the access decision is made. Rather, the list of attributes that need to be revealed, or the predicate that needs to be proved, should be explicitly specified by the server, or perhaps even be the result of a negotiation between the client and the server. The client should then have the option to reveal only those attributes that are strictly necessary. Whether this is possible of course not only depends on the policy language, but also on the underlying authentication technology. In case for example X.509 certificates are used, the user has to reveal all its attributes so that the digital signature can be checked. The data handling policy should be determined in a similar "minimal" way: the general principle should be that the DHP only allows actions that are needed to perform the service, rather than forcing the user to agree to a default, more lenient DHP. It should therefore be possible to associate different DHPs to different resources, and these should be tightly coupled to the relevant ACPs.

RG22 Anonymous and/or pseudonymous access control
A user shall have the possibility to access a resource in an anonymous or pseudonymous way. For an anonymous access, the server makes sure that the user fulfills the necessary requirements, e.g., age>18, while the required attributes allow the user to stay anonymous. This is of course only possible if (1) the required attributes (like age>18) are applicable to a big number of people and the user can therefore not be identified, and (2) the underlying technology supports proving of the attributes in an anonymous way (for example using the technology of 'anonymous credentials'). A pseudonymous access is similar to the anonymous one, with the difference that for every access a user does, he provides some kind of identifier - a pseudonym - which the server uses to recognize that the same (pseudonymous) user requests access. However, the server only knows the pseudonym and not the real identity of the user. This is important if a user wants to keep some profile on the server side, while the user still wants to be anonymous to the server. From a legal point of view, services must be offered pseudonymously whenever that can be considered reasonable for the service.

RG23 Link to data handling policy
Even though in this document we often focus on access control and data handling policies separately, they are in fact closely related. A server's access control policy should not only specify what PII it wants from the user, but also how it is planning to
treat the data. Here, the DHP is part of the ACP. On the other hand, a user's DHP may specify which third parties are allowed to see the PII, so that the ACP becomes part of the DHP.

4.2 Requirements for language model and expressivity

R410 Meta-policies and policy generation
In some cases it is necessary to restrict the possible policies that can be attached to data. This can be achieved by defining meta-policies, that specify policy templates. The same mechanism can be used to specify constraints on policies that are generated, e.g. by a service in response to a user request. Those constraints can be derived from trust or access control assertions. This mechanism can rely on a way to express policy generations rules in the policy language itself.

4.2.1 Requirements for Data Handling policies

R111 Business logic to describe data usage
The business logic of an enterprise determines what actually happens with the data after it is received. If this business logic is described in a standardized way, for example using WS-BPEL (Business Process Execution Language), then it should be possible to automatically derive the DHP from it, or perhaps the BPEL description itself could even be part of the DHP.

R112 New usage should trigger consent
The policy language must support a mechanism to acquire new consent from the user if the data controller wants to change the policy.

- R112a: the above mechanism must be suppressable by user preference

R113 Legal policies need differentiated layering (not much semantics for enforcement)
A policy language must include the possibility to express and address at least three layers of human-readable text to describe a policy to the user. This is recommended by the Op. 100 of the Article 29 group: Opinion on More Harmonised Information Provisions [Art.29 Opinion 100] and Annex [Art.29 Opinion 100 Annex]:

- a short version of the privacy policy, with an addressable substructure to be defined
- a condensed version of the privacy policy, with an addressable substructure to be defined
- a full (lawyers readable) version of the policy, with an addressable substructure to be defined
- a fourth layer to express the policy with iconography should be available, with a set of icons to be defined.

R114 Technical representation of legal policies
The policy language should be able to express legal policies in a form supporting their digital storage, transmission, and machine-based processing. The semantics of the representation should be carefully considered and be compatible with the capabilities of an efficient processing engine.
R115 Policy templates
The policy language should support the definition of templates, e.g. to represent optional services or portions of service. Constraints may be defined on the structure of conditions in order to permit their flexible application over policies.

R116 Policy matching (domains, variables, _fill the blank values_) Since different privacy policies should be compared with one another, they have to share the same tree structure. Therefore policy templates should be used for expressing policies.

R117 Support variables A policy language should support variables in order to handle and reason on constraints, e.g. obligation: store audit log during y years, 2 < y < 5. This feature would be important when it comes to combining policies as it provides a greater degree of freedom for policy resolution.

R118 Support nested policies The policy language must support nested policies. Thus, a policy could include a number of specific policies for further processing of the data.

R119 Express user preferences A policy language must allow to express users preferences about handling of their data. In particular, it must be possible to express preferences for use of given credentials for a given purpose. The user should also be able to express general trust relationships independently of a given scenario or purpose. The language should be extensible enough to express new user-defined preferences.

R120 Describe server policies There are two possible cases in this context,
- The server has no Data Handling policy. For instance, a basic data repository where the user stores his data and associated policy in the repository (whose duty is enforcing user policies).
- The server has its own Data Handling policy (one or multiple). The user's Data Handling policy should match one of the available server policies.
For simple user experience, complex Data Handling policies and preferences should be visualised in a simple way.

R121 Originator's policy A policy language must include a mechanism to identify what the data originator allows, no matter who transmits the data.

R122 Logging/Monitoring/Auditing Policies
- R1221 It must be possible to inform the user about the data collected during the usage of the service (date, location, actions, credentials used, etc.)
- R1222 It must be possible to express the scope of the retention (page, session, duration) and usage (extend user experience, debugging, legal requirements) of the collected data.
- R1223 It must be possible to express how the data is collected: how, when, where it is stored.

R131 Data model primitives The policy language must make consistent use of (at least) the following concepts:
- date and time
- location
R135 Security levels
The level of trust does not only depend on the claims made by the subject, but also on the underlying technology that is used to prove the validity of these claims. Probably the policy designer should not be bothered with technical details such as cryptographic algorithms and keylengths, but given that the language should be useful in both low- and high-security environments, some notion of "security levels" seems appropriate. What these security levels imply on the underlying technology and infrastructure could then be specified in a separate ontology.

R136 DHP ontology
The language should provide in a standardized list (i.e., an ontology) of data purposes and types of data. This list should be extendable, as we can impossibly foresee all items that should appear in this list. A requirement that can be derived from scenario UC2 (blogging use case) is that we also need a classification of "conversation topics", so that the policy one can relate the topic of an article to the qualifications of an author. Obviously this list should be extensible as well.

R139 Enforcing DHPs
Technological means to enforce data handling policies are limited. A trusted software infrastructure can assist in automatically adhering to a DHP (e.g. deleting data on time) and in logging access for audit purposes, but eventually these systems can always be circumvented by a malicious user (e.g. by forwarding a picture of the screen displaying the sensitive information). In the end one will have to either trust the receiver to adhere to the DHP that was agreed upon, or to trust an external auditing agency to correctly certify such receivers. It should be possible to express this type of trust in the policy.

R140 Auditability -- obligations toward where to log access
- A policy language must support mechanisms to express what types of processing (Access, Modification, Transferal, Deletion) need to be logged, and what specifically needs to be logged (change management, role, name or pseudonym of the person in charge of the processing)
- A language defining the logging also must be defined

R141 Breaking the glass
A special case of a policy: the law prescribes certain areas, where access and processing is compliant, although clearly not within the prior consent. In these cases, it might be reasonable to invoke special mechanisms for transparency (i.e. the glass is broken = the prior consent has been exceeded). Such a policy could state that certain entities are entitled to access the data, but then certain obligation regarding information of other parties, esp. the data subject or data protection supervisors might be triggered.

R142 Capture user intent
(As opposed to the service purpose of data handling. See also R143) This property is needed to differentiate the user intent when using a service (e.g. "buy a book about philosophy") from the purpose of the service itself (e.g. "sell products: book or CD or DVD or video game"). The user intent does not necessarily match the service purpose. The mechanism for capturing the user intent may be simple, however, processing it requires use of semantics and logic.

R143 Purpose of data processing
The purposes of data processing or data handling by a service usually stay the same across the usage of the service for all transactions. However, it is not always clear for the user whether a given piece of data is going to be reused for a purpose other than the most
obvious one. For example, an address can be used for shipping, but also for later marketing actions. The policy language must allow to express all the different purposes of data handling by the service, so that the user can be informed about the less obvious purpose of data processing.

**R144 Express obligations**

A policy language needs to express all the obligations of the processing party. Such obligations can derived from the law, but also from consent. Full coverage by a machine readable policy will never be achievable, as the law often requires human interpretation (that is why we have judges). Obligations should be ready to cover the scope of purpose limitations, which is difficult to translate, since it is hard to describe, whether an access, storage, or transfer was made for one or another purpose.

**R145 Constraint restrictions**

- Grant data usage: basic contextual information related to the data usage scope should be handled in order to add more granularity to the data usage constraints. E.g. some data can be displayed in some particular locations or domains and must be hidden otherwise.
- Limit data protection: "break the glass" systems must be supported in case of emergency. In this case all private data should be accessible for authorized users to manage crisis events. A policy language should support such exceptions limiting the scope of the privacy protection in case of emergency.

**R146 Notification/feedback channels**

Acknowledgement (ACK) messages should be sent to the data holder to inform him about the obligation enforcement conditions of his exported data. This is an important feature since actually few security systems are able to provide a report to the data holder about the usage of his private information. In case of misbehavior, this ACK can be used as a proof for accountability. Feedback channels can be used as a traceability system for studying the behavior of data handlers. Every data transaction and ACK message should share the same sequence identification. In case of problems (e.g. private data spread illegally), the ACK ID can be used to check for the traces and use it as a proof of privacy policy violation.

**4.2.2 Requirements for Access Control policies**

**R211 Declarative model to represent preferences**

The policy model should be accompanied by a language enabling the specification of policies on privacy. The language should be declarative and accompanied by a clear and unambiguous semantics for the policy specifications.

**R212 Inheritance; propagation (of credentials -- tokens)**

The policy model should provide explicit support for referencing and reasoning about credentials and tokens. Possible hierarchical organizations or other dependencies/derivations among credentials or properties within them could also be defined. Such hierarchies and relationships could be exploited by the policy model/language. For instance, mechanisms for the propagation of credentials among parties and for the derivation of credentials (e.g., if I have credential A for service S1 then credential B can be derived for service S2) should be provided.
R213 Role models - family, friends, wider access control
The access control model should support multiple access control paradigms, including role-based access control and attribute-based conditions. Roles could also be incorporated in attribute-based conditions by the consideration of proper attributes.

R214 Information from third party sources
The policy model/language should be able to leverage information certified by a given third party (e.g., government).

R215 Credentials or similar objects
The policy model should support expressions on attributes contained in digital credentials (see also R212). Different types of credentials may be integrated in the policy model/language, such as, for instance, anonymous credentials, X.509 credentials, pseudonym/password, Kerberos tickets, etc.

R216 Attribute based access control to data on fora
The policy model/language should support policies making explicit references to attributes of involved parties (e.g., requester of access, data on which access is requested, respondent/owner of data). Attribute values can be provided by means of credentials or can be metadata associated with objects.

R217 Expiration date: there should be an option for access control policies to expire after an amount of time
Access control policies should support conditions and reasoning about time. Time can impact the validity of certain conditions in the policies or be used to support policies that might be valid only up to some time or after some time (e.g., embargo on data, data that become public after a given time, data that should be deleted after a given time).

R218 Time or event of begin of validity
Access may be granted or denied for user or groups of users after an amount of time or after an event occurred (for instance to support history scientists etc.). The policy model may support event-based conditions other than those expressed as a time (see R217). Event-based conditions make policy restrictions/permissions become valid at the occurrence of certain events.

R219 Priority of policies or combination rules for policies
in case of contradicting policies we need a clear prioritization, that is a rule which determines which policy supersedes all others and how the others are combined with that policy and each other (you may think of a hierarchy of policies as well). The policy model should support a mechanism for combining policies according to different composition operators. The policy model/language should be accompanied by a clear definition of the possible composition operators and their semantics should be provided.

R220 User should be able to choose whether he wants to monitor
Users may be given the ability to monitor/track the releases of data referred to users (fulfilling the ‘notification’ principle). For instance, a user may want to monitor all the accesses to a blog. Monitoring/tracking can be provided by means of proper logging (active monitoring).

R221 Choose strength of protection
The policy model should provide different levels of protection and give users the ability to tune protection according to their need. For instance, requesting the application of specific cryptographic measure in communication or storage of private information.
R222 Change policy
In some scenarios, users can store/outsource their information to external servers. Access control policies can then be attached to the data to regulate their management. The users should be able to change/update those policies when needed.

R224 Property-based - challenge response, secret handshake
The policy model should allow the definition of rules and conditions to be verified and proved by means of verifiable encryption. A party should be able to demonstrate that a condition like “an attribute X is greater than a value Y?” is verified by only providing a chipertext and a proof that the condition is verified (hiding the real value). Also, a mechanism to decrypt the chipertext and access the real value should be provided in case of disputes or malicious users. This requires that the policy language supports:
- attributes
- predicates on attributes (e.g. yearlyIncomeEUR greater_than 50000)
- conditional request (for anonymous yet accountable access control); e.g., requesting the verifiable encryption of attributes to uncover in case of misconduct.

R225 Technical representation and description in user-centric terms
The policy model and the system architecture should be expressed in a form that allows standard users to understand them easily. The language for specifying access control policies and privacy preferences should be high level and easy to understand and use.

R226 Support for complex claims
The rules expressed in policies can take into account complex claims, i.e. certified attributes. For instance, claims can be identity (e.g. name), role (e.g. researcher), attribute (e.g. age), context (e.g. current location), relationship (e.g. friend/employee of Z), authorization (e.g. can read file X), rule (e.g. can delegate rights), etc.

R227 Support for complex rules
Access rules should allow expressive enough operators to compare and combine attributes (e.g. age>18 AND nationality = {German OR Swiss}) as well as simple requests for information (e.g. any user requesting the service must provide his phone number)

R229 Access requirements à la "provide third party X with attribute Y (but don't give it to me)"
Example usage:
- identity escrow: X = escrow server, Y = true identity
- privacy-friendly passing on of data, e.g. give address only to shipping company, not to webshop
The policy model should allow the definition of restrictions that regulate the access and the disclosure of information to other parties involved in a transaction. For instance, a bookshop, which relies on a shipping company for the shipment of the goods, should not access the address of the requester. This scenario should enforce the requirement of minimum disclosure.

R230 Use of ontologies
Ontologies can be a powerful tool to adapt the language to the particular needs of a particular context while maintaining interoperability. For example, a common ontology on the structure of personal identity information can be used to guarantee compatibility of digital passports issued by different countries. An ontology on countries and their
governments can be used to determine which instances are certified to issue passports for which countries.

4.2.3 Requirements for Trust policies

R311 Link to Data Handling policies
The policy language should provide the possibility to link a Data Handling policy with trust policies; this would permit the explicit representation of trust on the correct enforcement by the publisher of the Data Handling policy. When the binding with the trust policy is not expressed, trust on the publisher is implicitly assumed. The binding between Data Handling and Trust Policy could occur at different levels, expressing requirements on the subject/publisher (e.g., an identity must be provided, verified by a trusted identity manager) or on the source of the credentials referred in the policy (e.g., the publisher has to exhibit a valid certificate issued by the Better Business Bureau).

R312 Trust establishment
Several factors should be taken into account in the decision to trust another entity or not. A first could be the exchange of credentials ("I'll show you mine if you show me yours"). Trust could also be based on statements made by others, for example reputations or privacy seals.

R316 Statement and Certification
Certification validates that a server is authentic and trustworthy, so that the user can feel confident that their interaction with the server has not been eavesdropped and that the server is who it claims to be. The certificate is provided by a third party that should be trusted by the user.

- A trust statement is the explicit expression of a perceived trust level. It is made by the truster and represents the subjective judgment of the trustee's trustworthiness, according to the truster's point of view.
- A certificate is a digital document that describes a written statement from the issuer (certification authority, often considered trustworthy) about the identity of another party (in the form of public key and the identity of the owner) or a permission to use a specific service. It can be considered as a trust statement issued by a reputed third party.

R317 Make statement content-pluggable
Trust mechanisms have to configurable for different contexts. E.g., mechanisms used to assess trust for buying a book (e.g. they could rely on reputation only), are not typically the same for storing personal information (e.g. that may use certificates).

R318 Security breach
Security breaches come in different flavors, they include (but are not limited to):
- Loss of control on data (e.g., storage device stolen or lost)
- Unauthorized access control
- Social engineering
- Phishing
- Malicious proxy server

In some cases, legislation requires that when a privacy breach occurs, a notification has to be sent to the affected individual or organization. Technological means to enforce these rules have to be put in place, and a formal and quantitative definition of 'Privacy breach' assess to evaluate when the breach occurs and what is the level of risk associated.
R319 Link trust with Access Control
Conditions on policy may include trust evaluation, e.g. allow data writing if user complies with ACP and trust level greater than X or he has a certain certification.

R320 End user trust
Trust affects all levels of end-user interaction with the system, i.e. whenever a user wants to access to a service on the web. Trust should be assessed for all the layers involved in the transaction: user application, network, service provider. Trust being by definition related to a personal perception, each user has to be able to edit trust policies and trust preferences in an intuitive way.

R321 Building trust through a third party
Users may establish trust relationships using third party trust assessment. This may guarantee maximally a level of trust equal to the level of the certification authorities (best case scenario). Trust mechanisms have to support certificates as produced by certificate authorities (e.g., CACert, Thawte, etc.) and the corresponding hierarchical mechanisms (web of trust). Trust reasoning has to allow to combine this information with other trust metrics (e.g. reputation based).

R322 Trust reasoning
A trust policy evaluation component should be able to reason about trust, including composing various trust metrics (e.g. reputation system, KPI ..) and hierarchical structures. E.g. A can assess level of trust on B, combing information from a reputation system, certificates provided by B (or by a third party for B), measuring some indicators related to B (e.g., recent similar transactions). What kind of mechanisms could be used, and how to combine them (e.g., the relative weights in the combination) have to be context dependent, as well as modifiable by the user.

R324 Trust ontologies
To allow reasoning, a set of concepts commonly shared and agreed to by all involved parties (ontologies) have to be defined. However, combining different trust assessment mechanisms can be complex considering that typically just few of them (if any) are currently supported.

R323 Transparency, reciprocity
Transparency should be considered as one component of trust assessment (typically, transparency increases trust perception). Parameters may include: historical data, previous behaviors, access to log files, etc... . Reciprocity should be possibly taken in account as often characterised trust interaction, but this is not the general case. E.g. I trust a mail provider for storing my personal mails but it does not necessarily trust me for storing the same kind of information.

R329 Specification of liabilities
- Towards data subject: data protection obligations under the 95/46 Directive have to be fulfilled by data controllers. Data controllers are liable for data protection violations unless they can prove they are not responsible for the damage. It is necessary to differentiate between data controller and data processor. The role of data processor is reduced; he solely processes personal data as directed by the controller. The policy language should be able to express the role of each entity for each action to determine who is liable. Liability: compensation from the controller for the damage suffered. Remedy: a right to a judicial remedy for any breach of guaranteed rights. Sanctions: to be defined by member states in their national laws.
• Towards relaying parties: for ensuring data accuracy the following policies are important: validation of data at the moment of collection, procedures for reporting and dealing with suspected inaccuracies, regular updates, restriction of modification rights to authorized entities.

4.3 Requirements for Policy Composition

These are particular requirements when composing different policies. "Composition" can be understood within a larger scope than service composition, i.e. any kind of interaction between policies may imply requirements listed here.

4.3.1 Composition of Data Handling policies

RC11 Prior agreement and contract
Technical policies generally implement legal requirements (contract, directive, agreement, etc.). The technical policy should contain a pointer to or the full text of the source document (i.e. legal requirements). Regarding privacy policies this requirement is addressed in more detail in R113.

RC12a Aggregation of policies
The policy language should enable aggregation of policies when multiple policies from different sources refer to a specific piece of data. Aggregation, in this context, refers to a combination of policies that does not result in a conflict. In practice aggregation is represented by a union of a set of policies. Enforcing this set of policies should not result in incoherent behavior.

RC12b Combination of policies
The policy language should enable combination of policies when combining multiple pieces of data from different sources with associated policies. Combination, in this context, refers to handling conflicting rules. In simple cases, combination can be automated as the intersection of rights and union of obligations. Depending on the scenario, combination may result in more restrictive or less restrictive policies. For instance, depending on the manner in which conflict is resolved (ex. prioritisation, weighting), some of the conflicting rules may be deleted.

RC13a Cascading policies
When rules are defined at different levels (e.g. corporate, service, and action), mechanisms to select and aggregate appropriated rules should be provided.

RC13b Prioritisation of rules
Priorities are only necessary to resolve conflicts between rules. Depending on the expressiveness of the language, especially regarding the support of negations, priorities may be required.

RC14 Generalization of policies
Composite services rely on external services. Policies of composite services depend on properties of external services, which may be dynamically selected. The policy language should support such dynamic scenarios and offer a way to express general statements. For instance, instead of saying that "personal data could be shared with bank X, which deletes data after 6 months", the policy could state that "personal data could be shared with any bank in Europe that commits to delete before one year". Similarly, generalization of user preferences should be envisioned.
RC15a Multi-rounds policy definition
When associating a policy with some personal data, purely service-driven (e.g. P3P) or purely user-driven (e.g. arbitrary policy pushed by the user to the service) approaches do not work. A tradeoff letting the user customize some optional parts of a template should be possible. Such a mechanism already exists in PRIME-DHP. Multi-rounds could be added to let the service call back the user when an optional part was deliberately let open.

RC15b Policy negotiation
Negotiation only makes sense when the user and/or the service have a trade-off to make. (Without a trade-off, the optimal choice for both parties is deterministic and can be automatically deduced.) For instance, quality of service (features, speed, etc.) or service fees could change depending on the privacy (quantity and/or quality of released personal information). The policy language should be able to express what the trade-offs are.

4.3.2 Composition of Access Control policies

RC21 Delegation of Rights
Service providers (e.g. data owner) need a way to grant access rights to other parties (e.g. A says B can read dataZ). Delegation of rights is the ability to let another party provide rights (e.g. A says B can say x can read dataZ, B says C can read dataZ). This is a key feature of SPKI and SecPAL that enables complex distributed access control settings.

RC22 Revocation of Rights
A party providing rights must have a way to revoke them. Depending on the required time granularity, this can be achieved by periodically deciding to renew rights or by explicitly revoking (e.g. Certificate/Credential Revocation List).

RC23 Composition of Access Control Policies
The policy language should enable the creation of composite applications relying on different services. For instance, Single Sign On, federations, or delegation of rights should be in place to enable coherent enforcement of authentication and authorization policies associated with different services. See also requirements RC12a and RC12b on aggregation and combination of ACPs.

RC24 Prior agreement and contracts
Similar to RC11, applied to access control policies.

RC25 Privacy-aware audit mechanism
Access control policy language must offer support for logging mechanisms. The policy must offer mechanisms to specify privacy aspects of logs. For instance, it should be possible to specify that denied accesses are logged for statistics but that the identity of the requestor is not part of the log.

RC26 Support for data and PII
Legislations treat personal data (PII) differently from other types of data (e.g. internal notes, trade secret). The developed policy language must support specific requirements of PII (e.g. purpose). This may be solved by more generic mechanisms.

RC27 Link between AC and DH
Data Handling policy of the data owner impacts Access Control policy of the data consumer. In other words, let’s assume that A sends some PII to B with attached data
handling sticky policy. If B lets C access the PII provided by A, the access control of B regarding this PII must reflect A’s sticky policy. This is already illustrated in PRIME-DHP.

4.3.3 Composition and Trust policies

**RC31 Dynamic Trust**
Trust is not static: mechanisms to bootstrap, modify, and revoke trust are necessary.

**RC32 Scope**
Trust is not unconditional. The scope of the trust relationship has to be defined. A user will trust a medical association in the scope of health but not in the scope of finance.

**RC33 Proof of enforcement**
Trust aims at deciding whether a party will behave as expected. In the specific case of privacy and data handling policies, trust is necessary to decide whether a party will handle PII as specified in the data handling policy. Mechanisms to prove that a trusted platform (e.g. TCG) and/or audit mechanisms are in place are required.

4.4 Requirements for use of anonymous credentials

**RP01 Technology-independent certification of data by trusted third parties**
The policy language should be able to express access, trust, and data handling requirements over information that was certified by an external, trusted third party. The language should depend as little as possible however on exactly which underlying technology is used to certify this information. That is, the information could be certified by a trusted LDAP server (username/password), Kerberos tickets (symmetric crypto), X.509 certificates (asymmetric crypto), anonymous credentials (= advanced asymmetric crypto), … The policy language should make abstraction of the underlying technology and speak in terms of general concepts that are common to all technologies. Yet still, the language should be expressive enough to fully leverage the different capabilities of each of the technologies.

**RP02 Trust in certified data**
The policy language needs some elements to state what the requirements on a piece of data are in order to trust the authenticity of this piece of data. This can be done by stating who is trusted to certify this kind of data, i.e. by stating who the allowed issuer(s) of a certificate for this data are. For example, if a policy states that one has to reveal her last name, then the policy also needs to specify who is trusted to certify this last name (e.g., this might be a government, a city authority, etc).

**RP03 Predicates over attributes, extensible with ontologies**
The ACP should allow to express complex predicates over attributes, including AND/OR combinations, equality, greater/less-than, simple arithmetic, … These predicates should be extensible with new ontologies too. For example, if frequent flyer statuses are such that platinum is higher than gold, then the policy should be able to express ‘status>gold’.

**RP04 Embedded or referenced ontologies**
It should be possible to have the ontologies that are used in a policy contained in the policy itself. An alternatively approach would be to reference to the ontologies, i.e., to point to some resource/location where the ontology can be obtained. However, the
approach of referencing might lead to problems during an online policy evaluation, if the ontology for any reason cannot be obtained.

**RP05 Expression of proved statement (by using same policy language)**
The user should also be able to express the statement that he is proving, preferably using the same language. This statement may be different from the statement required by the server, but it should imply it. The server’s policy engine needs to be intelligent enough to check whether the statement proved by the user actually implies the access conditions.

**RP06 Derived PII**
One issue needing more thought is how one defines the DHP of derived PII. With derived PII we mean sensitive information that is computed based on actual PII. For example, one could associate a DHP to one’s exact date of birth, but what then is the DHP for just your year of birth, your zodiac sign, or the mere fact that you’re over 21? Are these bound to the same DHP? It seems natural to loosen the DHP as the quality of the information degrades, but quantifying this is highly non-trivial. For example, one could be tempted to say that by proving you’re over 21 you give 1 bit of information away, but it’s actually more complicated than that: by proving you’re over 0 you give no information away, by proving you’re over 115 you give quite a lot away since the number of people over 115 is probably very low. And what is the DHP on PII that is derived from multiple pieces of PII? A conservative solution would be to let the DHP of the derived information be the “intersection” of the individual DHPs, meaning allowing only those actions that are allowed by all DHPs.

**RP07 Revealing of data**
The policy language should clearly distinguish between, on the one hand, the PII that the requestor has to reveal in order to gain access to a service, and, on the other hand, the restrictions that the PII have to satisfy to be granted access. This difference is often neglected in existing languages, probably because the only way to prove a statement about an attribute using "classical" technologies is by revealing its exact value. Anonymous credentials, however, do allow to keep these very different types of access requirements separate, with great privacy benefits for the requestor.

To illustrate this difference, think of a service where users have to be over 18 and submit their phone number. The service is not interested in the exact value of the user's birthdate; rather, it wants to ensure the mere fact that it is longer than 18 years ago. On the other hand, the server does want to know the full telephone number, but there are no conditions associated to the number that could influence the access decision (other than the number being valid, perhaps).

**RP08 Alternative data recipient + associated access conditions**
It should also be possible to express the recipient of the information, i.e. to whom this information should be sent. By default this is probably the server itself, but in certain applications the server may want to be assured that the client sent some specific piece of information (e.g., her address for shipping) to someone else. When the recipient is not the server itself but a third party, then it should also be specified under what conditions the third party should reveal the information. This can probably be implemented as an ACP on the transmitted information.

**RP09 Notion of atomic credentials**
The language should support the concept of an atomic credential (or information card, or whatever one wants to call it) as a collection of certified attributes. This is important to prevent “mixing-and-matching” of attributes from several credentials in cases one has
multiple credentials of the same type and by the same issuer. For example, if I have two credit cards, then I shouldn’t be able to complete the checkout transaction by revealing the number of one card and the expiration date of the other card.

RP10 Macros
The policy language should have some sort of macro language to increase readability. For example, in the buying use case, the macros ‘anygovt’ and ‘goodbusiness’ act as easy links from the access control part into the trust part of the policy. In general, use of macros can avoid repetition throughout the policy, and make the policy layout more modular. In particular, it provides an easy way to group credential issuers into classes.

RP11 Limited spending
The policy language should have provisions to express that one can only authenticate oneself with the same credential for a limited number of times. There are different ways to calculate the number of shows: overall, at this website, at this service, on this day, etc. However, it is not clear whether the number of times a credential has been shown is an attribute of a credential or not, since it might depend on some “context” information like e.g. the time of day, the weather, the price of a stock, etc.

RP12 Alternative DHP
It might be the case that a party does/can not accept the DHP of another party. For example, the party has not the technical means to fulfill the required obligations. The policy language should then be able to express an alternative DHP that it is willing/able to fulfill.

RP13 Choice based Access Control Policy
The policy language should allow for making access control requirements on a resource dependent on (multiple) choices the user can make. For example consider a credit card application, where the access control requirements depend on choosing the monthly limit the credit card shall have. When choosing a limit of 2000 Euro, different access control requirements apply than with a choice of 5000 Euro.
References

[Art.29 Opinion 100]

[Art.29 Opinion 100 Annex]

[Art.29 Opinion 136]

[D6.2.1]

[Directive 2002/58/EC]

[Directive 95/46/EC]

[P3P 1.0 Specification]
The Platform for Privacy Preferences 1.0 (P3P1.0) Specification, 16 April 2002.
http://www.w3.org/TR/P3P

[PLING]
http://www.w3.org/Policy/pling/wiki/UseCases